НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

До захисту допущено:

Завідувач кафедри

_____ Олександр РОЛІК

«___»_____2025 р.

Дипломний проєкт

на здобуття ступеня бакалавра
за освітньо-професійною програмою «Інформаційні управляючі системи та технології»
спеціальності 126 «Інформаційні системи та технології»
на тему: «Система обліку фінансових операцій для малого бізнесу»

Виконала:
студентка IV курсу, групи ІС-11
Івасішина Анна Григорівна _____

Керівник:
д.т.н., професор кафедри
Поліщук Михайло Миколайович _____

Рецензент:
Доцент кафедри ОТ, к.т.н., доц.
Волокита Артем Миколайович _____

Засвідчую, що у цьому дипломному проєкті немає запозичень з праць інших авторів без відповідних посилань.
Студентка _____

Київ – 2025 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 126 «Інформаційні системи та технології»

Освітньо-професійна програма «Інформаційні управляючі системи та технології»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Олександр РОЛІК
«___»_____2025 р.

ЗАВДАННЯ
на дипломний проєкт студентці
Івасішиній Анні Григорівні

1. Тема проєкту «Система обліку фінансових операцій для малого бізнесу», керівник проєкту Поліщук Михайло Миколайович, д.т.н., професор кафедри затверджені наказом по університету від «23» травня 2025 р. № 1705-с

2. Термін подання студентом проєкту: 9 червня 2025 р.

3. Вихідні дані до проєкту: мова програмування Python (з використанням бібліотеки CustomTkinter для UI); Фреймворк / бібліотеки: CustomTkinter (модернізований Tkinter для сучасного UI), PIL (Pillow) для роботи з зображеннями, SQLite3 — база даних, Matplotlib — для графіків і візуалізації); n8n — платформа для автоматизації робочих процесів; Telegram Bot API — для організації взаємодії користувача з системою; Архітектурний шаблон: MVC-подібна архітектура з розділенням на Model

4. Зміст пояснювальної записки: опис предметної області, аналіз існуючих рішень, формування вимог до системи, вибір технології розроблення, розроблення інформаційної системи, математичне забезпечення, тестування системи.

5. Перелік графічного матеріалу (із зазначенням обов’язкових креслеників, плакатів, презентацій тощо):

1. Діаграма варіантів використання
2. Діаграма структури додатку
3. Діаграма «сутність – зв’язок»
4. Діаграма послідовностей

6. Дата видачі завдання: 03.03.2025

Календарний план

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1	Аналіз предметної області	15.03.2025	
2	Огляд існуючих рішень	20.03.2025	
3	Розроблення функціональної моделі	25.03.2025	
4	Формування вимог до системи	30.03.2025	
5	Вибір засобів та методів реалізації	05.04.2025	
6	Проєктування та реалізація програмного забезпечення	05.05.2025	
7	Розроблення математичного забезпечення	15.05.2025	
8	Тестування системи	20.05.2025	
9	Оформлення пояснювальної записки	30.05.2025	

Студентка Анна ІВАСІШИНА

Керівник Михайло ПОЛІЩУК

АНОТАЦІЯ

Система обліку фінансових операцій для малого бізнесу.

Проєкт містить 71 с. тексту, 26 рисунки, посилання на 17 літературних джерел, додатки та 4 конструкторські документи.

ОБЛІК ФІНАНСІВ, МАЛИЙ БІЗНЕС, ІНФОРМАЦІЙНА СИСТЕМА, АВТОМАТИЗАЦІЯ, PYTHON, SQLITE.

Об’єктом розробки є інформаційна система для обліку фінансових операцій у сфері малого бізнесу.

Мета розробки — підвищення ефективності ведення фінансового обліку за рахунок автоматизації запису, обробки та аналізу фінансових даних.

У дипломному проєкті реалізовано програмну систему, що дозволяє фіксувати доходи, витрати та перекази, групувати дані за категоріями, аналізувати фінансові потоки за обраними періодами, а також будувати графіки змін. Система реалізована мовою програмування Python із використанням бази даних SQLite та бібліотеки Matplotlib для візуалізації.

Проведено аналіз існуючих програмних рішень, обґрунтовано вибір технологій, розроблено структуру бази даних, описано математичну модель та алгоритми обробки транзакцій. Особливу увагу приділено простоті інтерфейсу, автономності роботи без підключення до Інтернету та можливості масштабування.

Для розширення функціоналу реалізовано Telegram-бота на основі платформи n8n, який надає користувачам цілодобову підтримку у вигляді фінансового консультанта.

Отримані результати можуть бути використані в малих підприємствах для організації базового фінансового обліку, а також як основа для подальшого розвитку або інтеграції в більші бізнес-системи.

SUMMARY

Financial operations management system for small business. The project contains 71 pages of text, 26 figures, references to 17 literary sources, appendices, and design documents.

KEYWORDS: FINANCIAL ACCOUNTING, SMALL BUSINESS, INFORMATION SYSTEM, AUTOMATION, PYTHON, SQLITE.

The object of development is an information system for managing financial operations in the field of small business.

The purpose of the development is to improve the efficiency of financial accounting through the automation of data entry, processing, and analysis.

The graduation project presents a software system that allows recording incomes, expenses, and transfers, grouping data by categories, analyzing financial flows over selected periods, and generating visual reports. The system is implemented using the Python programming language, the SQLite database, and the Matplotlib library for visualization.

An analysis of existing software solutions was carried out, technologies were selected and justified, the database structure was developed, and the mathematical model and transaction processing algorithms were described. Particular attention was paid to interface simplicity, offline functionality, and scalability potential.

To expand the functionality, a Telegram bot based on the n8n platform was implemented, providing users with round-the-clock support in the form of a financial advisor.

The obtained results can be applied in small enterprises for basic financial management, and the system may serve as a foundation for further development or integration into broader business platforms.

Номер рядка	Формат	Позначення	Найменування	Кільк. аркушів	Номер екзем.	Примітка
1			Документація загальна			
2						
3			Знову розроблена			
4						
5	А4	ІС11.120БАК.005 ПЗ	Пояснювальна записка	71		
6	А3	ІС11.120БАК.005 Д1	Система обліку фінансових	1		
7			операцій для малого бізнесу.			
8			Діаграма варіантів			
9			використання			
10	А3	ІС11.120БАК.005 Д2	Система обліку фінансових	1		
11			операцій для малого бізнесу.			
12			Діаграма розробленого			
13			алгоритму			
14	А3	ІС11.120БАК.005 Д3	Система обліку фінансових	1		
15			операцій для малого бізнесу.			
16			Діаграма «сутність – зв’язок»			
17	А3	ІС11.120БАК.005 Д4	Система обліку фінансових	1		
18			операцій для малого бізнесу.			
19			Діаграма послідовностей			
20						
21						
22						
23						
24						
25						
26						
27						
28						

Зм.	Аркуш	№ докум.	Підпис	Дата	ІС11.120БАК.005 ТП		
Розроб.		Івасішина А.Г.			Система обліку фінансових операцій для малого бізнесу. Відомість проєкту	Літ.	Аркуш 1 / Аркушів 1
Керівн.		Поліщук М.М.				КПІ ім. Ігоря Сікорського	
Затв.							

Пояснювальна записка
до дипломного проєкту
на тему: «Система обліку фінансових операцій для малого бізнесу»

Київ – 2025 року

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

GUI — *(англ. Graphical User Interface)* — графічний інтерфейс користувача; форма взаємодії користувача з програмою через вікна, кнопки та інші елементи.

SQLite — *(англ. Structured Query Language Lite)* — легка вбудована система управління реляційними базами даних, що не потребує серверної частини.

Python — високорівнева мова програмування загального призначення з відкритим вихідним кодом, яку було використано для реалізації програмної частини застосунку.

Tkinter — стандартна бібліотека Python для створення графічного інтерфейсу користувача (GUI), що дозволяє будувати вікна, кнопки, поля введення тощо.

Matplotlib — бібліотека для мови Python, яка використовується для побудови графіків, діаграм та візуалізації статистичних даних.

CustomTkinter — розширення стандартної бібліотеки Tkinter з підтримкою сучасного дизайну інтерфейсу, темної теми та кастомізації елементів.

CSV — *(англ. Comma-Separated Values)* — формат зберігання табличних даних у текстовому вигляді з розділенням значень комами або крапками з комами.

.exe — виконуваний файл у системі Windows, що запускає програму.

Dashboard — інтерактивна панель управління, яка відображає ключову інформацію про фінансові показники в одному вікні.

БД — база даних; структура, у якій зберігаються транзакції, категорії, контрагенти та інші елементи системи.

JSON — *(англ. JavaScript Object Notation)* — формат обміну даними у вигляді тексту, зручний для збереження та передачі структурованої інформації.

Entity — сутність; логічна одиниця в базі даних (наприклад, транзакція або категорія).

n8n — платформа для автоматизації робочих процесів, що дозволяє інтегрувати різні сервіси та налаштовувати логіку обробки даних.

ВСТУП

Інформаційні технології сьогодні стали невід'ємною частиною ефективного управління в усіх сферах діяльності, зокрема у сфері фінансів. Однією з таких сфер, де автоматизація процесів має особливо важливе значення, є облік фінансових операцій. Ефективне управління фінансами вимагає не лише точності, але й оперативності в обробці даних, аналітичних можливостей та зручного інтерфейсу для користувача. Це актуально як для великих підприємств, так і для суб'єктів малого бізнесу, які часто не мають достатніх ресурсів для використання складних бухгалтерських систем чи послуг окремих фахівців.

Малий бізнес, як важлива складова економіки будь-якої країни, зокрема України, постійно стикається з викликами, пов'язаними з обмеженістю фінансів, нестабільністю ринку, змінами у законодавстві та браком кваліфікованих кадрів. У таких умовах надзвичайно важливо забезпечити прозорий, зручний і доступний облік фінансових операцій, що дозволить власнику бізнесу контролювати витрати, планувати бюджети, виявляти точки фінансових втрат і приймати обґрунтовані управлінські рішення. Проте на практиці багато підприємців усе ще ведуть облік у зошитах, блокнотах або нескладних Excel-таблицях, що обмежує можливості аналізу даних і створює ризик помилок, втрати інформації чи дублювання транзакцій.

Актуальність даної теми зумовлена необхідністю розробки інструментів, які б відповідали специфічним потребам малого бізнесу — були простими, наочними, інтуїтивно зрозумілими, але водночас достатньо функціональними для базової фінансової аналітики. Особливої ваги набуває локальність системи, оскільки вона не повинна вимагати інтернет-з'єднання чи високих технічних вимог до комп'ютера користувача. Ідеальним рішенням у такому випадку є створення інформаційної системи, яка базується на сучасних, легких і безкоштовних інструментах — зокрема, мові програмування Python та вбудованій реляційній базі даних SQLite.

Дипломна робота присвячена розробці системи обліку фінансових операцій для малого бізнесу. Основна ідея полягає у створенні локального застосунку, що

дозволяє користувачеві легко реєструвати фінансові надходження та витрати, групувати їх за категоріями, зберігати інформацію у базі даних, переглядати статистику за певний період та будувати візуалізацію у вигляді графіків і діаграм. Особливу увагу в процесі розробки було приділено зручності використання, наочності відображення фінансових результатів та можливості подальшого розширення функціоналу.

Програмне рішення, створене в межах даної роботи, не потребує встановлення додаткового складного програмного забезпечення, а також не вимагає спеціальної технічної підготовки з боку користувача. Воно орієнтоване на підприємців, які бажають отримати зрозумілий і швидкий доступ до фінансових даних свого бізнесу без зайвих витрат часу й коштів. Також система може бути використана для персонального обліку фінансів, у стартапах, фрілансерському середовищі або невеликих командах, що ведуть спільну діяльність.

Метою даної дипломної роботи є проєктування та створення інформаційної системи, яка автоматизує процес обліку фінансових операцій для малого бізнесу, використовуючи мову Python і базу даних SQLite. Система повинна забезпечувати реєстрацію транзакцій, збереження та обробку фінансових даних, формування звітності, аналіз структури витрат і доходів, а також візуалізацію фінансових показників у вигляді графіків.

Для досягнення цієї мети в роботі розглянуто теоретичні основи побудови інформаційних систем, виконано аналіз предметної області, проаналізовано існуючі програмні рішення у сфері фінансового обліку, визначено технічні та функціональні вимоги до системи. У подальшому було обґрунтовано вибір інструментів розробки, спроєктовано архітектуру майбутньої системи, реалізовано програмну частину, проведено тестування основних функціональних модулів і здійснено аналіз результатів.

Результатом виконаної роботи є готовий програмний продукт, який дозволяє здійснювати облік фінансових операцій в інтуїтивно зрозумілому середовищі, швидко отримувати інформацію про фінансовий стан підприємства, створювати звіти

за обраними параметрами та візуалізувати дані. Одержані результати підтверджують практичну цінність запропонованої системи та її відповідність потребам цільової аудиторії.

Таким чином, розроблена інформаційна система є прикладом ефективного поєднання програмної інженерії, базових принципів бухгалтерського обліку та користувацько-орієнтованого дизайну, що разом формують практичний і функціональний інструмент для щоденного використання в умовах малого бізнесу.

Окрім цього, реалізоване рішення створює основу для подальших розробок у напрямку автоматизації бізнес-процесів. У майбутньому система може бути доповнена модулями інтеграції з податковими сервісами, інструментами прогнозування фінансових потоків, а також мобільним застосунком для ще більшої зручності користувачів. Таким чином, дана робота не лише задовольняє поточні запити ринку, а й закладає потенціал до масштабування та адаптації під ширший спектр підприємницьких потреб.

1 ОПИС ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Опис процесу діяльності

У сучасному бізнес-середовищі ефективне управління фінансовими ресурсами є одним із ключових чинників стабільної роботи підприємства. Незалежно від масштабів діяльності, фінансовий облік дозволяє контролювати грошові потоки, оптимізувати витрати, оцінювати прибутковість та забезпечувати прозорість у прийнятті рішень [16]. Особливо це стосується суб'єктів малого підприємництва, які часто не мають окремого фінансового відділу чи бухгалтера і змушені самостійно вести облік доходів та витрат. У таких умовах будь-яка помилка в обліку або недооцінка динаміки фінансів може призвести до серйозних наслідків, включаючи збитки, прострочення податкових платежів, втрату контролю над залишками коштів чи неправильне планування майбутніх витрат.

Процес фінансової діяльності у малому бізнесі включає реєстрацію всіх грошових операцій — як надходжень, так і витрат, — ведення внутрішнього бюджету, оцінювання фінансових результатів за різні періоди, аналіз витрат за категоріями, зберігання історії транзакцій та формування звітів. У найпростішому вигляді ці процеси реалізуються у паперових записах або у вигляді електронних таблиць, проте такий підхід не забезпечує ефективної аналітики, масштабованості та безпеки даних. Ручне введення і відсутність структури часто призводять до втрати інформації, дублювання записів або складнощів з її подальшою обробкою. Крім того, користувачі, які не мають спеціальних знань у бухгалтерії чи аналітиці, можуть відчувати труднощі при самостійному аналізі даних. Саме тому зростає потреба в простих, доступних і надійних інструментах, які б автоматизували ключові функції обліку.

З огляду на це, доцільним є створення автоматизованої системи, яка б підтримувала базовий фінансовий облік, дозволяла зручно і швидко вводити нові транзакції, зберігала їх у структурованому вигляді, забезпечувала візуалізацію результатів і мала мінімальні вимоги до ресурсів. Така система має спрощувати роботу

користувача, зменшувати ймовірність помилок, надавати можливість перегляду поточного фінансового стану та динаміки змін. Важливо також, щоб інтерфейс був інтуїтивно зрозумілим, а логіка роботи — прозорою.

1.2 Постановка задачі

У межах цього дипломного проєкту ставиться задача створення простої, проте ефективної інформаційної системи, яка дозволяє автоматизувати облік фінансових операцій у малому бізнесі. Для цього необхідно проаналізувати наявні підходи до вирішення подібних задач, сформулювати функціональні вимоги до майбутнього програмного продукту, обрати відповідні інструменти реалізації та побудувати систему, яка дозволить користувачу зберігати, обробляти й аналізувати фінансову інформацію без залучення стороннього програмного забезпечення або додаткового навчання.

Розробка такої системи має на меті покращення процесів фінансового обліку на рівні мікропідприємств та індивідуальних підприємців. Програмний продукт повинен охоплювати ключові етапи обробки фінансових даних: від введення транзакцій до побудови фінансової звітності й візуалізації динаміки витрат і доходів. Уся інформація має зберігатися в локальній базі даних з можливістю подальшого резервного копіювання, фільтрації та аналізу.

1.3 Призначення системи

Призначенням системи є автоматизація фінансового обліку в умовах малого бізнесу, де відсутні спеціалізовані бухгалтерські інструменти або штатні бухгалтери. Вона призначена для щоденного використання підприємцями, менеджерами або працівниками, відповідальними за фінанси, й охоплює діяльність, пов’язану з введенням фінансових операцій, збереженням записів, формуванням зведених звітів, аналізом витрат і контролем доходів.

Об’єктами автоматизації є процеси ведення обліку надходжень та витрат, їх класифікації за категоріями, обчислення сумарних показників, формування звітності за часовими інтервалами, а також представлення результатів у вигляді графіків. Таким чином, система охоплює повний цикл внутрішнього фінансового моніторингу на підприємстві з мінімальними витратами ресурсів.

1.4 Цілі та задачі розробки

Метою розробки є спрощення фінансового обліку в малому бізнесі шляхом створення інструменту, який дозволить покращити точність, швидкість і зручність роботи з фінансовими даними.

У межах дипломного проєкту передбачається вирішення наступних завдань: розробити структуру бази даних для зберігання фінансових транзакцій та категорій витрат і доходів; обґрунтувати вибір мови програмування та середовища розробки для реалізації системи; провести аналіз сучасних аналогів систем фінансового обліку; спроєктувати інтерфейс користувача з урахуванням принципів зручності та простоти; реалізувати програмний модуль для введення, редагування, видалення та перегляду фінансових записів; розробити механізм формування звітів за обраними параметрами; реалізувати графічне відображення динаміки фінансів; протестувати систему на коректність обробки даних і стабільність роботи.

Висновки до розділу 1

Предметна область, на основі якої реалізується розробка, охоплює ключові процеси фінансової діяльності малого підприємства, зокрема облік витрат і доходів, збереження даних, їх обробку та аналіз. Запропонована система має на меті автоматизацію обліку, підвищення точності й ефективності обробки інформації, а також спрощення доступу до фінансових показників підприємства.

2 АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ

На сьогодні ринок програмного забезпечення пропонує велику кількість рішень, що автоматизують процеси фінансового обліку в компаніях. Однак більшість із цих продуктів розроблені з урахуванням потреб середнього та великого бізнесу, і вимагають значних ресурсів для впровадження, обслуговування та користування. У той же час малий бізнес, який має обмежений бюджет, невеликий штат і часто відсутність фахівців у галузі бухгалтерського чи фінансового програмного забезпечення, потребує максимально простих, доступних і ефективних рішень.

У цьому розділі буде розглянуто три найбільш поширені програмні рішення, які частково використовуються представниками малого бізнесу в Україні: Finmap, Fredo, Wave Accounting.

2.1 Finmap

Finmap — це український онлайн-сервіс, що позиціонується як рішення для обліку фінансів малого бізнесу та підприємців [11]. Основною метою цього програмного забезпечення є спрощення контролю за грошовими потоками підприємства, зокрема облік доходів і витрат, аналіз фінансових результатів, прогнозування залишків на рахунках та управління бюджетами. Сервіс пропонує зручний інтерфейс, доступний як у браузерній версії, так і на мобільних пристроях. Finmap підтримує мультивалютність, інтеграції з банками, а також командну роботу, що дозволяє залучати кількох користувачів з різними правами доступу.

Головна панель Finmap містить найважливішу інформацію — залишки по всіх рахунках підприємства, перелік проведених і запланованих транзакцій, контрагентів, категорій та коментарів до кожної операції. Завдяки цьому користувач може швидко оцінити фінансову ситуацію, простежити рух коштів та виявити проблемні моменти. Інтерфейс реалізовано у вигляді зручної таблиці з фільтрами та кольоровими позначеннями, що дозволяють моментально виокремити критичні ви-

трати або доходи. Наприклад, можна відфільтрувати лише готівкові операції, переглянути платежі за певною категорією або вивести лише заплановані надходження на конкретну дату.

Додатково в панелі користувач має змогу переглядати аналітику — графіки доходів та витрат за обраний період, динаміку залишків по рахунках, а також отримати візуальне порівняння фактичних і запланованих показників. Це дає змогу приймати більш обґрунтовані управлінські рішення, наприклад, оптимізувати витрати, скоригувати бюджет або змінити графік платежів.

На рисунку 2.1 зображено головну панель Finmap з прикладом обліку операцій на кількох рахунках, серед яких банківські карти, готівка в різних валютах та корпоративні рахунки.

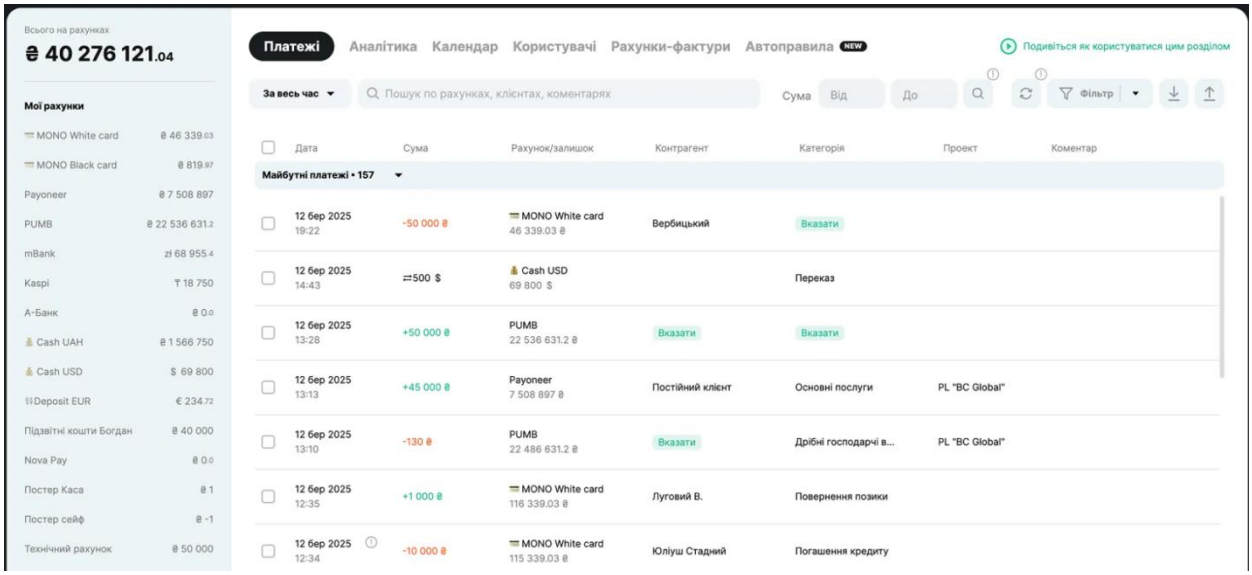


Рисунок 2.1 – Головна панель Finmap з відображенням рахунків і транзакцій

Однією з найзручніших функцій системи є можливість створення нових фінансових операцій — доходів, витрат або переказів між рахунками. Інтерфейс створення транзакції інтуїтивно зрозумілий: потрібно лише обрати рахунок, суму, категорію, контрагента, а також дату надходження або списання коштів. За потреби користувач може зробити транзакцію повторюваною, додати тег, коментар чи прикріпити файл. На рисунку 2.2 наведено приклад вікна створення нового доходу.

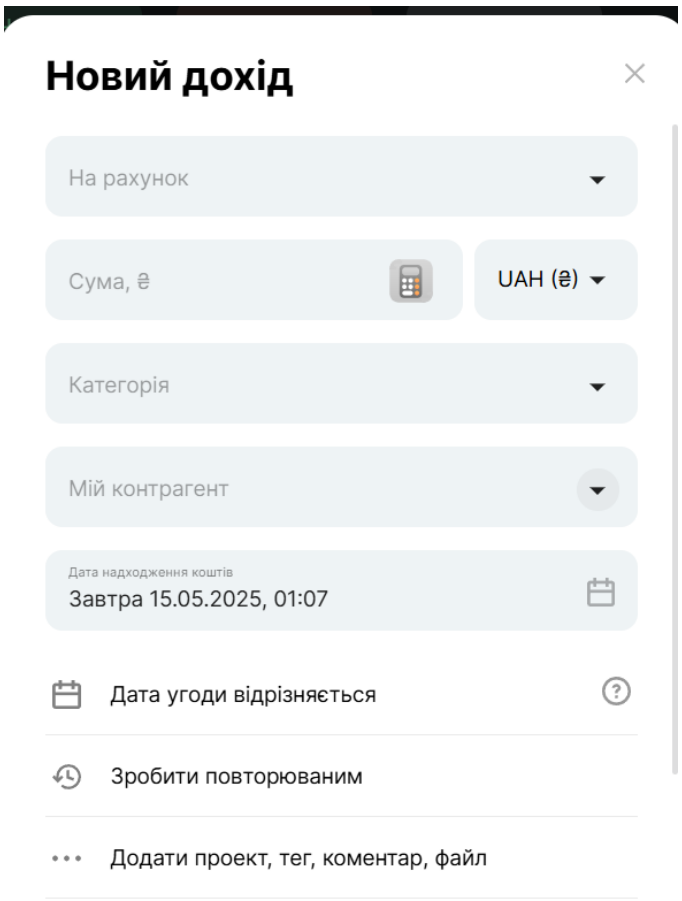


Рисунок 2.2 – Вікно створення доходу у Finmap

Попри значну кількість переваг, Finmap має і низку обмежень. Насамперед, це хмарна система, тобто всі дані зберігаються на сторонніх серверах. Для деяких користувачів це може становити проблему через міркування конфіденційності або відсутність постійного доступу до Інтернету. Також варто зазначити, що повноцінний функціонал Finmap доступний лише у платній версії — безкоштовна версія обмежена за кількістю рахунків, аналітичних інструментів та користувачів. Інтерфейс хоч і адаптований для українського ринку, проте в деяких розділах він може здаватися перевантаженим, особливо для користувачів без досвіду роботи з фінансовими системами.

Таким чином, Finmap можна вважати зручним та сучасним рішенням, що охоплює більшість базових потреб малого бізнесу. Водночас залежність від хмарної інфраструктури та платна модель доступу до розширеного функціоналу суттєво обмежують застосування цього сервісу у випадках, коли користувач прагне мати

повний контроль над своїми даними або використовувати програму без додаткових витрат.

2.2 Fredo

Fredo — це веб-сервіс, який інтегрується в екосистему українських бухгалтерських систем і активно використовується в комерційній практиці [12]. Його основне призначення — електронна звітність, документообіг, інтеграція з обліковими системами, зокрема BAS. Fredo підтримує контроль платежів, обмін первинними документами, облік податкових накладних, звітність до контролюючих органів. Це рішення є актуальним для бухгалтерів, які працюють із великою кількістю юридичних операцій і повинні дотримуватися чинного законодавства України.

Проте система Fredo складна у використанні для осіб без спеціальної підготовки. Вона вимагає попереднього навчання, певного досвіду з бухгалтерським ПЗ, а також має залежність від оновлень і підключення до Інтернету. Крім того, цей сервіс є платним і має абонентську модель користування. Інтерфейс орієнтований на фахівців, а не на підприємців, які самостійно ведуть облік. Для малого бізнесу, де облік фінансів часто здійснюється однією людиною без знань у бухгалтерії, використання Fredo є надто складним та обтяжливим.

Отже, попри свою технічну потужність, система Fredo не є оптимальним вибором для малих підприємців, які не мають бухгалтерської освіти чи доступу до фахівців. Високий поріг входу, необхідність регулярного оновлення та залежність від Інтернету знижують її придатність для використання в умовах малого бізнесу.

2.3 Wave Accounting

Wave Accounting — безкоштовний онлайн-сервіс, розроблений переважно для ринку США та Канади [13]. Його функціонал охоплює базовий облік доходів і витрат, виставлення рахунків, звітність, контроль за грошовими потоками та сповіщення. Wave приваблює тим, що є повністю безкоштовним, має зрозумілий

інтерфейс і можливість підключення банківських рахунків (у підтримуваних регіонах). Він орієнтований на фрілансерів та малий бізнес, які не потребують складних бухгалтерських операцій. На рисунку 2.3 зображено головну панель сервісу.

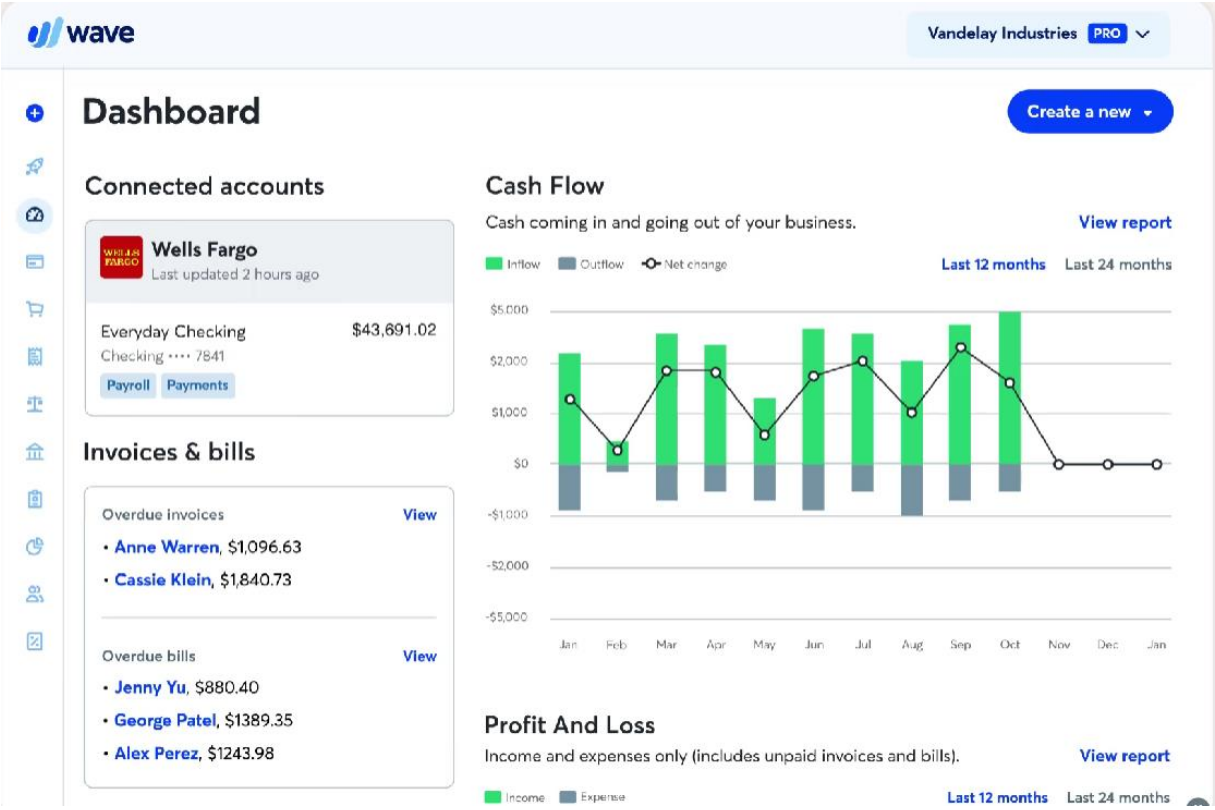


Рисунок 2.3 – Головна панель Wave Accounting з відображенням грошових потоків та інвойсів

Крім загального огляду фінансової ситуації, платформа надає зручний інтерфейс для створення рахунків-фактур. Користувач може вказати клієнта, вид послуги, кількість, податкові ставки та терміни оплати. Wave також дозволяє створювати повторювані інвойси або зберігати шаблони. Приклад форми створення рахунку наведено на рисунку 2.4.

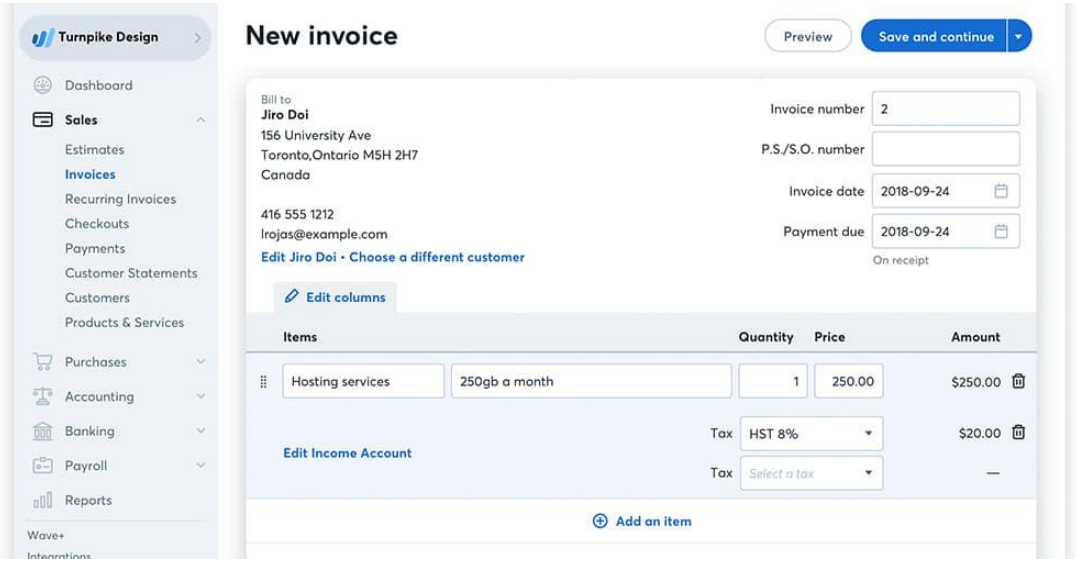


Рисунок 2.4 – Створення інвойсу у Wave Accounting: додавання клієнта, послуги та податків

Однак Wave має низку обмежень для українських користувачів. По-перше, платформа повністю англомовна і не має української локалізації. По-друге, її функціонал не адаптований до податкової системи України, і не підтримує вітчизняні банківські інтеграції. По-третє, сервіс працює лише в хмарі — усі дані зберігаються на зовнішньому сервері, що є неприйнятним для частини користувачів із міркувань безпеки. Технічна підтримка також орієнтована переважно на західні країни, а сам сервіс іноді обмежує доступ із певних регіонів.

Незважаючи на свою безкоштовність та простоту, Wave Accounting не може повною мірою відповідати вимогам українських користувачів через мовні та правові обмеження. Сервіс залишається актуальним лише для базового обліку в межах англомовного простору, але його практичне використання в Україні є малоймовірним без адаптації.

Таким чином, проведений аналіз дозволяє сформувати набір ключових вимог, які мають задовольняти сучасні програмні рішення для малого бізнесу. До них відносяться: простий та зручний інтерфейс, підтримка локального зберігання даних без обов'язкової синхронізації з хмарою, мінімальна кількість налаштувань, можливість офлайн-роботи, відсутність необхідності в додатковому ліцензуванні, а та-

кож прозора структура бази даних. Більшість існуючих систем лише частково відповідають цим критеріям, що створює потребу у створенні нових рішень, адаптованих до потреб малого бізнесу в Україні.

Висновки до розділу 2

Проведений аналіз трьох поширених програмних рішень — Finmap, Fredo та Wave — показав, що кожне з них має суттєві обмеження для представників малого бізнесу в Україні. Основними недоліками є надлишкова складність, платність, англомовність, залежність від Інтернету, а також збереження даних у хмарі. Жодна з систем не поєднує у собі простоти, локального зберігання, безкоштовності та фокусування саме на мікропідприємствах, які не мають технічних ресурсів для роботи з великим бухгалтерським ПЗ. Огляди користувачів на платформі Capterra також підтверджують, що більшість популярних рішень мають обмеження щодо мови, функціоналу або зручності використання для малого бізнесу [17].

Ці фактори обґрунтовують доцільність створення нової системи, розробленої в межах даної дипломної роботи. Вона передбачає збереження даних локально, використання інтуїтивно зрозумілого інтерфейсу, мінімальну кількість налаштувань і просту логіку обліку фінансових операцій, що робить її доступною для звичайного підприємця без додаткових знань у сфері програмування або бухгалтерії.

3 ФОРМУВАННЯ ВИМОГ ДО СИСТЕМИ

3.1 Вимоги до системи в цілому

У цьому підрозділі сформульовано загальні вимоги до структури та функціонування інформаційної системи обліку фінансових операцій для малого бізнесу, що розробляється в межах дипломного проєкту. Вимоги охоплюють організацію підсистем, режимів роботи, перспективи розвитку, а також аспекти надійності та збереження інформації у випадку критичних подій.

Вимоги до структури та функціонування системи

Структура створюваної системи повинна відповідати модульному принципу, що забезпечує гнучкість, масштабованість і простоту підтримки. Програмна система умовно поділяється на такі логічні підсистеми:

– підсистема введення даних — відповідає за інтерфейс користувача, який дозволяє швидко та зручно вносити фінансові операції (доходи, витрати, перекази), а також категорії, контрагентів і дати.

– підсистема обробки та перевірки даних — виконує базову перевірку коректності введених значень (наприклад, формат дати, числові значення суми, унікальність категорій), здійснює збереження інформації до бази даних.

– підсистема збереження даних — реалізована за допомогою вбудованої СУБД SQLite. Забезпечує централізоване зберігання всієї інформації у структурованому вигляді.

– аналітична підсистема — відповідає за формування фінансових звітів (за періодами, категоріями тощо), розрахунок підсумкових значень і побудову графічної візуалізації.

– підсистема виведення результатів — забезпечує вивід інформації у зручному форматі (таблиці, графіки), можливість перегляду результатів обліку без додаткових інструментів.

Кожна з підсистем повинна бути максимально ізольованою, що дозволяє оновлювати або змінювати окремі модулі без порушення функціонування всієї системи.

Система повинна функціонувати в локальному режимі, без потреби у підключенні до Інтернету. Це дає змогу використовувати її у будь-яких умовах — навіть за відсутності стабільного доступу до мережі. Водночас структура повинна передбачати можливість розширення функціоналу, наприклад, додавання експортних модулів, формування PDF-звітів або інтеграції з бухгалтерськими платформами в майбутньому.

Перспективи розвитку системи передбачають:

– впровадження ролей користувачів (адміністратор, касир тощо),

– автоматичне резервне копіювання бази даних,

– підтримку хмарної синхронізації за бажанням користувача,

– інтеграцію із зовнішніми джерелами даних (API банків, Excel, Google Sheets),

– багатомовність інтерфейсу.

Вимоги до надійності

Інформаційна система має бути стійкою до типових збоїв і працювати у штатному режимі при виникненні незначних технічних відмов. Надійність системи розглядається як здатність зберігати працездатність і точність обробки даних у разі таких подій:

– раптове завершення роботи програми;

– несподіване вимкнення електроживлення;

– пошкодження або тимчасова недоступність окремих файлів;

– некоректне завершення операції користувачем.

Для забезпечення надійності повинні бути реалізовані наступні заходи:

– автоматичне збереження даних після кожної транзакції до бази даних;

– обробка помилок вводу з відповідними повідомленнями;

– локальне кешування змін перед записом до файлу;

– перевірка цілісності даних при відкритті програми.

Крім того, система має виявляти та запобігати дублюванню записів, що може виникнути при повторному введенні даних після збою. У разі аварійного завершення сесії програма повинна мати можливість відновити незбережені зміни.

Програмне забезпечення повинне мати мінімальні вимоги до ресурсів та бути сумісним із основними версіями операційної системи Windows, а також, за можливості, Linux. Всі модулі повинні бути реалізовані без використання нестабільних або експериментальних бібліотек.

Вимоги до збереження інформації

Збереження даних є критично важливою вимогою до фінансових систем. Будь-яка втрата даних, особливо за підсумками тривалого періоду, може мати серйозні наслідки для бізнесу. У зв'язку з цим система повинна гарантувати надійне зберігання інформації навіть у випадках:

– аварійного завершення роботи програми;

– відмови в роботі жорсткого диска або флеш-носія;

– спроби несанкціонованого втручання або редагування бази вручну;

– втрати живлення або зависання ОС.

Для зменшення ризиків передбачається:

– локальне зберігання бази даних SQLite з обмеженням доступу (тільки через додаток);

– можливість ручного створення резервної копії бази даних користувачем;

– захист від дублювання транзакцій або порожніх записів через логіку перевірки;

– вивід підтверджень та повідомлень після збереження важливої інформації.

У перспективі може бути реалізовано автоматичне створення резервної копії при кожному запуску програми або після певної кількості змін. Також система повинна мати мінімальні зовнішні залежності, щоб її можна було перенести на інший комп'ютер без втрати функціоналу.

3.2 Вимоги до функціональних характеристик

Функціональні характеристики є однією з ключових складових, що визначають якість, придатність та ефективність інформаційної системи. Для розроблюва-

ного програмного забезпечення обліку фінансових операцій малого бізнесу важливо забезпечити не лише реалізацію основного функціоналу, але й гарантувати його доступність, точність, швидкість виконання та зручність для кінцевого користувача.

Перелік функцій та задач, що підлягають автоматизації

Інформаційна система має забезпечувати повноцінне виконання таких функціональних задач:

– додавання нових фінансових операцій (дохід, витрата, переказ між рахунками) з можливістю вказання дати, суми, категорії, контрагента, коментаря.

– редагування та видалення існуючих записів, у тому числі з перевіркою на повторне збереження.

– категоризація фінансових операцій — користувач повинен мати можливість створювати власні категорії витрат і доходів, а також змінювати їхні назви.

– ведення списку контрагентів — з метою полегшення повторного вибору при створенні нових транзакцій.

– перегляд історії всіх операцій, відфільтрованої за датою, типом, категорією чи контрагентом.

– формування фінансових звітів за вибраний період з підсумками доходів, витрат, залишку та балансу.

– побудова графіків динаміки фінансових потоків за категоріями, днями або місяцями.

– резервне копіювання бази даних (у перспективі).

– пошук записів за ключовими словами чи параметрами (у перспективі).

Весь функціонал повинен бути інтуїтивно зрозумілим і не вимагати від користувача спеціальної підготовки. Система має бути готовою до використання одразу після запуску, без потреби у тривалому налаштуванні.

Вимоги до реалізації функцій та якості виконання

Усі функції мають реалізовуватися з дотриманням таких принципів:

– швидкість виконання: час додавання або перегляду транзакції не повинен перевищувати 0,5–1 секунди.

– стабільність: навіть за багаторазового використання система не повинна зависати або викликати помилки.

– захист від помилок вводу: при введенні некоректних даних (наприклад, тексту замість чисел у полі "сума") система повинна повідомляти користувача і не зберігати некоректну транзакцію.

– автоматичне оновлення інформації в інтерфейсі після внесення змін — наприклад, при додаванні нової категорії або операції.

– логічна послідовність дій: користувач повинен чітко розуміти, які етапи слід виконати для створення або перегляду звіту.

Інформація про транзакції повинна відображатися у табличному вигляді, з колонками: дата, сума, тип (дохід/витрата), категорія, контрагент, коментар. Таблиці повинні мати можливість сортування та фільтрації.

Графічна візуалізація реалізується у вигляді стовпчикових діаграм, які наочно демонструють розподіл витрат і доходів по категоріях або динаміку руху коштів протягом періоду. Цей функціонал значно підвищує аналітичну цінність системи і дозволяє користувачу швидко виявити фінансові тенденції.

Взаємодія користувачів із системою проілюстрована на діаграмі варіантів використання, що наведена у кресленику ІС11.12БАК.005 Д1.

Всі вихідні результати повинні бути зрозумілими, стисло поданими, з можливістю копіювання, збереження або екранного знімка для подальшого використання. У перспективі можливим буде реалізація експорту у форматах CSV, PDF або XLSX, однак на базовому рівні головна вимога — коректне виведення даних у вікні застосунку.

3.3 Вимоги до видів забезпечення

Функціонування сучасної інформаційної системи неможливе без чіткого визначення вимог до всіх видів забезпечення, що беруть участь у процесах обробки, зберігання, аналізу та представлення даних. У цьому підрозділі наведено вимоги до

математичного, інформаційного, програмного та технічного забезпечення розроблюваної системи обліку фінансових операцій для малого бізнесу.

3.3.1 Вимоги до математичного забезпечення

Математичне забезпечення системи визначає сукупність математичних моделей, методів та алгоритмів, необхідних для обробки фінансових даних, формування звітів, аналітики та візуалізації.

У рамках розробки системи передбачено використання таких основних методів і моделей:

– базова модель бухгалтерського обліку: у формі спрощеної подвійної записи (дохід/витрата/переказ).

– методи накопичення та групування даних: розрахунок загальної суми операцій за категоріями, періодами, рахунками.

– алгоритми агрегації: щомісячна, щоденна або щотижнева агрегація операцій із метою побудови динамічних графіків.

– статистичні обчислення: обчислення середнього значення, мінімуму, максимуму, залишку на рахунках.

– методи візуалізації: побудова стовпчикових діаграм на основі агрегованих значень із використанням бібліотек matplotlib.

Усі обчислення реалізуються через математичні функції, які вмонтовані в середовище розробки (Python).

3.3.2 Вимоги до інформаційного забезпечення

Інформаційне забезпечення є основою для зберігання, структурування та доступу до фінансових даних. У системі реалізовано використання локальної реляційної бази даних SQLite, що дозволяє забезпечити простоту, надійність та автономність роботи без зовнішніх залежностей.

Інформаційна модель включає такі основні сутності:

– таблиця користувачів (у перспективі, при розширенні функціоналу);

– таблиця транзакцій — містить поля: id, дата, тип (дохід, витрата, переказ), сума, категорія, контрагент, коментар;

– таблиця категорій — назви категорій доходів та витрат, які можуть бути обрані під час створення транзакції;

– таблиця контрагентів — зберігає імена/назви контрагентів;

– таблиця рахунків — дозволяє відстежувати рух коштів між різними джерелами (готівка, банківський рахунок тощо).

Інформаційна структура повинна:

– забезпечувати унікальність записів;

– дозволяти швидкий пошук і фільтрацію;

– підтримувати збереження в локальному форматі .db;

– бути придатною для резервного копіювання;

– мати логічну ієрархію для масштабування.

Резервне копіювання даних реалізується як можливість копіювати файл бази вручну. У майбутньому може бути реалізована функція автоматичного створення резервної копії при запуску системи або після кожних 10 змін.

3.3.3 Вимоги до програмного забезпечення

Програмне забезпечення системи має бути реалізоване із застосуванням вільних, стабільних і кросплатформних технологій, які не потребують додаткових витрат користувача на ліцензування. Основні інструменти та компоненти:

– мова програмування: Python (версія 3.9+), як одна з найбільш доступних та читабельних мов із широкою екосистемою.

– бібліотеки:

tkinter — для створення графічного інтерфейсу користувача (GUI);

sqlite3 — для взаємодії з базою даних SQLite;

matplotlib — для побудови графіків та візуалізації;

datetime, os, csv — для допоміжних операцій.

Використання Python гарантує кросплатформність, можливість запуску програми на Windows, Linux і macOS без змін у коді.

Якість програмного забезпечення забезпечується:

– перевіркою коректності введених даних;

– наявністю обробки помилок та сповіщень;

– стабільністю при багаторазовому використанні;

– можливістю легкої адаптації або розширення в майбутньому (відкритий код, модульна структура).

Програма не вимагає встановлення сторонніх середовищ або серверів, працює у вигляді .py-файлу або зібраного .exe.

3.3.4 Вимоги до технічного забезпечення

Для повноцінного функціонування системи не потрібні спеціалізовані апаратні ресурси. Це дозволяє використовувати її навіть на застарілих або малопотужних пристроях.

Мінімальні технічні вимоги:

– процесор: Intel Core i3 або аналог;

– оперативна пам’ять: від 2 ГБ;

– вільне місце на диску: від 200 МБ;

– операційна система: Windows 7/10/11, Linux (Ubuntu, Mint тощо);

– роздільна здатність екрану: від 1366x768.

Додаткові вимоги:

– можливість встановлення Python або запуску .exe;

– бажано — наявність миші та клавіатури для зручної взаємодії з інтерфейсом;

– підтримка локального зберігання файлів (без хмарних сервісів).

Жодні спеціальні сервери, інтернет-з’єднання або ліцензійні ключі не потрібні. Це забезпечує доступність системи навіть у складних умовах, наприклад, у по-

льових офісах або за нестабільного зв’язку. Такий підхід робить систему універсальною та придатною для використання широким колом користувачів, включаючи представників малого бізнесу, фрілансерів та громадські організації. Простота встановлення та відсутність залежності від зовнішніх ресурсів мінімізує витрати на впровадження і технічну підтримку.

Висновки до розділу 3

У цьому розділі було визначено та систематизовано основні вимоги до інформаційної системи обліку фінансових операцій для малого бізнесу. Детально проаналізовано вимоги до структури та функціонування системи, її надійності, збереження інформації, а також до функціональних характеристик та всіх видів забезпечення — математичного, інформаційного, програмного та технічного.

Сформульовані вимоги дозволяють забезпечити ефективну, стабільну та безпечну роботу системи в умовах обмежених ресурсів, характерних для малого бізнесу. Запропонована модульна структура, локальне зберігання даних, інтуїтивно зрозумілий інтерфейс і мінімальні технічні вимоги гарантують доступність продукту для широкого кола користувачів без необхідності в спеціалізованому обладнанні або знаннях.

Таким чином, сформовані вимоги стали фундаментом для подальшої реалізації інформаційної системи, забезпечуючи баланс між простотою використання та функціональністю, що відповідає сучасним потребам підприємців та фахівців мікро- і малого бізнесу.

4 ВИБІР ТЕХНОЛОГІЇ РОЗРОБЛЕННЯ

4.1 Вибір стеку технологій

З огляду на нефункціональні вимоги, сформульовані у попередньому розділі, доцільним рішенням для реалізації системи обліку фінансових операцій малого бізнесу є створення десктопного застосунку з можливістю подальшого розширення у вебформат. Основними причинами такого вибору є:

– простота впровадження. Система працює локально, не вимагає постійного доступу до мережі Інтернет, що є критично важливим для багатьох підприємств з обмеженими технічними ресурсами.

– відсутність необхідності встановлення серверної інфраструктури. База даних SQLite функціонує як звичайний файл, що значно спрощує розгортання та обслуговування програми.

– гнучкість у розробці. Обрана архітектура дозволяє в подальшому реалізувати інтеграцію з вебінтерфейсом або мобільним додатком без значних змін у бізнес-логіці.

Нижче буде наведено ключові технології, які застосовувались під час реалізації даної системи:

– Python як основна мова програмування

– SQLite для зберігання даних

– Matplotlib як бібліотека для побудови візуалізацій

– Gettext для реалізації мультимовного інтерфейсу

– n8n для організації потоків обробки даних і інтеграції з зовнішніми сервісами

– Telegram API для реалізації чат-бота, який виступає фінансовим консультантом

– Gemini як мовна модель для аналізу фінансової інформації та надання консультацій

– Tkinter (або інший GUI-фреймворк) для створення користувацького інтерфейсу десктопного застосунку

– JSON для обміну даними між модулями та зовнішніми системами

4.2 Python

Python — високорівнева мова програмування, що поєднує у собі простоту синтаксису, широкі можливості бібліотек і гнучкість використання [1]. Основними характеристиками, що обумовили вибір цієї мови, є:

– мінімальний поріг входу. Завдяки простому синтаксису мова ідеально підходить для швидкої розробки прототипів;

– величезна кількість бібліотек. Python має сотні готових рішень, що дозволяє суттєво зекономити час на розробку допоміжного функціоналу;

– універсальність. Python використовується у веброзробці, аналітиці, машинному навчанні, автоматизації тощо;

– безкоштовність та відкритість. Мова є вільною для комерційного використання, що знижує витрати бізнесу.

4.3 SQLite

SQLite — це легка вбудована система керування реляційними базами даних, яка зберігає всю базу у вигляді одного файлу. Вона не потребує окремого серверного процесу, що робить її дуже простою у використанні і розгортанні [2].

Основні переваги SQLite для проєкту:

– простота впровадження. Не потребує налаштувань серверної інфраструктури, що економить час та ресурси на адміністрування.

– відсутність додаткових залежностей. Вбудований модуль sqlite3 у Python дозволяє безпечно та ефективно працювати з базою даних без необхідності встановлювати сторонні пакети. Додаткову перевагу становить велика кількість готових прикладів і рішень, доступних у відкритому доступі — зокрема на технічних форумах, таких як Stack Overflow [15].

– надійність і ефективність. SQLite є надійною СУБД, що забезпечує транзакції та цілісність даних навіть у випадках аварійного завершення роботи.

– легкість резервного копіювання. Оскільки база зберігається у вигляді одного файлу, резервне копіювання і відновлення здійснюється простою копією файлу.

У системі обліку фінансів SQLite відповідає за збереження інформації про операції, категорії витрат і користувачів. Вона підтримує зручне фільтрування та пошук даних, що забезпечує швидкий доступ до необхідної інформації [8, 10]. Завдяки своїй стабільності та активній підтримці спільноти, SQLite широко використовується у численних комерційних і відкритих програмних продуктах. Її використання у даному проєкті гарантує масштабованість рішення, а також спрощує подальше обслуговування системи без залучення додаткових технічних ресурсів.

4.4 Matplotlib

Matplotlib — це бібліотека для побудови статичних, анімованих і інтерактивних графіків у мові Python. Вона є однією з найпопулярніших і найпотужніших бібліотек візуалізації даних.

Чому Matplotlib використовується в проєкті:

– різноманітність типів графіків. Бібліотека підтримує побудову лінійних графіків, гістограм, кругових діаграм, які допомагають візуалізувати структуру витрат, доходів, тенденції за часом.

– гнучкість і налаштування. Можна змінювати кольори, стилі ліній, шрифти, додавати підписи та легенди, що робить звіти більш інформативними та зручними для користувача.

– інтеграція з іншими бібліотеками Python. Matplotlib легко працює разом із бібліотеками для аналізу даних (наприклад, pandas), що дає змогу створювати комплексні аналітичні інструменти.

У системі обліку фінансових операцій Matplotlib використовується для побудови графіків, які демонструють розподіл витрат за категоріями, динаміку змін

доходів і витрат за обраний період. Це допомагає бізнесу більш усвідомлено приймати фінансові рішення.

4.5 Gettext

Gettext — це система локалізації, яка дозволяє розробникам створювати багатомовні інтерфейси без необхідності дублювати код. Вона працює через зовнішні файли перекладів (.po та .mo), що забезпечує гнучкість у підтримці різних мов.

Переваги використання Gettext у проєкті:

– відокремлення текстів інтерфейсу від логіки програми. Це робить код чистішим і дозволяє легко залучати перекладачів без знань програмування.

– підтримка необмеженої кількості мов. Легко додати підтримку нових мов у майбутньому.

– поширеність та сумісність. Gettext широко використовується у відкритому програмному забезпеченні, що забезпечує стабільність і підтримку.

У системі обліку фінансів реалізовано перемикання між українською та російською мовами, що робить продукт доступним для ширшої аудиторії та покращує користувацький досвід.

4.6 n8n

n8n — це платформа для автоматизації робочих процесів (workflow automation), яка дозволяє організувати послідовну обробку даних та інтегрувати систему з різними зовнішніми сервісами без написання великої кількості коду.

Чому n8n використовується в проєкті:

– гнучкість у створенні сценаріїв обробки інформації, що дозволяє налаштовувати логіку роботи чат-бота та інтеграції з API;

– можливість підключення до різних джерел даних і сервісів (наприклад, Telegram, зовнішні бази, вебсервіси);

– візуальний редактор потоків, що полегшує підтримку та розширення функціоналу без глибоких знань програмування.

У системі n8n використовується для організації взаємодії між чат-ботом, мовною моделлю Gemini та базою даних, що забезпечує оперативну та точкову консультацію користувачам.

4.7 Telegram API

Telegram API — набір інструментів для створення ботів і інтеграції з месенджером Telegram, який дозволяє реалізувати спілкування користувача з системою через зручний інтерфейс.

Чому Telegram API використовується в проєкті:

– популярність і зручність месенджера серед малого бізнесу;

– можливість швидко отримувати відповіді на фінансові питання в реальному часі;

– підтримка передачі текстових повідомлень, таблиць, зображень і документів.

У проєкті Telegram API застосовується для реалізації чат-бота, що виступає консультантом, який цілодобово допомагає користувачам розбиратися з фінансовими звітами та питаннями.

4.8 Gemini

Gemini — сучасна мовна модель, розроблена для аналізу та генерації текстової інформації на основі великих обсягів даних.

Чому Gemini використовується в проєкті:

– здатність розуміти контекст фінансових даних і формувати чіткі, конкретні відповіді;

– можливість задавати уточнюючі питання для отримання додаткової інформації;

– підтримка української мови з можливістю перемикання на англійську при необхідності.

У системі Gemini використовується для надання консультацій на основі даних, які надходять від користувача, підвищуючи якість аналітики та підтримки.

4.9 Логування

Для забезпечення безпеки та аудиту в системі реалізовано механізм логування спроб доступу користувачів. У файл access_requests.log записуються:

– дата і час спроби входу;

– електронна пошта користувача.

– значення логування у проєкті:

– контроль безпеки. Допомагає виявляти несанкціоновані спроби доступу;

– аналіз активності. Дозволяє відслідковувати поведінку користувачів для подальшого покращення системи;

– документування. Записи можуть бути використані при аудиті чи розслідуванні інцидентів.

Висновки до розділу 4

Обраний набір технологій повністю відповідає вимогам малого бізнесу — простота, економічність, швидкість розробки. Рішення дозволяє реалізувати систему, яка забезпечує базовий фінансовий контроль, підтримує зручну фільтрацію, категоризацію, а також візуалізацію витрат. У разі потреби, її легко можна масштабувати під нові бізнес-процеси.

5 РОЗРОБЛЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

5.1 Середовища розробки

Для створення інформаційної системи обліку фінансових операцій був обраний сучасний і популярний стек технологій, що забезпечує гнучкість розробки, високу продуктивність та зручність подальшого супроводу системи.

Основою проєкту стала мова програмування Python (версія 3.x), яка широко використовується у розробці як простих, так і складних десктопних застосунків. Python відомий своєю читабельністю коду, великою кількістю бібліотек і активним ком'юніті. Це дозволило розробити функціональний додаток із зручним інтерфейсом у короткі терміни.

Середовище розробки

Для написання та редагування вихідного коду було використано середовище розробки Visual Studio Code (VS Code), яке на сьогодні є одним із найбільш популярних редакторів коду. VS Code поєднує в собі легкість текстового редактора і потужність IDE, пропонуючи розширений функціонал, необхідний для комфортної роботи з Python-проєктами.

Основні переваги використання VS Code:

Підтримка багатьох мов програмування та розширень, що дають змогу адаптувати середовище під конкретні потреби розробника;

Інтелектуальне автодоповнення коду (IntelliSense), яке значно прискорює процес написання програм і знижує кількість помилок;

Вбудований відлагоджувач, що дозволяє швидко знаходити та виправляти помилки;

Інтеграція з системами контролю версій, такими як Git, що полегшує керування кодом і командну роботу [7, 14];

Підтримка віртуальних середовищ Python, що допомагає ізолювати залежності проєкту.

Додатково, VS Code має інтуїтивно зрозумілий інтерфейс, що прискорює освоєння та роботу навіть початківцям [6].

Використані бібліотеки

Для реалізації різних аспектів системи було використано декілька спеціалізованих бібліотек Python, які дозволили ефективно розв'язати задачі роботи з графічним інтерфейсом, базою даних та візуалізацією:

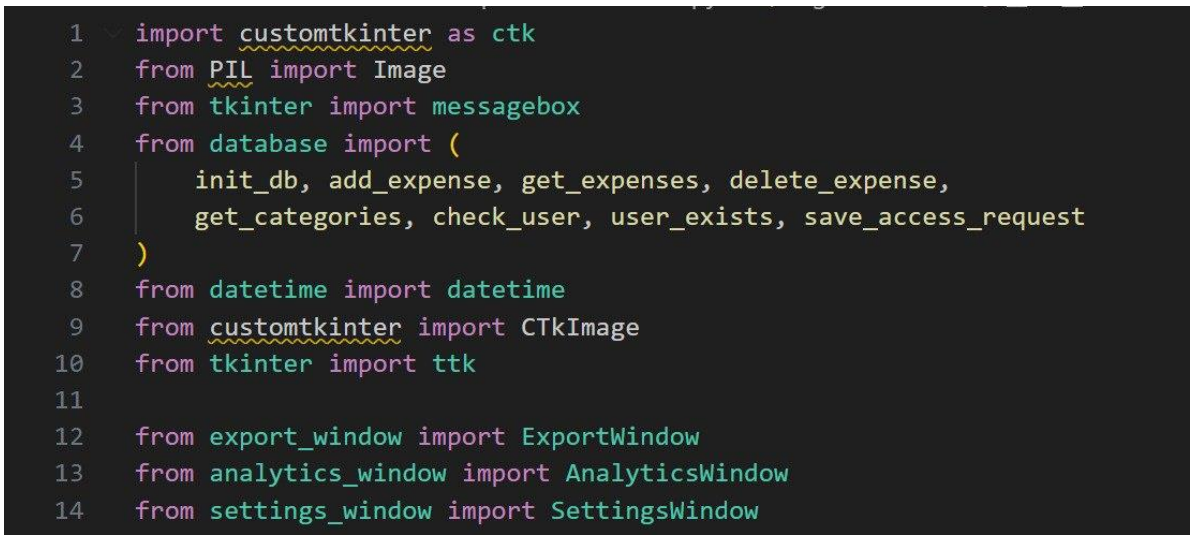
CustomTkinter — розширення для стандартної бібліотеки tkinter, яке надає сучасний і привабливий інтерфейс із підтримкою темної та світлої тем, сучасних віджетів і стилізації. Використання цієї бібліотеки дозволило покращити юзабіліті системи та зробити інтерфейс більш сучасним і комфортним для користувача [5].

sqlite3 — стандартна бібліотека Python для роботи з базою даних SQLite. Вона забезпечує простий і водночас потужний спосіб зберігання структурованих даних локально без потреби налаштування складних серверних СУБД. SQLite ідеально підходить для невеликих і середніх за обсягом даних проєктів, таких як система обліку бюджету.

Pillow (PIL) — бібліотека для роботи з графікою, яка дозволяє завантажувати, обробляти та масштабувати зображення, зокрема іконки для інтерфейсу. Вона забезпечує підтримку популярних форматів зображень і оптимізує їх відображення у вікнах програми.

tkinter.ttk — розширення для стандартного модуля tkinter, що додає покращені віджети з сучасним оформленням. За допомогою ttk реалізовано таблиці з можливістю прокрутки, що використовуються для відображення даних у структурованому вигляді. Це підвищує зручність перегляду великих обсягів інформації [4].

matplotlib — одна з найпопулярніших бібліотек для побудови графіків [3] і візуалізації статистичних даних. Вона дозволяє створювати різноманітні типи діаграм (лінійні, стовпчикові, кругові), які інтегруються у графічний інтерфейс через спеціальні віджети. Використання matplotlib дозволяє користувачам швидко аналізувати динаміку доходів і витрат, а також оцінювати розподіл по категоріях. Приклад структури імпорту всіх необхідних бібліотек та модулів наведено на рисунку 5.1.



```python
import customtkinter as ctk
from PIL import Image
from tkinter import messagebox
from database import (
    init_db, add_expense, get_expenses, delete_expense,
    get_categories, check_user, user_exists, save_access_request
)
from datetime import datetime
from customtkinter import CTkImage
from tkinter import ttk

from export_window import ExportWindow
from analytics_window import AnalyticsWindow
from settings_window import SettingsWindow
```

Рисунок 5.1 – Імпорт стандартних бібліотек, сторонніх бібліотек та локальних модулів

Пояснення вибору технологій

Комбінація цих інструментів та бібліотек забезпечує оптимальний баланс між простотою реалізації, функціональністю та продуктивністю. Вибір Python зумовлений його популярністю у сфері розробки, наявністю великої кількості готових рішень і простотою вивчення, що особливо важливо для підтримки та масштабування проєкту в майбутньому. Крім того, Python підтримує мультиплатформеність, що дозволяє запускати програму як на Windows, так і на Linux або macOS без істотних змін у коді. Це розширює потенційне коло користувачів і полегшує розгортання в різних середовищах.

CustomTkinter дозволяє зробити графічний інтерфейс сучасним і приємним для користувача, що підвищує загальний користувацький досвід, а sqlite3 гарантує швидкий і надійний доступ до локальних даних без зайвих складнощів. Бібліотека sqlite3 гарантує швидкий і надійний доступ до локальних даних без зайвих складнощів. Вона ідеально підходить для невеликих бізнес-систем, де не потрібна складна серверна інфраструктура. База зберігається в одному файлі, що спрощує резервне копіювання та перенос даних. Зовнішній вигляд середовища розробки з відкритим кодом проєкту показано на рисунку 5.2.

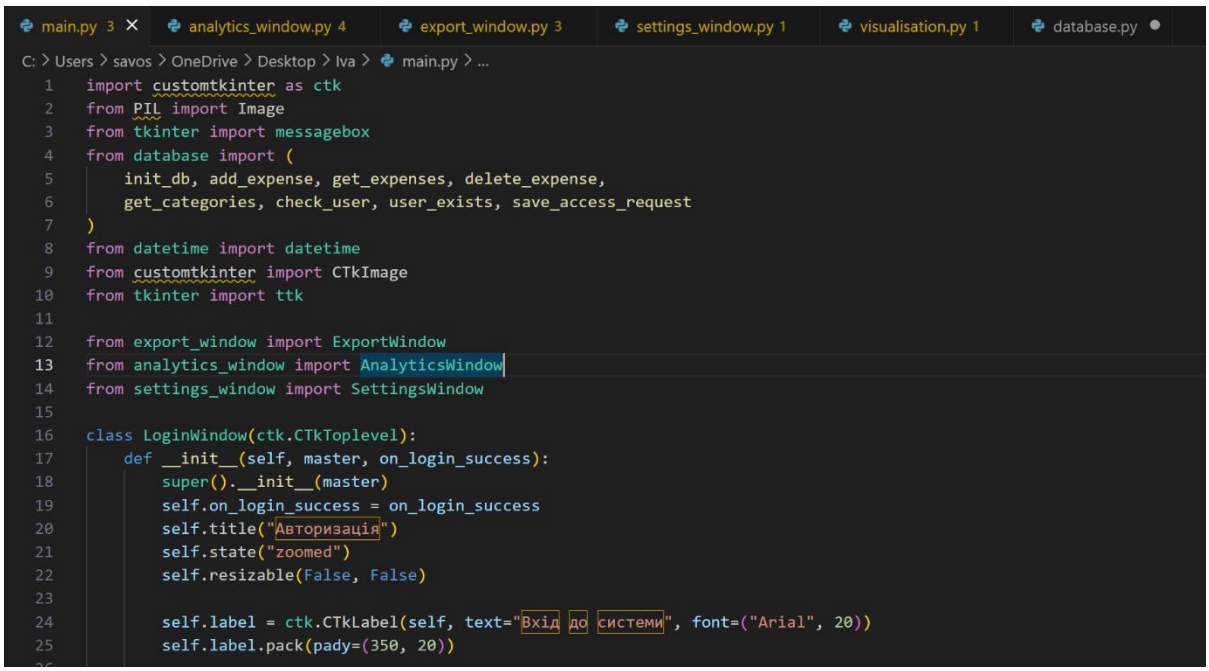


Рисунок 5.2 – VS Code, середовище з кодом проєкту

5.2 Python-застосунок

Застосунок має графічний інтерфейс, побудований на основі бібліотеки CustomTkinter. Основна мета — надати користувачу інтуїтивно зрозумілий інтерфейс для обліку та контролю фінансових операцій. Інтерфейс розроблений таким чином, щоб забезпечити простоту використання навіть для користувачів без спеціальної технічної підготовки. Зовнішній вигляд додатку виконаний у сучасному стилі з використанням кольорових тем, адаптованих шрифтів та іконок. Після запуску програми користувача зустрічає екран авторизації.

Якщо облікові дані введені правильно, система відкриває головне вікно, де доступні такі можливості:

– додавання нових витрат або доходів;

– вибір або створення категорій;

– перегляд та сортування записів за таблицею;

– використання бічної панелі для доступу до налаштувань, аналітики та експорту

Інтерфейс побудований у стилі «dashboard», з логічно згрупованими елементами управління. Введення суми, категорії та дати відбувається у верхній частині вікна, а нижче розміщено інтерактивну таблицю з усіма записами. Бічна панель за замовчуванням прихована, але відкривається кнопкою зі стрілкою для переходу до додаткових функцій. Головне вікно застосунку з основними елементами інтерфейсу зображено на рисунку 5.3.

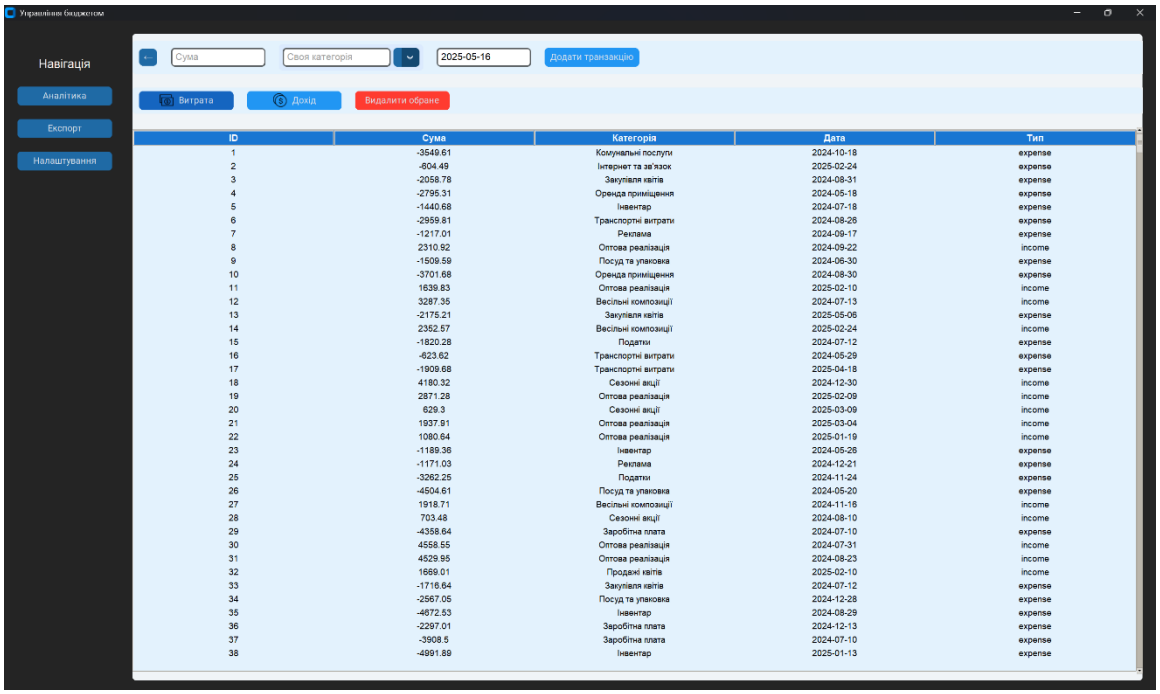


Рисунок 5.3 – Dashboard, головна сторінка

Однією з важливих функцій застосунку є вікно налаштувань, що дозволяє користувачу адаптувати інтерфейс під власні потреби. Вікно реалізоване як окреме модальне вікно, що відкривається поверх основного інтерфейсу. Це реалізовано за допомогою компонента CTkToplevel, який не блокує головний процес, але тимчасово привертає увагу користувача виключно до параметрів конфігурації.

У налаштуваннях реалізовано два основні пункти:

– вибір мови інтерфейсу — доступні варіанти «Українська» та «English». Поточна реалізація підтримує вибір зі списку, а в майбутньому можна передбачити зміну мовних ресурсів у реальному часі. Приклад вікна вибору мови показано на

рисунку 5.4. Мова інтерфейсу змінює усі підписи, кнопки та повідомлення в додатку відповідно до обраного варіанту. Це дозволяє користувачам з різною мовною підготовкою комфортно працювати з програмою без потреби в додатковому перекладі або інструкціях. Реалізація мультимовності також важлива з точки зору подальшого масштабування програмного продукту на міжнародні ринки.

– вибір валюти — користувач може встановити символ валюти, що буде використовуватись у відображенні сум. Підтримуються гривня (₴), долар ($) та євро (€). Це дозволяє адаптувати відображення даних до валютного середовища конкретного користувача чи компанії. Обрана валюта автоматично застосовується до всіх фінансових операцій, звітів і графіків у системі, що запобігає плутанині та помилкам у трактуванні сум. У майбутньому планується додати підтримку кількох валют одночасно для компаній, які ведуть облік у різних країнах. Вигляд вікна для вибору валюти наведено на рисунку 5.5.

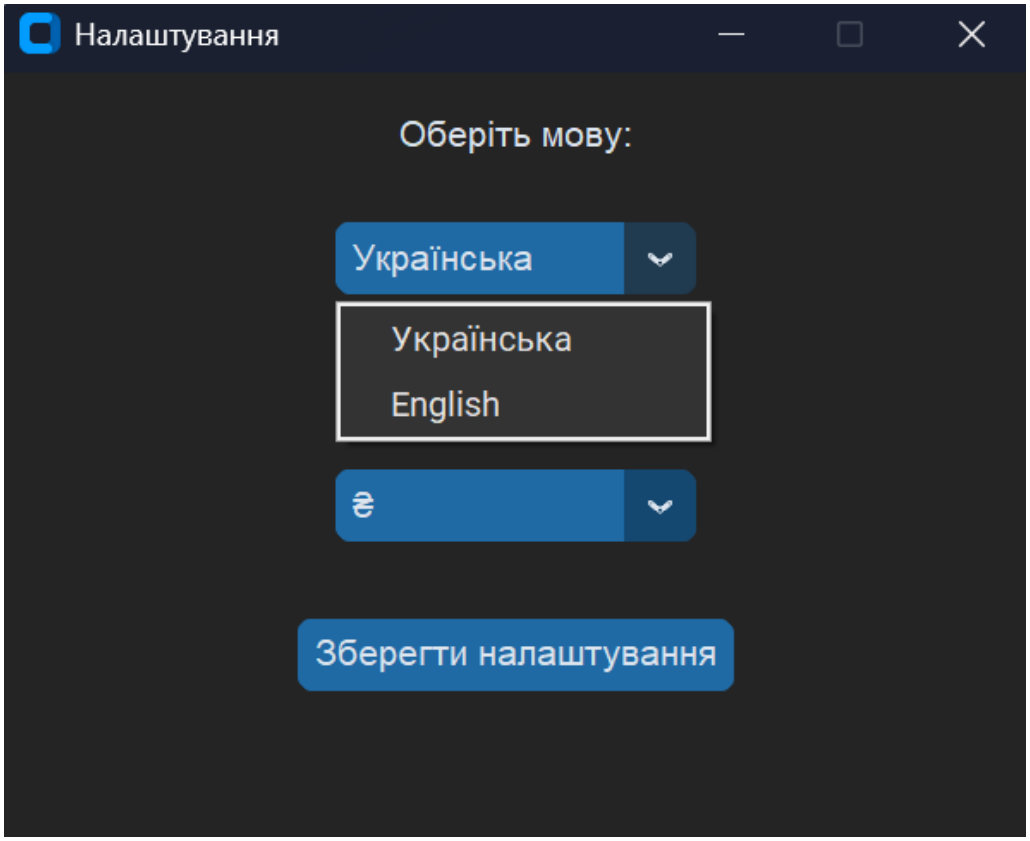


Рисунок 5.4 – Налаштування мови інтерфейсу

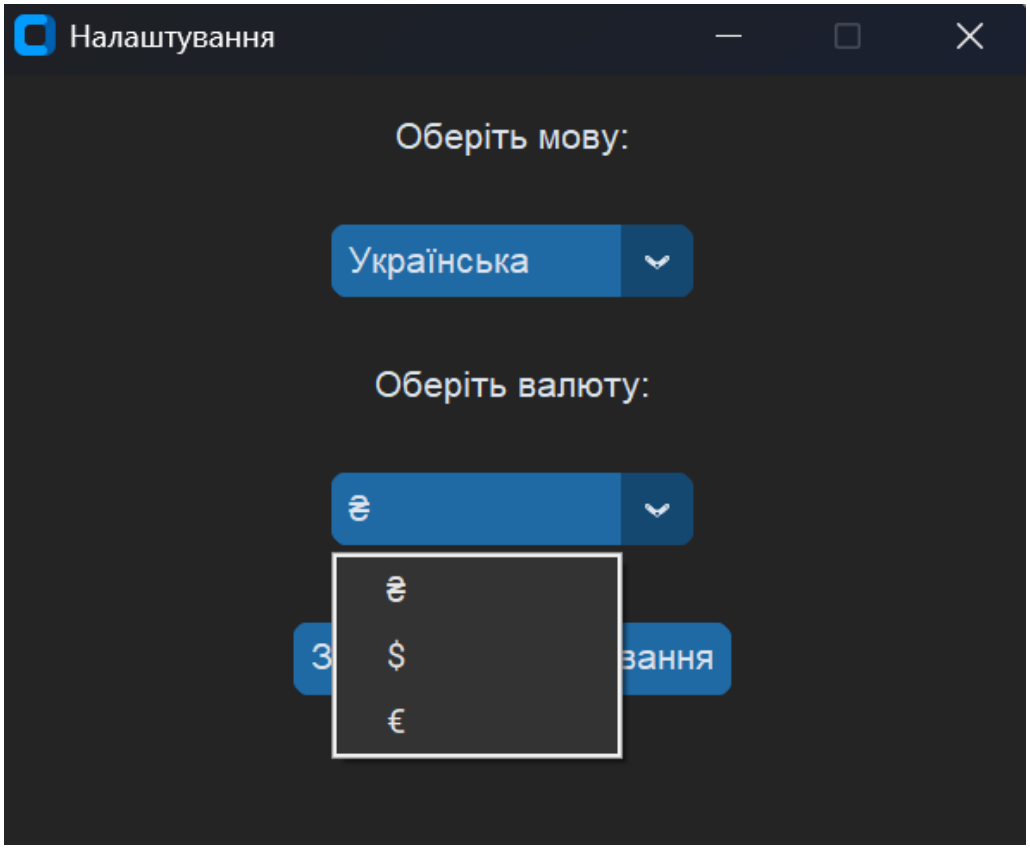


Рисунок 5.5 – Налаштування валюти інтерфейсу

Після вибору параметрів користувач натискає кнопку «Зберегти налаштування», після чого з'являється повідомлення з підтвердженням, реалізоване через messagebox. У базовій реалізації налаштування не зберігаються між сесіями, проте ця функціональність легко доповнюється записом параметрів у файл конфігурації або таблицю бази даних.

Ще однією важливою складовою інтерфейсу є вікно аналітики, яке дозволяє візуалізувати фінансові дані у вигляді графіків і зведень. Реалізоване як окреме модальне вікно (CTkToplevel), воно автоматично фільтрує та відображає записи з бази залежно від обраних параметрів.

У верхній частині вікна розташовано панель фільтрів: користувач може обрати тип транзакцій (усі, витрати, доходи) та часовий період (увесь час, останні 30 або 7 днів).

Основна частина інтерфейсу поділена на вкладки:

– динаміка — графік змін доходів і витрат у часі;

– категорії — кругова діаграма розподілу витрат;

– зведення — підсумки: дохід, витрати, баланс;

– топ витрат — 5 категорій із найбільшими витратами;

– баланс — графік кумулятивного залишку коштів;

– дні тижня — аналіз активності по днях.

Для побудови графіків використано бібліотеки matplotlib та pandas. Це дозволяє гнучко обробляти дані з бази та наочно їх виводити. Якщо для вибраного фільтру даних немає — виводиться відповідне повідомлення.

Аналітика допомагає користувачу оперативно оцінити фінансову ситуацію, знайти найвитратніші категорії або проаналізувати поведінку доходів та витрат за тиждень. Результат наведено на рисунку 5.6.

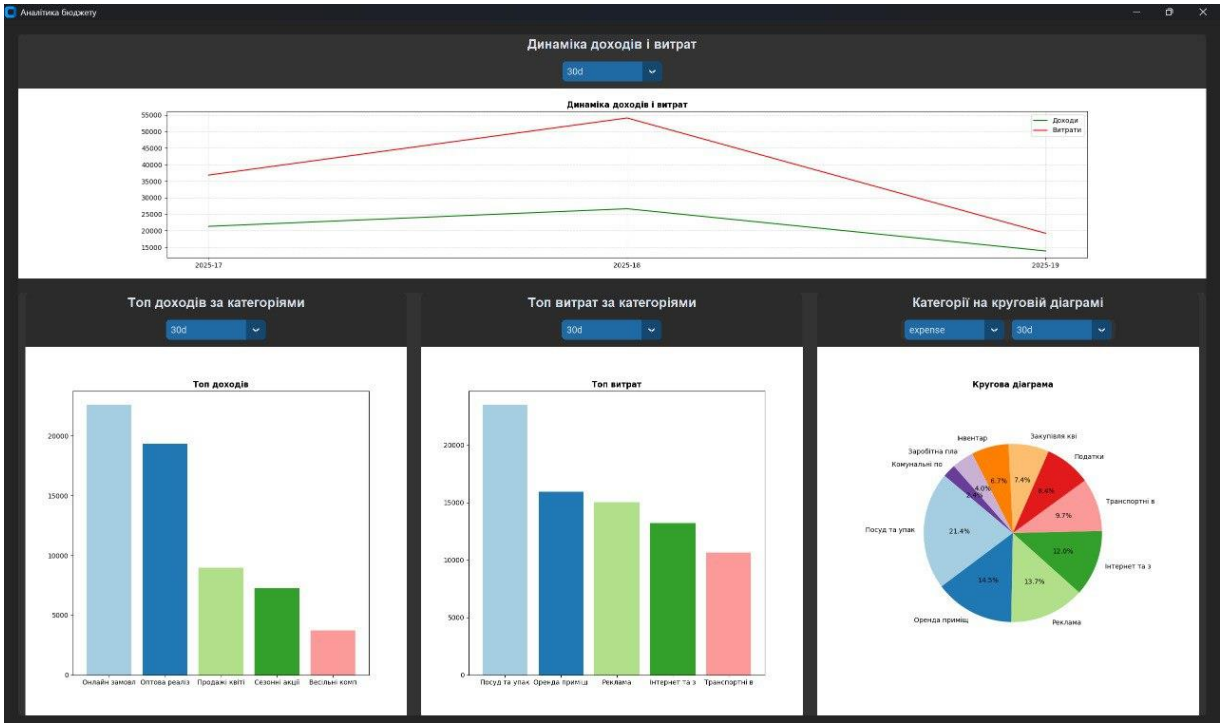


Рисунок 5.6 – Розділ аналітики

На даному екрані користувач бачить комплексний огляд фінансової діяльності: графік динаміки доходів і витрат за увесь час, топ категорій доходів і витрат за сумами, а також кругову діаграму розподілу витрат. Це дозволяє оперативно оцінити загальну ситуацію в бізнесі та виявити ключові напрями доходів і витрат.

Рисунок 5.7 – Динаміка доходів та витрат

На рисунку 5.7 наведено графік, який демонструє щоденну зміну обсягів доходів та витрат. Зелена лінія відповідає доходам, червона – витратам. Завдяки щільному часовому ряду можна відслідкувати коливання грошових потоків, виявити пікові значення та проаналізувати можливі причини фінансових спадів або зростань. Такий графік є особливо корисним для виявлення сезонних трендів або повторюваних витрат, що дозволяє користувачу краще планувати бюджет. Наприклад, якщо червона лінія стабільно зростає в окремі періоди (наприклад, на початку кожного місяця), це може свідчити про регулярні витрати на зарплати чи оренду. У той же час падіння зеленої лінії у вихідні дні може вказувати на зниження доходів у неробочі періоди. Також на основі цього візуального аналізу можна своєчасно реагувати на небажані зміни у фінансовій поведінці. Інтерфейс дозволяє налаштувати часовий інтервал, що підвищує гнучкість у роботі з даними. Користувач може обрати, наприклад, лише один місяць або квартал, щоб побачити тенденції в межах короткого періоду, або переглянути річну динаміку для глобального аналізу. Це дозволяє враховувати як короткострокові, так і довгострокові фінансові зміни. Наявність візуального відображення фінансових потоків спрощує прийняття рішень для користувача без необхідності поглибленого аналізу числових таблиць. Застосування лінійних графіків у поєднанні з кольоровим кодуванням сприяє швидкому розумінню поточної фінансової ситуації. Крім того, графік є динамічним — після зміни вхідних даних або додавання нових транзакцій, зображення автоматично оновлюється, що робить аналітику актуальною в режимі реального часу. Це особливо важливо для керівників або бухгалтерів, які приймають рішення щодня.

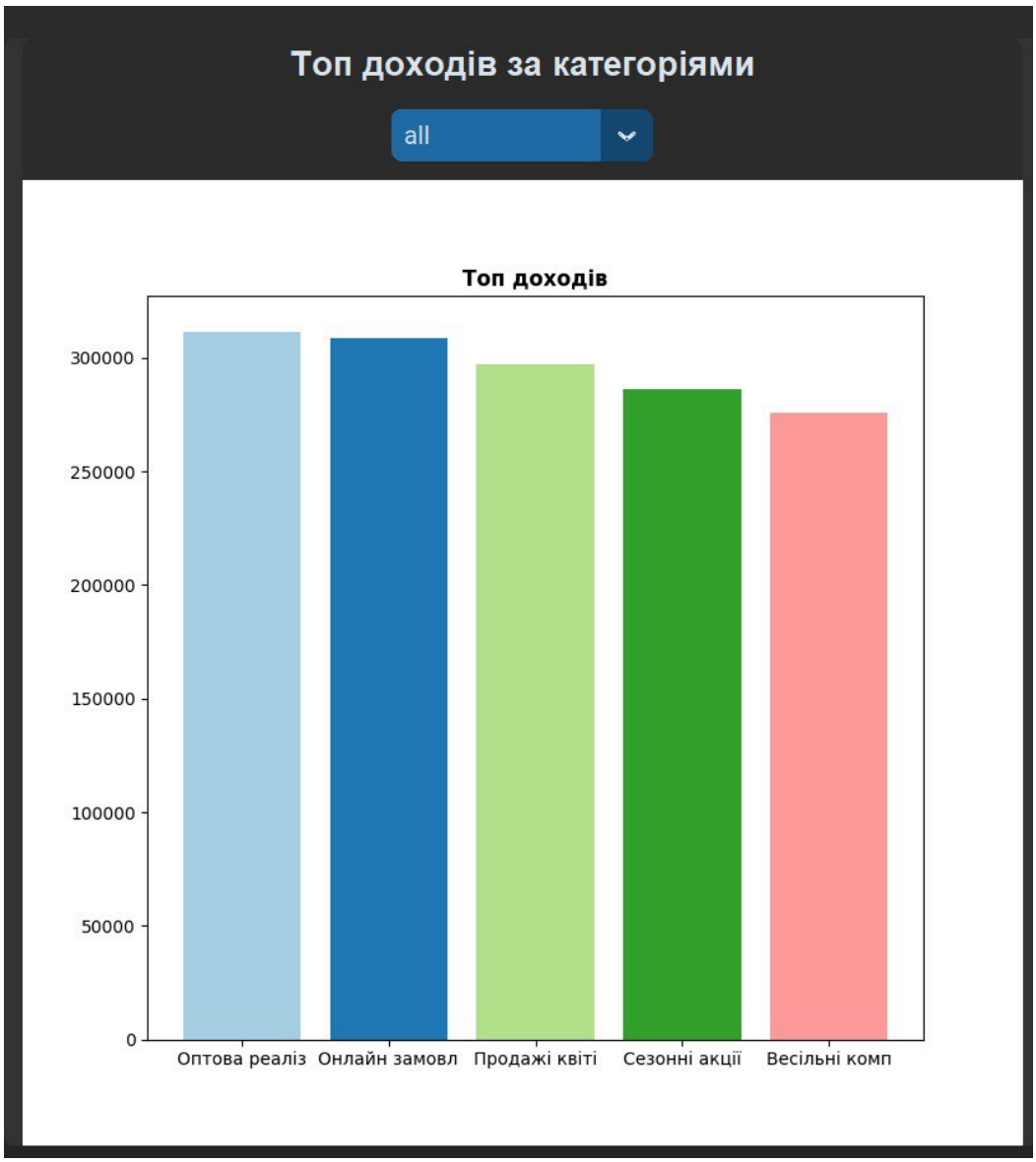


Рисунок 5.8 – Топ доходів за категоріями

На рисунку 5.8 зображено графік, де представлено п’ять основних категорій, що формують дохід підприємства. Серед них – оптова реалізація, онлайн замовлення, продажі квітів, сезонні акції та весільні композиції. Даний звіт дозволяє визначити ключові напрями прибутковості бізнесу та потенційні точки зростання. Стовпчаста діаграма дозволяє візуально порівняти обсяги доходів за різними напрямами та виявити, яка з категорій є найбільш прибутковою. Наприклад, оптова реалізація та онлайн замовлення показують найвищі результати, що може свідчити про ефективність каналів дистрибуції або маркетингових стратегій у цих сегментах.

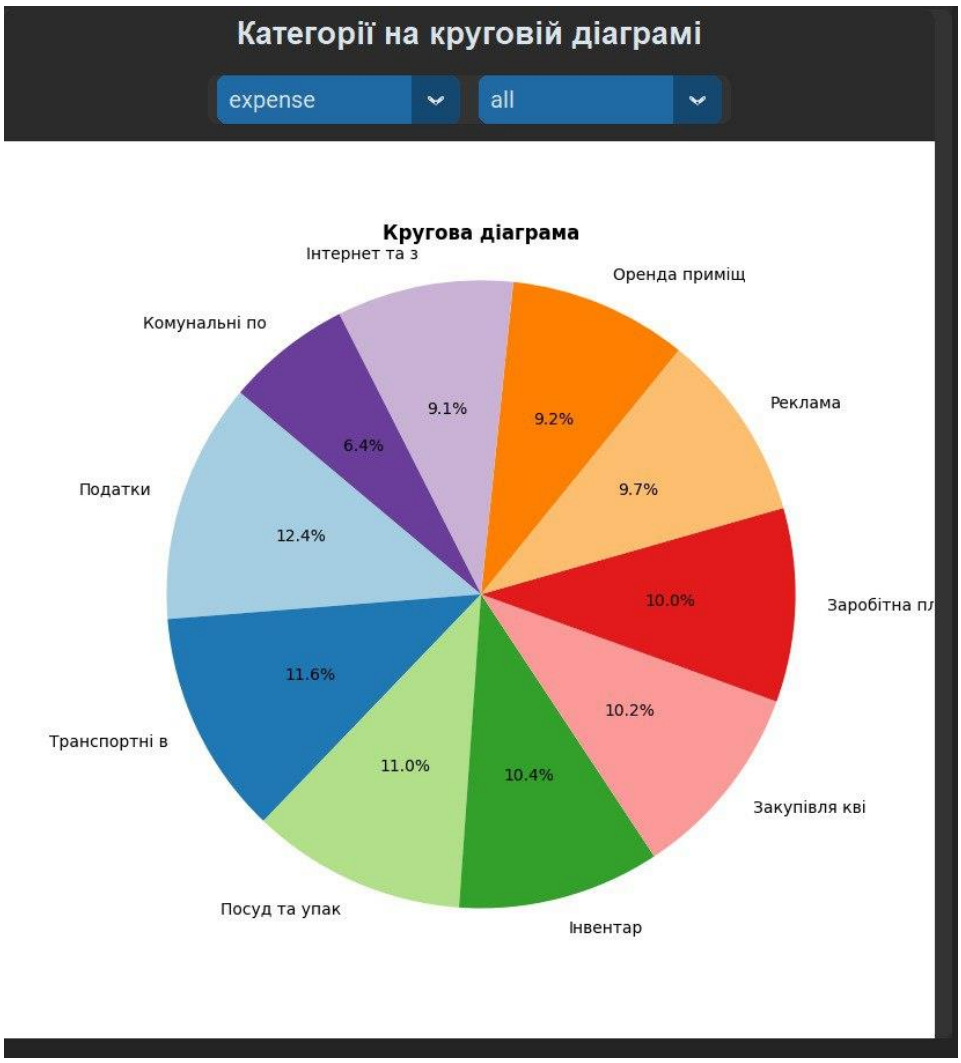


Рисунок 5.9 – Топ категорій

Рисунок 5.9 показує кругову діаграму з структурою витрат у відсотковому співвідношенні. Кожен сектор відображає питому вагу відповідної категорії у загальному обсязі витрат. Такий тип візуалізації дозволяє краще оцінити баланс витрат і виявити, які статті мають найбільший вплив на загальні витрати. Зокрема, видно, що найбільшу частку становлять податки (12.4%), транспортні витрати (11.6%) та витрати на упаковку й посуд (11.0%). Це дає змогу підприємству звернути увагу на ці напрями під час оптимізації бюджету.

Інші суттєві витрати включають інвентар, закупівлю квітів, зарплату, рекламу та оренду приміщень. Наявність таких категорій у діаграмі допомагає проаналізувати ефективність витрат у розрізі функціональних підрозділів компанії.

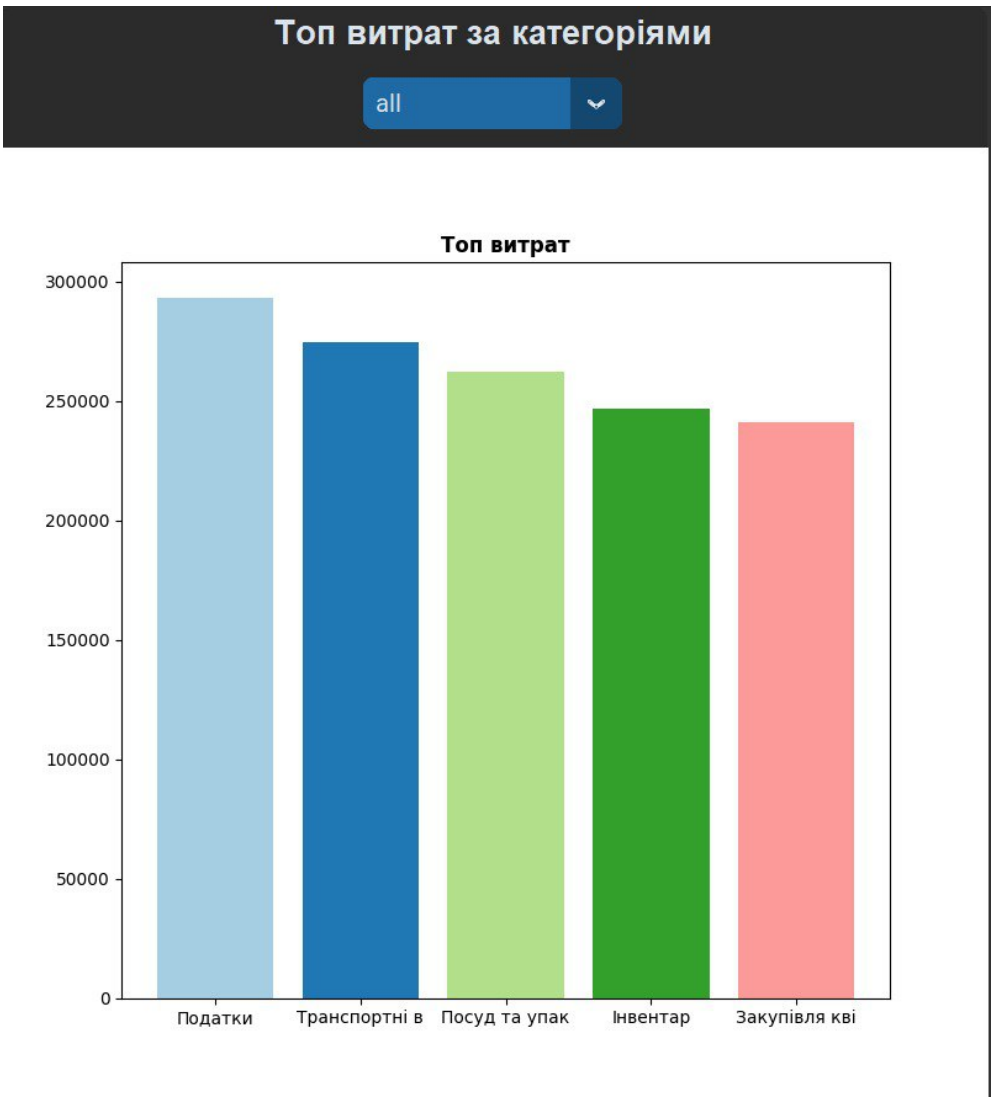


Рисунок 5.10 – Топ витрат за категоріями

На рисунку 5.10 візуалізовано п’ять основних категорій витрат, які за весь час спостереження сформували найбільшу суму фінансових витрат. Такий аналіз є важливою частиною фінансового планування, оскільки дозволяє виявити, куди саме спрямовується найбільше коштів.

У представленому прикладі найбільшу частку витрат займають податки — це свідчить про значне фіскальне навантаження на бізнес, що є типовим для офіційно зареєстрованих підприємств. Наступною за величиною є категорія транспортні витрати, що може включати доставку товарів, логістику та обслуговування автомобілів. Посуд та упаковка є критично важливими для бізнесу, пов’язаного з фізичним товаром (у нашому випадку — з квітами), адже кожен продукт потребує належної презентації.

Також до топу входять інвентар — йдеться про закупівлю обладнання або витратних матеріалів, необхідних для щоденної діяльності, — та закупівля квітів, яка є основою продукційної складової для квіткового бізнесу. Аналізуючи такі дані, власник малого бізнесу може ухвалити рішення щодо оптимізації витрат, наприклад — пошуку більш вигідних постачальників або зменшення частки необов'язкових закупівель.

Крім того, така діаграма дозволяє швидко визначити, які саме категорії потребують детальнішого моніторингу. За потреби користувач може змінити часовий період і подивитися, як змінювались витрати за останній місяць, квартал чи інший відрізок часу — це відкриває можливість для оперативного управлінського контролю та прийняття обґрунтованих фінансових рішень.

Також у застосунку реалізовано окреме вікно експорту, яке дозволяє користувачеві зберігати дані про доходи та витрати у форматах Excel (.xlsx) або CSV (.csv). Вікно відкривається як модальне (CTkToplevel) і забезпечує зручну взаємодію завдяки параметрам фільтрації.

У вікні передбачено:

– вибір типу транзакцій (витрати, доходи, усі);

– вибір періоду (сьогодні, останні 7 днів, цей місяць, увесь час, або вручну заданий інтервал);

– динамічне відображення календарних полів у разі вибору «custom»;

– кнопки для експорту в Excel або CSV.

Після натискання відповідної кнопки відкривається діалог збереження файлу, а дані з бази формуються згідно обраних параметрів. При збереженні у .xlsx створюється таблиця з заголовками; при виборі .csv дані формуються у текстовому вигляді з роздільниками. Реалізовано перевірку правильності введених дат та інформування користувача у разі помилки.

Користувач не має потреби додатково форматувати дані, оскільки експортована інформація вже структурована відповідно до обраного типу транзакцій, категорій, дати, суми та коментарів. Це дозволяє одразу використовувати таблиці у презентаціях, звітності або для внутрішнього аналізу.

Крім того, структура експорту враховує локальні налаштування мови та валюти — це означає, що навіть при роботі в іншомовному інтерфейсі значення залишаються зрозумілими та адаптованими. У перспективі передбачено додавання форматування для друку, а також можливість надсилання звітів електронною поштою.

Цей модуль дозволяє легко створювати резервні копії або ділитися звітами з колегами у зручному форматі. Рисунок 5.11 демонструє розділ експорту даних.

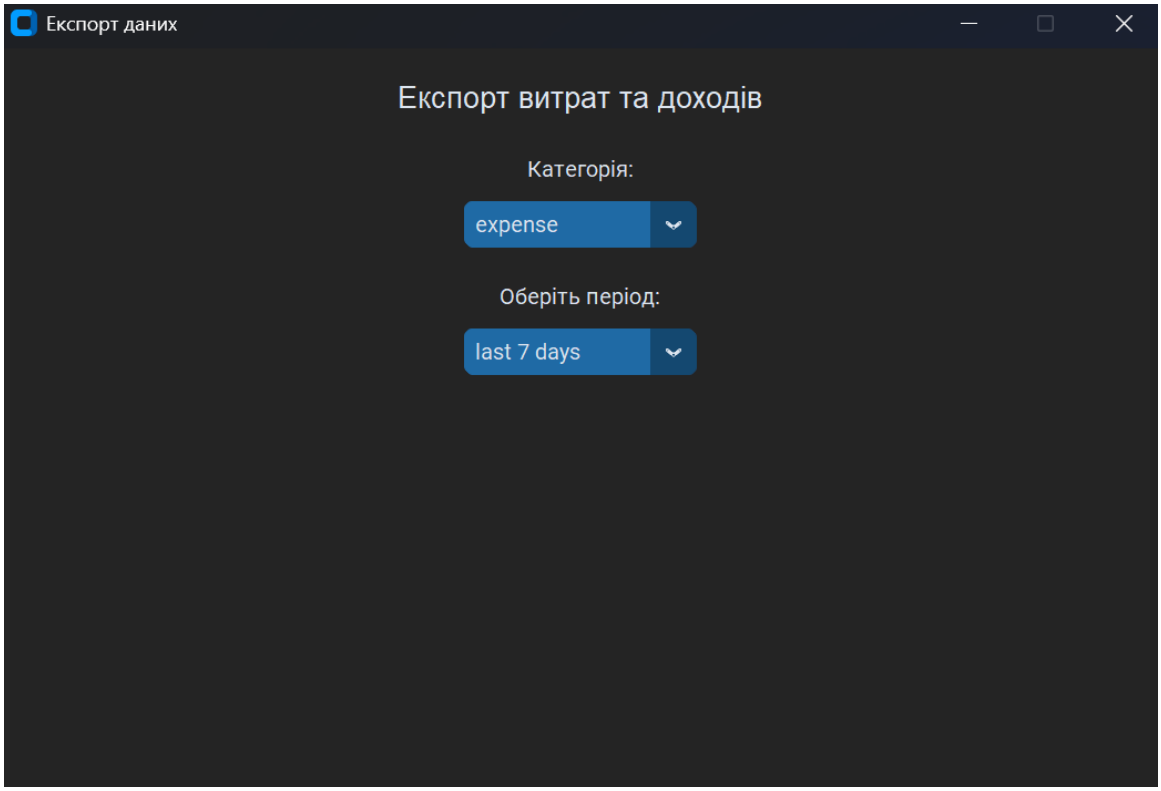


Рисунок 5.11 – Розділ експорту даних

Також до існуючого функціоналу застосунку для ведення бюджету малого і середнього бізнесу було додано унікальне рішення — чат-бот Budget Helper, який виконує роль консультанта на основі мовної моделі. Його оформлення наведено на рисунку 5.12. Інтерфейс бота розроблено у дружньому стилі, що викликає довіру користувачів. Аватарка зі стилізованим роботом і фінансовими символами підкреслює основну функцію асистента — допомагати у фінансовому плануванні.

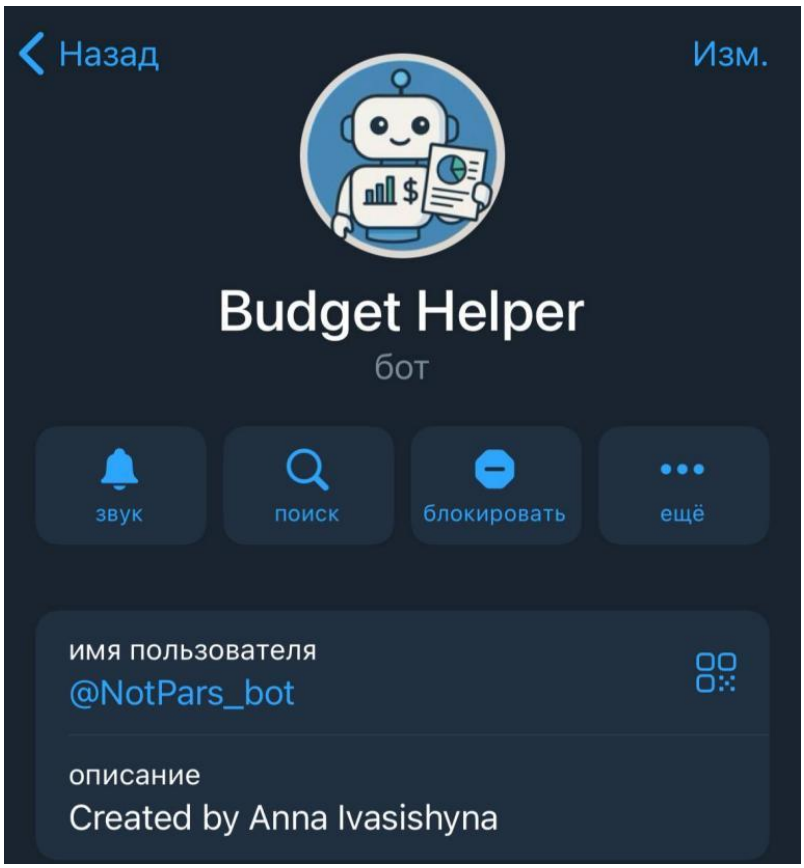


Рисунок 5.12 - Оформлення бота

Цей бот розроблений для оперативного та точкового надання консультацій щодо фінансових звітів та питань клієнтів. Завдяки інтеграції з платформою n8n та спеціально налаштованому промту, бот виступає фінансовим консультантом, який 24/7 доступний для малого бізнесу, якщо штатний бухгалтер або аналітик не можуть бути залучені через обмеження бюджету.

Промт, заданий боту, формулює його завдання таким чином:

– чітко і лаконічно відповідати на запити, використовуючи лише надані дані;

– за потреби запитувати конкретну інформацію для уточнення;

– переадресовувати розмову на фінансову тематику у разі відхилення від теми;

– відповідати українською мовою за замовчуванням, переходячи на англійську лише за вимогою користувача.

У відповідях бот використовує фінансовий аналіз, метрики та індикатори, проводить необхідні розрахунки, оцінює ризики і можливості.

Для роботи бота було налаштовано схему взаємодії з Telegram API та мовною моделлю Gemini. Це забезпечує стабільну і надійну роботу консультанта, здатного оперативно обробляти інформацію у вигляді тексту, таблиць, описів та числових даних.

Основна мета цього модуля — створити додатковий рівень підтримки для малого бізнесу, забезпечуючи цілодобовий доступ до експертної допомоги з фінансових питань, що особливо актуально при обмеженому бюджеті на штатних спеціалістів.

Для реалізації функціоналу чат-бота був використаний сервіс автоматизації n8n, який дозволяє створювати складні робочі процеси з мінімальними зусиллями.

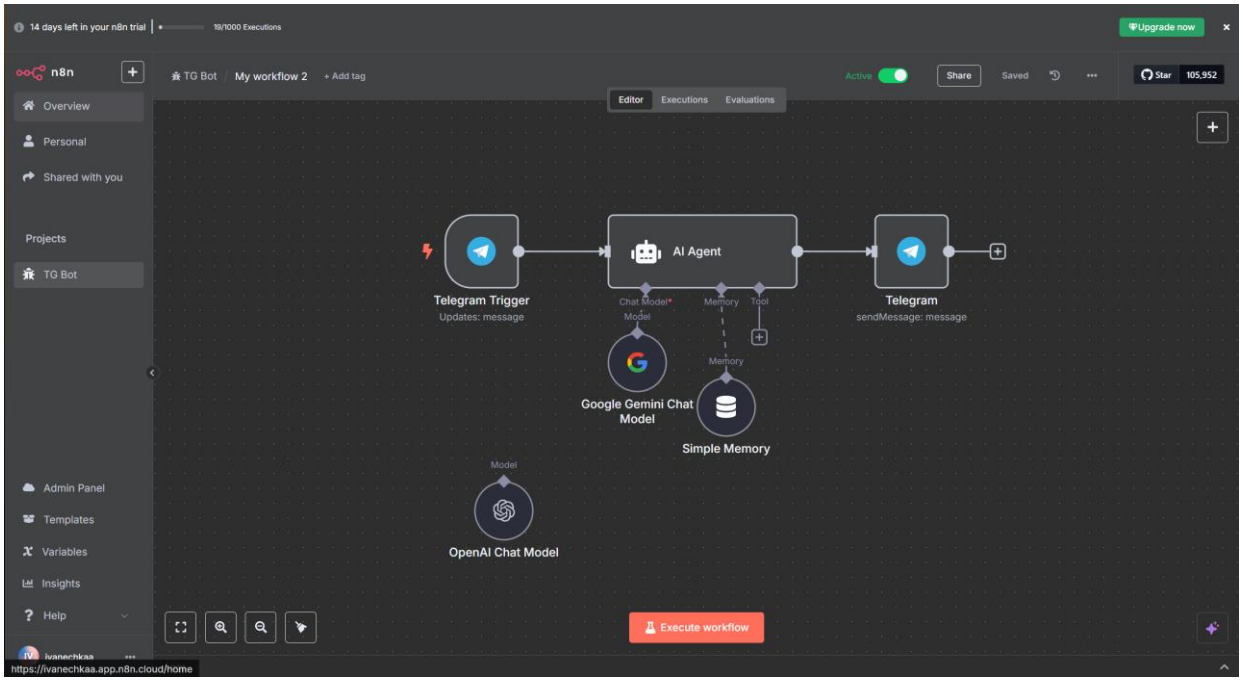


Рисунок 5.13 - Схема інтеграції через n8n

На рисунку 5.13 показана схема інтеграції, де основні компоненти включають:

– Telegram Trigger — вузол, який отримує повідомлення від користувача через Telegram;

– AI Agent — центральний блок обробки запитів, який працює як агент з використанням мовної моделі;

– Google Gemini Chat Model та OpenAI Chat Model — інтегровані моделі штучного інтелекту, що забезпечують генерацію відповідей на основі фінансових даних та заданого промту;

– Simple Memory — механізм збереження контексту розмови, що дозволяє боту враховувати попередні повідомлення користувача для більш релевантних консультацій;

– Telegram — вузол, що відправляє сформовані відповіді назад користувачу в Telegram.

Ця структура дозволяє бот-агенту ефективно отримувати вхідні дані, обробляти їх за допомогою сучасних мовних моделей, зберігати важливу інформацію у пам’яті та відповідати користувачу у зручному месенджері.

Завдяки використанню декількох моделей ШІ, система здатна адаптувати відповіді до різних стилів комунікації або типів запитів: наприклад, Gemini підходить для генерації коротких порад, а OpenAI — для глибших аналітичних пояснень.

Наявність "Simple Memory" дозволяє зберігати короткотермінову історію діалогу, що критично важливо для діалогів, де користувач послідовно уточнює деталі або ставить кілька запитань підряд. Це підвищує якість консультування, оскільки бот зберігає контекст.

Використання платформи n8n надає гнучкість у налаштуванні логіки роботи бота, а також спрощує інтеграцію з API Telegram та сервісами штучного інтелекту. Це рішення дозволяє забезпечити постійну доступність консультанта для малого бізнесу, що значно підвищує якість і швидкість прийняття фінансових рішень. У майбутньому цю схему можна масштабувати: додати вузли для аналізу витрат, інтеграції з CRM або автоматичної генерації фінансових порад.

На рисунку 5.14 наведено приклад щотижневого звіту, який включає загальні показники прибутку: загальні доходи, витрати та чистий прибуток. Нижче подано деталізацію — які категорії принесли найбільше доходу (наприклад, продаж кімнатних рослин, інші доходи, доставка), а також основні статті витрат (інші витрати, комунальні послуги, зарплата).

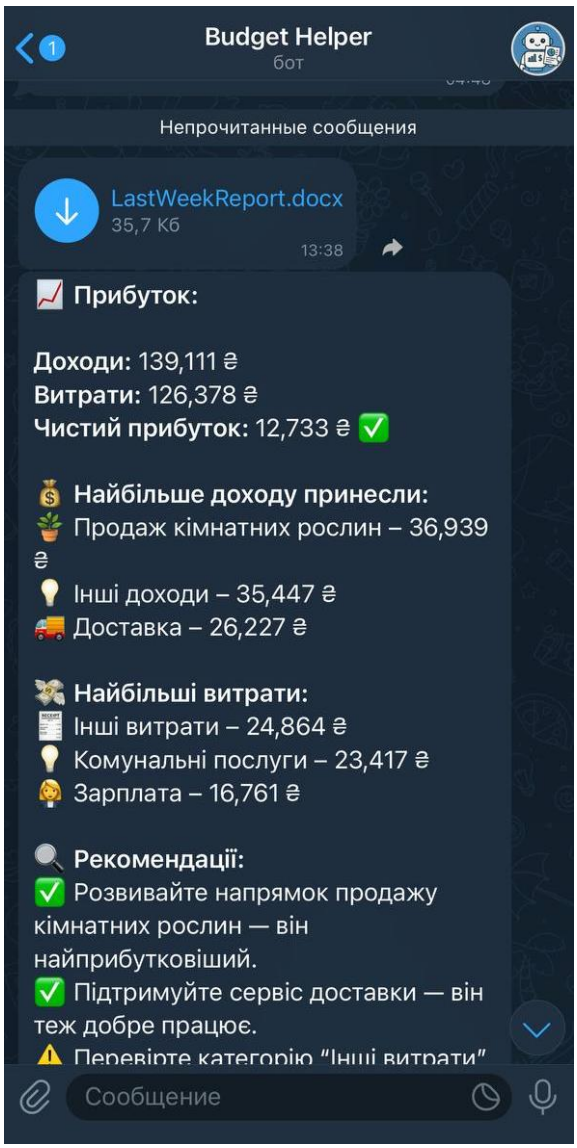


Рисунок 5.14 – Приклад щотижневого звіту

Окрім сухих цифр, бот автоматично формує короткі рекомендації — наприклад, розвивати напрямок продажу, звернути увагу на високу частку витрат у певних категоріях тощо. Це дозволяє користувачеві не лише ознайомитись із поточними результатами, а й отримати практичні поради щодо оптимізації бюджету, зменшення витрат або розвитку прибуткових напрямів.

Формат звіту структурований і зрозумілий — використання емодзі допомагає швидко зорієнтуватись у категоріях, візуально розділяє блоки та полегшує сприйняття навіть на мобільному пристрої.

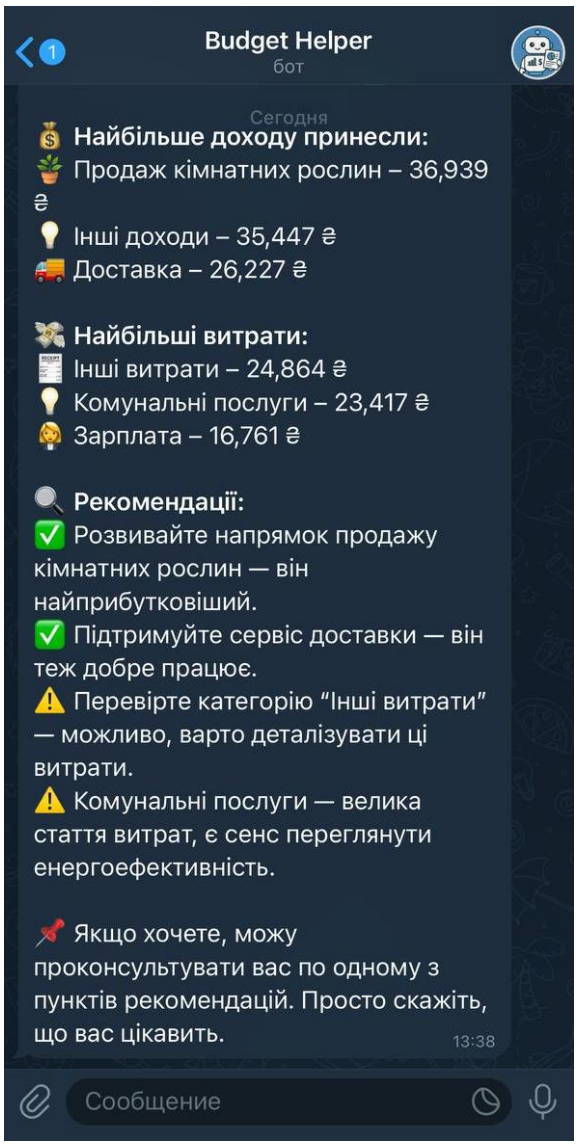


Рисунок 5.15 – Продовження прикладу щотижневого звіту

Рисунок 5.15 демонструє продовження звіту, де показано ті ж самі дані, але з розширеним блоком рекомендацій. Бот не лише аналізує поточні дані, а й порівнює їх із попередніми звітами, виокремлює зміни та допомагає користувачу ухвалювати зважені рішення. Наприкінці бот також пропонує індивідуальну консультацію — достатньо написати, яка з рекомендацій цікавить найбільше.

Таким чином, Budget Helper виконує не просто роль інформатора, а діє як віртуальний фінансовий асистент, який об'єднує у собі функції аналітика, консультанта і помічника з прийняття рішень. Завдяки цьому бот суттєво підвищує ефективність управління фінансами та заощаджує час користувачів, автоматизуючи ру-

тинні процеси аналізу. Його використання дозволяє підприємцям оперативно реагувати на зміни у фінансових показниках та приймати обґрунтовані рішення на основі актуальних даних.

Бот особливо корисний для малих бізнесів, які не мають окремого фінансового аналітика, але прагнуть прозорості та контролю над грошовими потоками. Крім того, він сприяє підвищенню фінансової грамотності, пропонуючи пояснення та поради у доступній формі.

Розробка Budget Helper стала відповіддю на потребу малого та середнього бізнесу в доступному інструменті для щоденного фінансового контролю без залучення складного програмного забезпечення. Бот легко інтегрується з системою обліку, отримує необхідні дані автоматично та перетворює їх на зрозумілі звіти. Його інтерфейс інтуїтивно простий, а комунікація відбувається в зручному для користувача середовищі — месенджері Telegram. Завдяки цьому навіть користувачі без спеціальної освіти можуть отримати корисну інформацію про стан справ у бізнесі, виявити проблемні ділянки та запобігти фінансовим втратам ще до того, як вони стануть критичними.

5.3 Архітектура системи

Система для ведення особистого бюджету реалізована у вигляді десктопного застосунку з графічним інтерфейсом користувача, створеним за допомогою бібліотеки CustomTkinter на мові програмування Python. Архітектура програми спроектована таким чином, щоб забезпечити розділення відповідальностей між різними функціональними модулями, що сприяє зручності розробки, розширення та підтримки коду.

Компоненти системи

Графічний інтерфейс користувача (GUI)

Інтерфейс користувача організовано у вигляді окремих вікон, які реалізують основні функції системи:

Вікно експорту даних (ExportWindow) надає можливість вибору категорій (витрати, доходи, або обидві), періоду (зокрема сьогодні, останні 7 днів, поточний місяць, весь час або власний інтервал) та формату експорту (Excel, CSV).

Вікно аналітики (AnalyticsWindow) містить графічні віджети для візуалізації тенденцій доходів і витрат, а також для відображення топових категорій у вигляді стовпчикових і кругових діаграм. Візуалізація здійснюється за допомогою бібліотеки matplotlib з інтеграцією у GUI через спеціальний віджет FigureCanvasTkAgg.

Вікно налаштувань (SettingsWindow) забезпечує користувачу вибір мови інтерфейсу та валюти, з подальшим збереженням цих параметрів.

Для зручного вибору дат у формі експорту використовується віджет DateEntry із бібліотеки tkcalendar, що дозволяє інтуїтивно задавати початкову та кінцеву дати для фільтрації.

Збереження даних

Дані про доходи та витрати зберігаються у базі даних SQLite, що є вбудованим рішенням з низькими вимогами до налаштувань і високою швидкодією для локальних застосунків. Таблиця expenses містить записи із полями, що включають дату, категорію, тип операції (дохід або витрата), суму та опис.

Вказані сутності та їхні зв’язки подано у вигляді ER-діаграми на кресленику ІС11.12БАК.005 Д3.

Взаємодія із базою даних здійснюється через модуль sqlite3 стандартної бібліотеки Python. Запити до БД організовані таким чином, щоб повертати дані, відфільтровані за категорією та часовим інтервалом відповідно до вибору користувача.

Обробка та фільтрація даних

Логіка фільтрації даних передбачає вибір категорії (дохід, витрата, або обидві), а також визначення періоду часу. Для періодів передбачені стандартні значення («сьогодні», «7 днів», «30 днів», «365 днів», «весь час») і можливість кастомного вибору дат.

Отримані дані використовуються для подальшої агрегації та побудови звітів, зокрема для обчислення підсумків по тижнях, топових категорій за сумами витрат чи доходів, що відображається у вигляді різних типів графіків.

Опис процесів з використанням діаграми BPMN (Business Process Model and Notation) наведено у кресленику ІС11.12БАК.005 Д4.

Візуалізація даних

Для представлення аналітичної інформації використовується бібліотека matplotlib, що дозволяє створювати різноманітні графіки: лінійні тренди, стовпчикові діаграми та кругові діаграми. Відображення графіків у вікнах Tkinter реалізовано за допомогою FigureCanvasTkAgg, що забезпечує інтеграцію візуалізації безпосередньо у GUI. Загальну архітектуру системи та взаємозв'язки між її компонентами подано на рисунку 5.14.

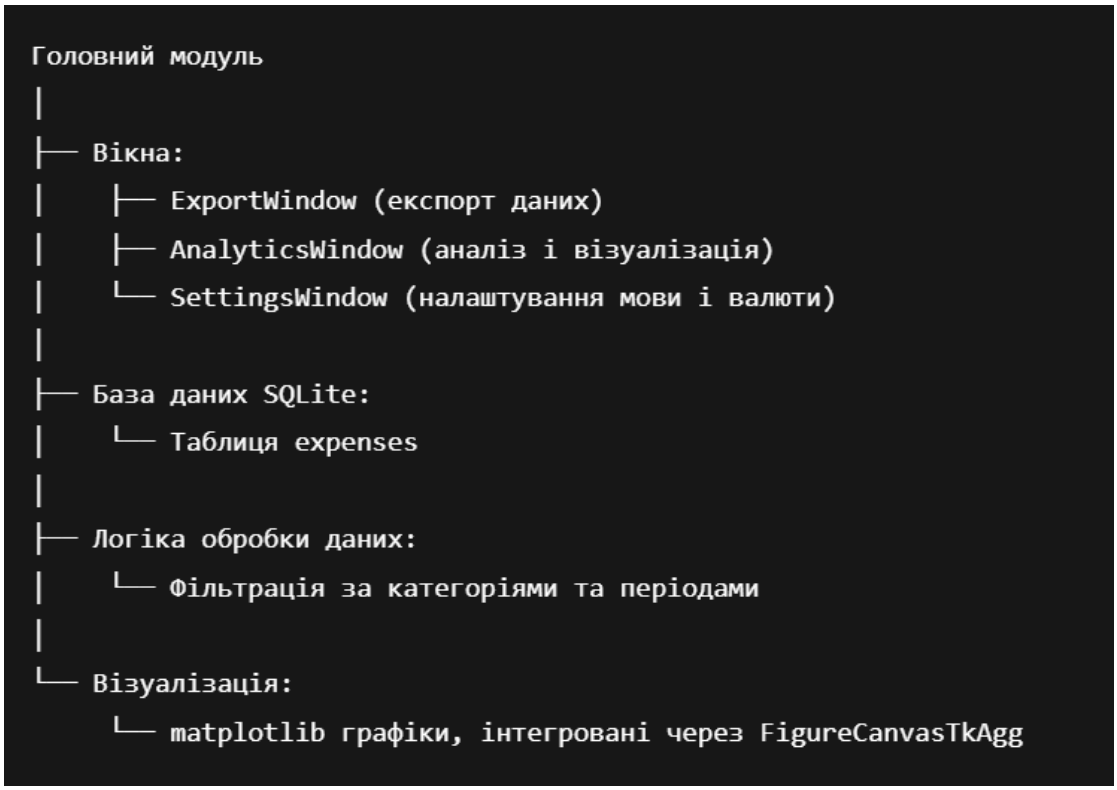


Рисунок 5.14 – Загальна структура системи

Особливості реалізації

Застосування змінних стану (tk.StringVar) для забезпечення динамічного оновлення інтерфейсу відповідно до змін фільтрів.

Валідація введених користувачем даних, зокрема перевірка правильності інтервалів дат.

Забезпечення можливості збереження та експорту даних у популярних форматах, що підвищує зручність використання додатку.

Таким чином, архітектура системи відповідає вимогам масштабованості, зручності використання та подальшого розвитку, поєднуючи простоту реалізації з функціональністю, необхідною для ефективного управління особистими фінансами.

Діаграму розробленого алгоритму зображено у кресленику ІС11.12БАК.005 Д2.

Висновки до розділу 5

У п’ятому розділі дипломної роботи було детально описано процес розробки інформаційної системи обліку фінансових операцій для малого бізнесу. Розглянуто архітектуру системи, її основні модулі та логіку взаємодії між ними. Здійснено реалізацію інтерфейсу користувача із використанням бібліотеки tkinter, реалізовано функції додавання, редагування, перегляду та видалення фінансових операцій, а також формування звітів і побудови графіків на основі даних.

Особливу увагу приділено структурі бази даних, яка побудована на основі SQLite та забезпечує зберігання і обробку транзакцій у локальному режимі. Реалізовано логіку валідації введених даних, обробки помилок, а також побудову аналітичних діаграм з використанням matplotlib. Усі функціональні модулі протестовано, підтверджено коректність їхньої роботи відповідно до вимог, сформованих на попередніх етапах проєкту.

Загалом, у межах цього розділу вдалося реалізувати стабільний, інтуїтивно зрозумілий та придатний до розширення програмний продукт, який відповідає поставленій меті та вимогам системи. Створене рішення дозволяє ефективно вести фінансовий облік, здійснювати базову аналітику та візуалізацію даних у зручному для користувача вигляді.

6 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

6.1 Змістовна постановка задачі

Метою створення інформаційної системи є автоматизація процесів обліку фінансових операцій у малому бізнесі. До ключових завдань, які система повинна виконувати, належить фіксація доходів і витрат, збереження записів про транзакції, їх групування за категоріями, а також формування підсумкових звітів і візуалізація фінансових даних. На основі цих даних користувач повинен мати змогу оцінювати загальний фінансовий стан, аналізувати структуру витрат та доходів, визначати тенденції та ухвалювати обґрунтовані управлінські рішення.

З огляду на це, постає задача автоматизованого обчислення сукупних значень по вибраних параметрах (тип, категорія, дата), обчислення залишку коштів на рахунках, а також побудови графіків, що відображають зміну фінансових показників у часі. Таким чином, задачі мають не лише обліковий, а й аналітичний характер.

6.2 Математична постановка задачі

Розроблювана система оперує множиною фінансових операцій, яку можна описати як множину:

$T = \{t_1, t_2, ..., t_n\}$,

де кожна операція t_i — це кортеж:

$t_i = (d_i, s_i, c_i, k_i, t_{ip})$,

де:

– d_i — дата операції,

– s_i — сума (дійсне число),

– c_i — категорія,

– k_i — контрагент,

– t_{ip} — тип операції (дохід/витрата/переказ), тобто $t_{ip} \in \{+1, -1, 0\}$.

Задача полягає у знаходженні агрегованих значень по підмножинах T:

– обчислення загальної суми доходів:

$$S_{\text{доходів}} = \sum_{i=1}^{n} s_i * \delta(t_{ip} = +1) \quad (6.1)$$

– обчислення загальної суми витрат:

$$S_{\text{витрат}} = \sum_{i=1}^{n} s_i * \delta(t_{ip} = -1) \quad (6.2)$$

– баланс:

$$B = S_{\text{доходів}} - S_{\text{витрат}} \quad (6.3)$$

– побудова ряду значень за днями/місяцями:

$$f(t) = \sum_{i=1}^{n} s_i * \delta(d_i \in t)\,, \quad (6.4)$$

де t — часовий інтервал.

Функція δ(умова) — індикаторна функція, яка дорівнює 1, якщо умова істинна, і 0 — інакше.

6.3 Обґрунтування методу розв’язання

Для вирішення задачі обліку та аналізу фінансових операцій можуть бути використані кілька підходів, зокрема:

– використання табличних процесорів (Excel, Google Sheets);

– створення звітів вручну або за допомогою готових шаблонів;

– застосування спеціалізованих платформ типу ERP-систем;

– створення автоматизованих програмних рішень на основі баз даних та алгоритмів обробки даних.

Перші два варіанти неефективні при зростанні обсягу даних, а ERP-системи надмірно складні та дорогі для малого бізнесу.

Обраним методом є розробка компактної програми на мові Python з використанням реляційної бази даних SQLite, яка дозволяє реалізувати всі необхідні функції за допомогою простих математичних операцій: сумування, групування, фільтрація, підрахунок залишку. Для візуалізації буде застосовано бібліотеку matplotlib, яка дозволяє будувати графіки на основі отриманих агрегованих значень.

Цей метод забезпечує:

– достатню точність обчислень;

– гнучкість у роботі з базою даних;

– швидкість обробки;

– можливість масштабування;

– простоту реалізації без використання складних статистичних або фінансових моделей.

6.4 Опис методу розв’язання

Для реалізації обчислень було використано стандартні засоби Python та SQL-запити до бази даних SQLite.

Основні кроки алгоритму:

1) завантаження бази даних та підключення до таблиці транзакцій.

2) фільтрація записів за заданими критеріями (тип операції, дата, категорія).

3) агрегація даних:

– підрахунок загальної суми доходів і витрат;

– розрахунок балансу;

– групування по категоріях та по періодах.

4) Побудова графіків (стовпчикові або лінійні діаграми) для:

– витрат за категоріями [9];

– щомісячної динаміки витрат/доходів;

– співвідношення типів транзакцій.

Приклад реалізації наведено на рисунку 6.1.

```
import sqlite3
import matplotlib.pyplot as plt

conn = sqlite3.connect("budget.db")
cursor = conn.cursor()

cursor.execute("""
SELECT category, SUM(amount)
FROM expenses
WHERE type = 'expense'
GROUP BY category
""")
data = cursor.fetchall()

categories = [row[0] for row in data]
amounts = [row[1] for row in data]

plt.bar(categories, amounts)
plt.title("Витрати за категоріями")
plt.xlabel("Категорія")
plt.ylabel("Сума")
plt.show()
```

Рисунок 6.1 – Фрагмент коду побудови діаграми витрат за категоріями на основі даних з бази даних SQLite

Висновки до розділу 6

У цьому розділі було визначено математичне підґрунтя функціонування системи. Задача обліку фінансових операцій була формалізована, обґрунтовано вибір методів її розв’язання та описано конкретні алгоритми, що використовуються для підрахунків і побудови аналітичних графіків.

7 ТЕСТУВАННЯ СИСТЕМИ

7.1 Мета випробувань

Основною метою тестування розробленої системи управління особистим бюджетом було забезпечення коректної, надійної та стабільної роботи всіх функціональних можливостей, що реалізовані у додатку. З огляду на специфіку роботи із фінансовими даними, тестування особливо сфокусоване на точності обробки витрат і доходів, а також на правильності взаємодії між користувацьким інтерфейсом (реалізованим на CustomTkinter) та базою даних.

Ключовим завданням було гарантувати, що:

Всі операції додавання, редагування та видалення витрат і доходів коректно зберігаються в базі;

Фільтрація даних за категоріями та часовими періодами працює без помилок;

Експорт даних у формати Excel та CSV виконується з урахуванням обраних фільтрів;

Інтерфейс реагує на дії користувача без збоїв, а елементи керування (DateEntry, OptionMenu) працюють правильно;

Забезпечена стабільність і безперервність роботи застосунку навіть при нетипових сценаріях, наприклад, пустих виборах дат, відсутності витрат за період чи пошкодженні бази.

7.2 Загальні положення

Для забезпечення високої якості та надійності розробленої системи управління бюджетом було обрано комплексний підхід до тестування, що включає три основні типи: модульне, інтеграційне та приймальне тестування. Такий підхід дає змогу провести детальну перевірку роботи програмного продукту на різних рівнях, що сприяє своєчасному виявленню помилок і підвищенню якості кінцевого результату.

Організація тестування відповідала принципу «піраміди тестування», згідно з яким більша частина тестів належить до модульних. Вони швидкі, автономні та спрямовані на перевірку окремих функціональних блоків системи. Наступним рівнем є інтеграційні тести, які контролюють правильність взаємодії між компонентами, а найменшу, але не менш важливу частку складають приймальні тести, що відтворюють реальні сценарії роботи кінцевого користувача.

Застосування такого підходу дозволяє оптимізувати витрати часу на тестування і мінімізувати ймовірність появи дефектів на пізніх етапах розробки, які можуть спричинити значні витрати на виправлення.

Модульне тестування виконувало функцію перевірки логіки окремих компонентів, зокрема функцій роботи з базою даних, обробки категорій та фінансових операцій. Інтеграційні тести забезпечували коректність взаємодії між різними частинами системи — наприклад, перевірку експорту даних у формати Excel і CSV, а також роботу механізму авторизації користувачів. Приймальне тестування ґрунтувалося на сценаріях, що імітували типові дії користувачів, з метою підтвердження відповідності системи вимогам.

Така комплексна система тестування дозволила забезпечити стабільну роботу програмного продукту, що є особливо важливим для застосувань, пов’язаних з фінансовою інформацією, де точність і надійність мають першочергове значення.

7.3 Модульне тестування

Модульне тестування є одним із фундаментальних етапів забезпечення якості програмного забезпечення, оскільки спрямоване на перевірку коректності роботи окремих функціональних блоків системи у відокремленому середовищі. В контексті розробленої системи управління бюджетом основна увага була зосереджена на тестуванні логіки взаємодії з базою даних, а також функцій додавання та видалення фінансових записів.

Для реалізації модульних тестів використовувався фреймворк unittest, а для ізоляції від зовнішніх ресурсів — база даних SQLite у режимі in-memory. Такий

підхід дозволяє швидко виконувати тести без необхідності роботи з фізичними файлами, що забезпечує повторюваність і незалежність тестів від стану зовнішнього середовища.

Основним об’єктом тестування була функція get_expenses, що реалізує вибірку записів із бази даних із застосуванням різноманітних фільтрів за категоріями (витрати, доходи або обидва типи) та часовими проміжками (поточний день, останні сім днів, поточний місяць, користувацький період). У ході тестування перевірялась відповідність отриманих даних заданим параметрам, коректність роботи з датами, а також правильність формування результатів у різних сценаріях.

Крім того, особлива увага приділялась функціям додавання нових записів, що передбачає не лише створення витрати або доходу, але й автоматичне додавання нової категорії у випадку її відсутності. Тести перевіряли, що після додавання нової категорії вона коректно зберігається у базі даних без дублювань, а також що всі параметри запису (сума, дата, категорія) передаються і зберігаються належним чином.

Ще одним важливим аспектом була перевірка функції видалення фінансових записів. Тести демонстрували, що після вилучення останнього запису певної категорії ця категорія також видаляється з бази даних. Такий підхід дозволяє підтримувати базу даних у актуальному стані, уникати накопичення непотрібних даних та забезпечувати чистоту структури. Особливу увагу приділено поведінці інтерфейсу у випадках, коли користувач не обрав жодного запису перед видаленням. У таких ситуаціях система відображає діалогове вікно з попередженням про необхідність вибрати хоча б один елемент. При спробі видалення записів без попереднього вибору система відображає відповідне попередження, приклад наведено на рисунку 7.1. Цей механізм захисту від помилкових дій є прикладом превентивної взаємодії з користувачем, що мінімізує ризики втрати даних або системних збоїв.

Крім того, система забезпечує зворотний зв’язок у зручному форматі, що сприяє покращенню загального користувацького досвіду та дозволяє уникати непорозумінь при роботі з великою кількістю транзакцій.

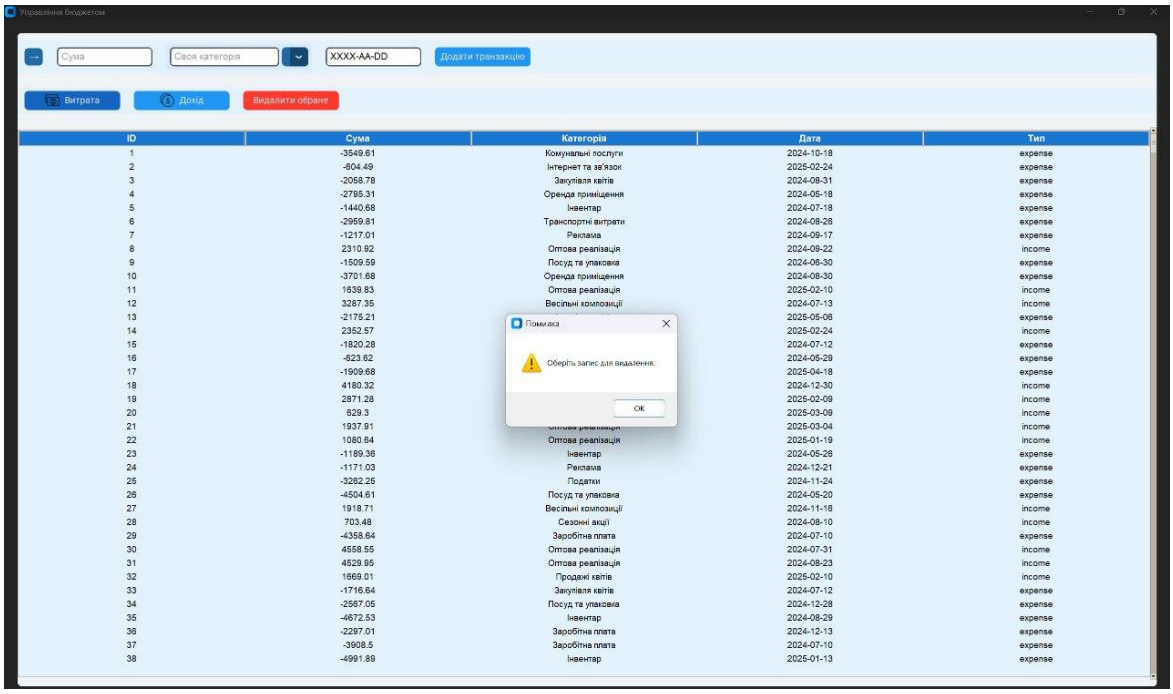


Рисунок 7.1 – Попередження для користувача про необхідність вибрати записи для видалення

Подібні запобіжники в інтерфейсі дозволяють уникати випадкового видалення важливої інформації, що підвищує надійність системи і покращує користувацький досвід. Повідомлення оформлене у стандартному стилі з іконкою попередження та кнопкою підтвердження (OK), що забезпечує зрозумілу взаємодію навіть для новачків.

Окремо тестувалась логіка взаємодії інтерфейсу користувача з функціоналом вибору дат та категорій. Віджети вибору дат (DateEntry) і меню категорій (CTkOptionMenu) піддавались перевірці на коректність зміни станів, зокрема відображення додаткових елементів при виборі нестандартного періоду ("custom"). Це важливо для зручності користувача та запобігання помилкам під час формування звітів. На рисунках 7.2–7.4 показано інтерфейс вибору періоду та категорій, механізм додавання нових категорій, а також приклад попередження при введенні некоректної дати. Перевірка охоплювала як стандартні сценарії взаємодії, так і граничні випадки, що дозволило виявити потенційні помилки та підвищити стабільність роботи інтерфейсу.

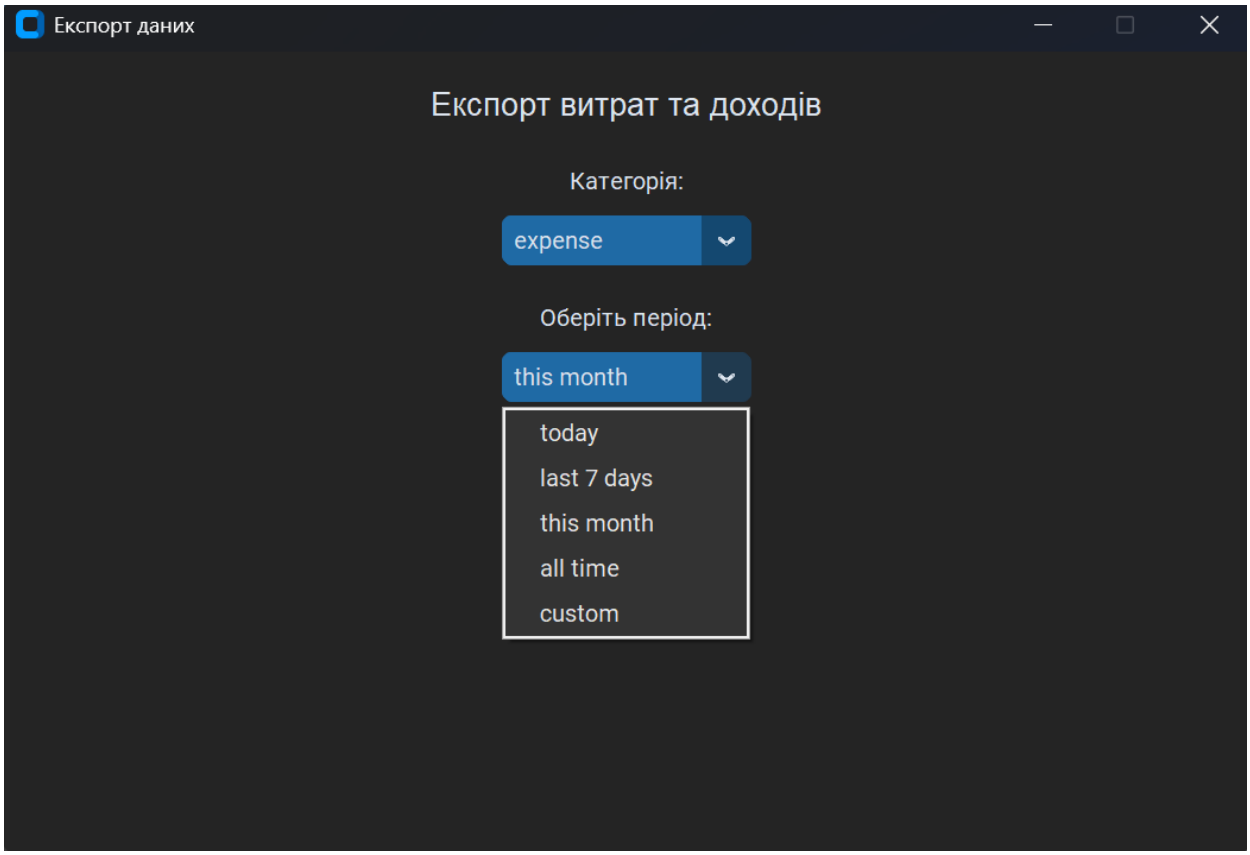


Рисунок 7.2 – Вікно з інтерфейсом вибору періоду та категорії перед експортом.

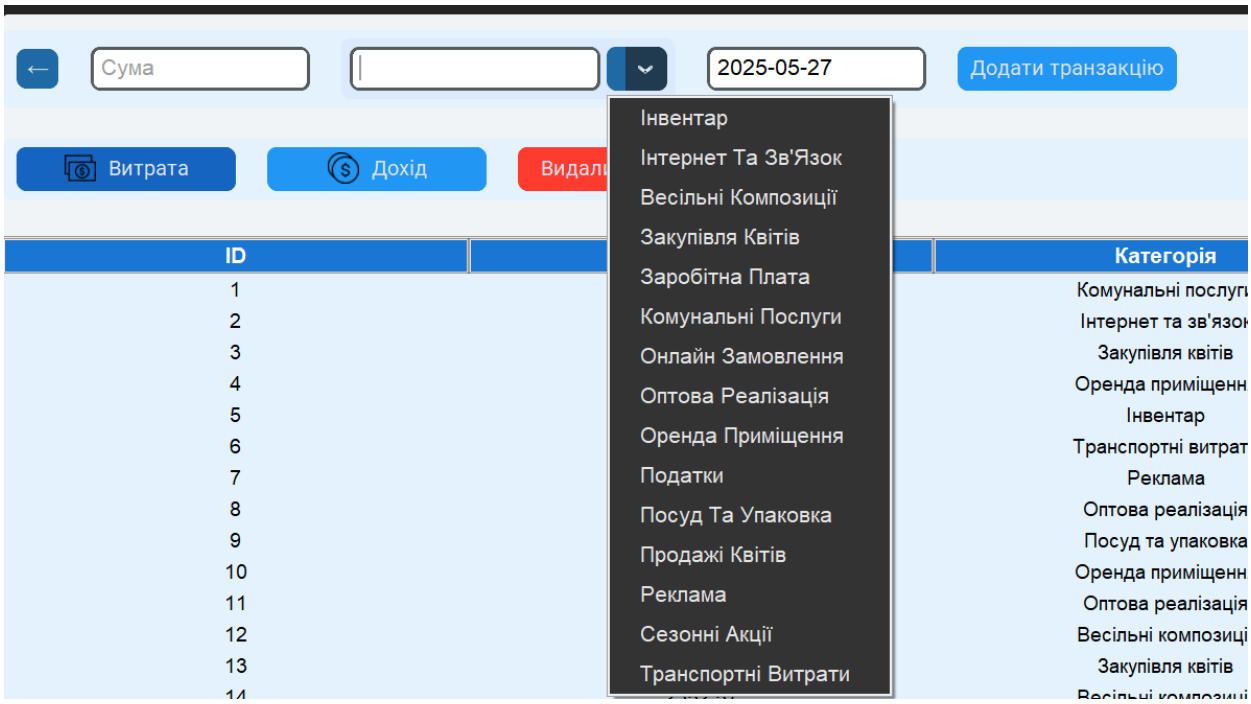


Рисунок 7.3 – Відображення додавання нової категорії та вже існуючих

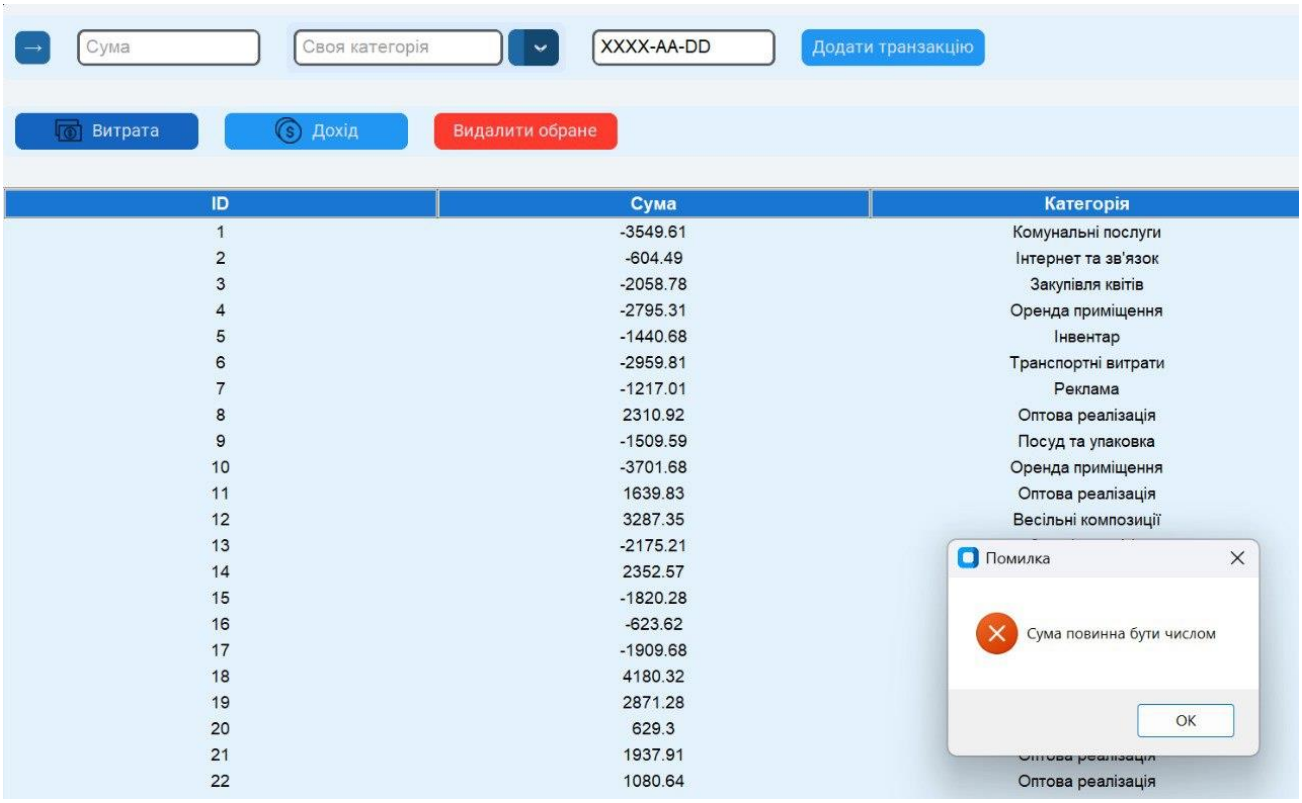


Рисунок 7.4 – Попередження у разі введення некоректної дати

7.4 Інтеграційне тестування

В інтеграційних тестах основна увага приділялась перевірці взаємодії між трьома ключовими компонентами:

UI (CustomTkinter) — обробка вибору періоду, категорії, запуск експорту;

Логіка фільтрації та отримання даних (функції з database.py);

Експортні функції (export_to_excel, export_to_csv), що створюють файли на основі отриманих фільтрованих даних.

Типові сценарії

Користувач обирає період "last 7 days" та категорію "expense" → дані коректно фільтруються → виконується експорт у Excel → файл містить лише витрати за останній тиждень;

Вибір "custom" дати з 01.05.2025 по 15.05.2025, категорія "income" → експорт у CSV → файл відповідає вибраному періоду і категорії;

Перевірка роботи кнопок експорту при відсутності витрат (порожній результат) — система не падає, виводить повідомлення або створює пустий файл.

Автоматизація

Частина інтеграційних тестів виконувалась автоматично за допомогою тестових скриптів, що симулювали зміну значень у category_var, range_var, запускали методи export_to_excel і export_to_csv, а потім перевіряли існування і вміст згенерованих файлів.

7.5 Приймальне тестування

Приймальне тестування проводилося вручну із залученням користувачів, які тестували інтерфейс у реальних умовах. Вони виконували типові дії:

Вибір категорії "both", "expense", "income" та перевірка коректності відображення відповідних записів;

Вибір швидких періодів ("today", "this month") та власного діапазону дат із допомогою DateEntry;

Експорт даних у Excel та CSV з подальшим відкриттям файлів для перевірки правильності даних та форматування;

Перевірка поведінки при введенні некоректних дат (початкова дата після кінцевої), при відсутності даних за період.

Виявлені недоліки і їх виправлення

Було помічено, що при виборі кастомного періоду не завжди коректно оновлюються результати фільтрації — додано додаткові перевірки і оновлення UI після зміни дат;

Виявлено, що при експорті порожніх наборів даних створюються файли з помилками — додано перевірку на пустий результат і відповідне повідомлення користувачу;

Покращено підказки в інтерфейсі, щоб уникнути плутанини із форматами дат. На рисунку 7.5 продемонстровано приклад повідомлення про помилку, що виникає при виборі діапазону дат, за якими не знайдено жодного запису.

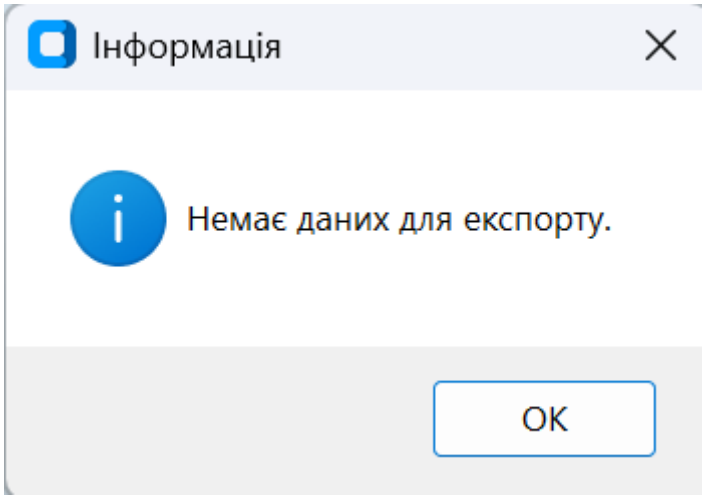


Рисунок 7.5 – Повідомлення про помилку при виборі діапазону дат без даних

7.6 Результати тестування

Проведене тестування показало, що розроблена система управління бюджетом є стабільною, надійною і зручною у використанні. Конкретно:

Модульні тести підтвердили коректність функцій взаємодії з базою, правильність логіки обробки даних витрат та доходів;

Приймальне тестування показало високу зручність інтерфейсу, що побудований на CustomTkinter, та задоволеність користувачів, які змогли ефективно управляти бюджетом та експортувати звіти.

У підсумку система відповідає вимогам технічного завдання і готова до подальшої експлуатації.

Висновки до розділу 7

У сьомому розділі підсумовано результати виконаної дипломної роботи. Сформульовано оцінку розробленої інформаційної системи, визначено її практичну цінність для малого бізнесу, а також окреслено можливі напрями подальшого розвитку.

ВИСНОВКИ

У дипломній роботі було розроблено та реалізовано інформаційну систему обліку фінансових операцій для малого бізнесу, яка дозволяє автоматизувати щоденний облік доходів, витрат та переказів, а також формувати звіти й графічну візуалізацію на основі зібраних даних. Розроблена система повністю відповідає поставленим у вступі цілям та задачам, зокрема:

– проведено аналіз предметної області, виявлено специфіку фінансового обліку в малому бізнесі;

– досліджено та проаналізовано наявні програмні рішення з аналогічною функціональністю;

– сформульовано функціональні та нефункціональні вимоги до інформаційної системи;

– обґрунтовано вибір технологій розробки — мови Python, бази даних SQLite та бібліотек для візуалізації;

– реалізовано програмну систему, що забезпечує збереження фінансових даних, групування, підсумковий аналіз і побудову графіків;

– розглянуто математичну модель задачі обліку операцій і описано алгоритми обробки даних.

Отримані результати повністю відповідають сучасному рівню технічних знань у галузі розробки прикладного програмного забезпечення. Створена система використовує відкриті, доступні й популярні інструменти, що дозволяє легко адаптувати її для потреб конкретного бізнесу.

З практичної точки зору, система може бути впроваджена у малих компаніях, індивідуальних підприємців або проєктних групах, де немає потреби у складному бухгалтерському обліку, але необхідна простота, зручність та наочність у контролі за фінансами. Вона не потребує ліцензування, працює автономно та може бути адаптована до різних мовних або валютних стандартів. Робота має соціально-економічне значення, оскільки сприяє цифровізації фінансової діяльності малого

бізнесу та зменшенню залежності від великих хмарних сервісів, що іноді є недоступними, надто складними або фінансово обтяжливими.

З науково-технічної точки зору було реалізовано повний цикл проєктування інформаційної системи — від постановки задачі до її програмної реалізації з урахуванням вимог надійності, структури бази даних, збереження інформації, а також інтерфейсної та функціональної зручності. Використано принципи модульності, мінімізації зовнішніх залежностей і масштабованості.

Результати цієї роботи можуть слугувати основою для подальших досліджень у таких напрямках:

– розширення функціоналу за рахунок додавання аналітичного модуля з прогнозуванням на основі історичних даних;

– реалізація багатокористувацького доступу з ролями (адміністратор, касир, аналітик тощо);

– інтеграція з банківськими API для автоматичного імпорту транзакцій;

– створення веб-версії або мобільного застосунку для зручного доступу з будь-якого пристрою.

Загалом, всі поставлені завдання у межах дипломного проєкту виконано повною мірою, а кінцевий результат має як прикладну, так і навчальну цінність. Отримана система є готовим продуктом, придатним до практичного використання, а її подальший розвиток може бути реалізований у межах магістерського або інноваційного проєкту.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Python Programming Language – Official Website. URL: https://www.python.org (дата звернення 03.03.2025).
2. SQLite – Lightweight Database Engine. URL: https://www.sqlite.org (дата звернення 08.03.2025).
3. Matplotlib: Visualization with Python – Official Docs. URL: https://matplotlib.org/stable/index.html (дата звернення 04.03.2025).
4. Tkinter – Python Interface to Tcl/Tk. URL: https://docs.python.org/3/library/tk.html (дата звернення 10.03.2025).
5. CustomTkinter – Modern GUI for Python. URL: https://github.com/TomSchimansky/CustomTkinter (дата звернення 14.03.2025).
6. Visual Studio Code – Source-code editor. URL: https://code.visualstudio.com (дата звернення 03.03.2025).
7. Git – Distributed Version Control System. URL: https://git-scm.com (дата звернення 18.03.2025).
8. Python SQLite Tutorial – W3Schools. URL: https://www.w3schools.com/sql/sql_intro.asp (дата звернення 11.03.2025).
9. Datatypes In SQLite – Official Docs. URL: https://sqlite.org/datatype3.html (дата звернення 22.03.2025).
10. Creating Bar Charts in Python – Matplotlib Documentation. URL: https://matplotlib.org/stable/gallery/index.html (дата звернення 09.03.2025).
11. Finmap – Фінансовий облік для підприємців. URL: https://finmap.online (дата звернення 24.03.2025).
12. Fredo – Електронний документообіг і звітність. URL: https://fredo.com.ua (дата звернення 17.03.2025).
13. Wave Accounting – Online Accounting Software. URL: https://www.waveapps.com (дата звернення 15.03.2025).
14. GitHub – Documentation for Python Projects. URL: https://docs.github.com (дата звернення 19.03.2025).

15. Stack Overflow – SQLite Join Examples. URL: https://stackoverflow.com/questions/tagged/sqlite (дата звернення 07.03.2025).

16. The Importance of Financial Reporting for SMEs – Investopedia. URL: https://www.investopedia.com/terms/f/financial-reporting.asp (дата звернення 21.03.2025).

17. Capterra – Accounting Software Reviews. URL: https://www.capterra.com/accounting-software/ (дата звернення 12.03.2025).