

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління

ГУРНЯК Сніжана Степанівна

**Веб-застосунок для керування персональними фінансами / Web Application
for Personal Finance Management**

спеціальність: 122 - Комп'ютерні науки
освітньо-професійна програма - Комп'ютерні науки

Кваліфікаційна робота

Виконала студентка групи КН-41
С.С. Гурняк

Науковий керівник
к.т.н., доцент П.Є. Биковий

Кваліфікаційну роботу допущено до
захисту

«___» _____ 20___ р.

Завідувач кафедри
_____ Н.В. Дзюбановська

ТЕРНОПІЛЬ - 2026 р.

Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління
Освітній ступінь «бакалавр»
спеціальність: 122 «Комп'ютерні науки» освітньо-професійна програма – «Комп'ютерні науки»

«ЗАТВЕРДЖУЮ»
В.о. завідувача кафедри
_____ Н.В. Дзюбановська «
_____» _____ 20__р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

ГУРНЯК Сніжані Степанівні

(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи: Веб-застосунок для керування персональними фінансами / Web Application for Personal Finance Management
 2. керівник роботи _____ Биковий П.Є., к.т.н., доц. _____,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
- затверджені наказом по університету від 01 грудня 2025 року № 894.
3. Строк подання студентом закінченої кваліфікаційної роботи 25 травня 2026р.
 4. Вихідні дані до кваліфікаційної роботи: завдання на кваліфікаційну роботу студента, наукові статті, технічна література, офіційна документація до бібліотек Flask (Python), Tailwind CSS, Chart.js, canvas-confetti та SQLite3.
 5. Основні питання, які потрібно розробити:
 - дослідити предметну область та виконати аналіз існуючих аналогів програмного забезпечення;
 - сформулювати технічні, функціональні та нефункціональні вимоги до програмної системи;
 - обґрунтувати архітектурні рішення та розробити UML-моделі системи;
 - спроектувати та реалізувати структуру реляційної бази даних;
 - розробити програмний вебзастосунок відповідно до визначених вимог;
 - реалізувати модулі візуалізації даних та побудови графіків;
 - провести тестування та аналіз отриманих результатів.
 6. Перелік графічного матеріалу у роботі:
 - діаграма активності процесу додавання фінансової операції;
 - діаграма активності створення та поповнення фінансової цілі;
 - структурна схема інформаційної системи «FinTrack»;
 - діаграма станів користувацького інтерфейсу системи.

7. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

8. Дата видачі завдання 01 грудня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів кваліфікаційної роботи	Примітка
1	Огляд літературних джерел відповідно до предметної області та складання плану роботи	до 01.01.2026 р.	
2	Написання 1 розділу кваліфікаційної роботи	до 01.02.2026 р.	
3	Написання 2 розділу кваліфікаційної роботи	до 15.03.2026 р.	
4	Написання 3 розділу кваліфікаційної роботи	до 01.05.2026 р.	
5	Представлення попереднього варіанту кваліфікаційної роботи, перевірка та внесення змін керівником	до 15.05.2026 р.	
6	Опрацювання зауважень та представлення завершеного варіанту кваліфікаційної роботи. Підготовка супроводжуючих документів	до 25.05.2026 р.	
7	Перевірка кваліфікаційної роботи на оригінальність тексту	до 30.05.2026 р.	
8	Оформлення кваліфікаційної роботи та отримання допуску до захисту	до 10.06.2026 р.	
9	Подання кваліфікаційної роботи до захисту на засіданні атестаційної комісії	до 10.06.2026 р.	

Студент _____ С.С. Гурняк
підпис

Керівник роботи _____ к.т.н., доцент П.Є. Биковий

АНОТАЦІЯ

Кваліфікаційна робота на тему «Веб-застосунок для керування персональними фінансами» на здобуття освітнього ступеня «бакалавр» зі спеціальності 122 «Комп'ютерні науки» освітньої програми «Комп'ютерні науки» написана обсягом в 74 сторінки і містить 13 ілюстрацій, 3 таблиці та 28 використаних джерел.

Метою роботи є проєктування та програмна реалізація веб-застосунку для керування персональними фінансами «FinTrack» на базі архітектурного шаблону клієнт-серверної взаємодії, що забезпечує автоматизацію контролю особистих доходів і витрат, асинхронну візуалізацію поточного балансу та інтерактивне планування накопичень.

Для розроблення застосунку застосовано системний та порівняльний аналіз предметної області, об'єктно-орієнтоване проєктування, UML-моделювання архітектури й діаграм поведінки, а також методи даталогічного моделювання реляційних баз даних.

Внаслідок виконання роботи обґрунтовано архітектурно-алгоритмічний базис системи управління капіталом та реалізовано веб-застосунок «FinTrack». Програмний продукт поєднує адаптивний Single Page Application інтерфейс на основі JavaScript та Tailwind CSS, серверну REST API логіку мовою Python на базі мікрофреймворку Flask, а також автономне реляційне сховище даних на базі СКБД SQLite, що забезпечує високу швидкодію та конфіденційність транзакцій.

Результати дослідження можуть бути використані широким колом індивідуальних користувачів та всіма охочими оптимізувати власні фінансові ресурси, незалежно від рівня їхньої технічної чи фінансової підготовки. Впровадження розробленого веб-застосунку сприяє підвищенню рівня особистої фінансової самодисципліни, раціональному розподілу бюджету та забезпечує автоматизований моніторинг прогресу досягнення фінансових цілей у реальному часі.

Ключові слова: ВЕБ-ЗАСТОСУНОК, ПЕРСОНАЛЬНІ ФІНАНСИ, FLASK, SQLITE, SINGLE PAGE APPLICATION, ФІНАНСОВІ ЦІЛІ.

ANNOTATION

The qualification paper on the topic "Web Application for Personal Finance Management" submitted for the bachelor's degree in specialty 122 "Computer Science", educational program "Computer Science", consists of 74 pages and contains 13 illustrations, 3 tables, and 28 references.

The aim of the paper is the design and software implementation of the "FinTrack" web application for personal finance management based on the client-server architecture pattern, which provides automated control of personal income and expenses, asynchronous visualization of the current balance, and interactive savings planning.

To develop the application, methods of systemic and comparative analysis of the domain area, object-oriented design, UML-modeling of architecture and behavioral diagrams, as well as datalogic modeling of relational databases were applied.

As a result of the work, the architectural and algorithmic basis of the capital management system has been substantiated, and the "FinTrack" web application has been implemented. The software product combines a responsive Single Page Application interface based on JavaScript and Tailwind CSS, server-side REST API logic written in Python using the Flask microframework, and an autonomous relational data storage based on the SQLite DBMS, which ensures high performance and transaction privacy.

The research results can be utilized by a wide range of individual users and anyone wishing to optimize their own financial resources, regardless of their technical or financial background. The deployment of the developed web application enhances personal financial self-discipline, facilitates rational budget allocation, and provides automated, real-time monitoring of progress toward achieving financial goals.

Keywords: WEB APPLICATION, PERSONAL FINANCE, FLASK, SQLITE, SINGLE PAGE APPLICATION, FINANCIAL GOALS.

ЗМІСТ

Перелік умовних позначень	7
Вступ.....	8
1 Аналіз предметної області.....	11
1.1 Опис предметної області управління персональними фінансами.....	11
1.2 Огляд і аналіз існуючих аналогів.....	13
1.3 Аналіз користувачів та їхніх цифрових звичок.....	15
1.4 Постановка задачі дослідження	16
2 Алгоритмічне та інформаційне забезпечення системи	20
2.1 Загальна структура проєктованої системи.....	20
2.2 Алгоритмічне забезпечення системи.....	21
2.3 Інформаційне забезпечення.....	25
3 Програмно-технологічне забезпечення системи.....	29
3.1 Реалізація програмного забезпечення	29
3.2 Інтерфейс користувача.....	36
3.3 Тестування розробленої програми.....	42
Висновки	44
Список використаних джерел	46
Додаток А Лістинг програмного коду клієнтської частини (index.html)	50
Додаток Б Лістинг програмного коду серверної частини (app.py)	64
Додаток В Копія опублікованих результатів	69

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БД – база даних

API – програмний інтерфейс додатка (англ. Application Programming Interface)

СКБД – система керування базами даних

ПЗ – програмне забезпечення

ІС – інформаційна система

SPA – односторінковий вебдодаток (англ. Single Page Application)

HTTP – протокол передачі гіпертексту (англ. Hypertext Transfer Protocol)

JSON – текстовий формат обміну даними (англ. JavaScript Object Notation)

REST – архітектурний стиль взаємодії компонентів (англ. Representational State Transfer)

CORS – технологія спільного використання ресурсів різних джерел (англ. Cross-Origin Resource Sharing)

HTML – стандартизована мова розмітки вебсторінок (англ. Hypertext Markup Language)

CSS – каскадні таблиці стилів (англ. Cascading Style Sheets)

JS – мова програмування ДжаваСкріпт (англ. JavaScript)

SQL – мова структурованих запитів (англ. Structured Query Language)

UML – уніфікована мова моделювання (англ. Unified Modeling Language)

DOM – об'єктна модель документа (англ. Document Object Model)

UX / UI – досвід користувача / інтерфейс користувача (англ. User Experience / User Interface)

ВСТУП

У сучасну цифрову епоху, що характеризується стрімкою трансформацією фінансових послуг та зростанням ролі мобільних технологій, ефективне управління персональними активами стало фундаментальним чинником економічної стабільності особистості. Управління власним бюджетом займає центральне місце в життєдіяльності людини, оскільки воно є необхідним для контролю споживчих витрат, мінімізації фінансових ризиків та виконання довгострокових планів накопичення [1, 2].

Історично процес управління персональними фінансами стикався з численними викликами, зокрема з низькою точністю ручного обліку, неефективністю планування та відсутністю інтегрованих систем, які дозволяють автоматизувати ці процеси в реальному часі. Завдяки появі клієнт-серверних архітектур додатка та алгоритмів обробки даних виникає можливість автоматизовано розв'язувати ці проблеми, забезпечуючи користувачів прозорим та зручним інструментарієм [3, 4].

Попри досягнення в цифровізації банківського сектору, багато традиційних підходів до ведення домашньої бухгалтерії все ще значною мірою покладаються на ручне втручання, що призводить до помилок, втрати даних та зниження мотивації користувача до ведення обліку. Технології веб-аналітики та сучасні архітектури баз даних мають потенціал докорінно перетворити ці процеси, забезпечуючи точні та ефективні рішення для класифікації та аналізу фінансових потоків [5, 6]. Однак реалізація таких систем має свої виклики, як-от забезпечення високого рівня конфіденційності даних та подолання бар'єру складності інтерфейсів для пересічного користувача [7, 8].

Актуальність роботи обґрунтована потребою в оптимізації персональних фінансових операцій, де точність обліку власних коштів та ефективність планування заощаджень залишаються постійними викликами. У цьому дослідженні розглядається потенціал веб-орієнтованих платформ для управління капіталом, пропонуючи рішення, яке покращує точність моніторингу та органічно інтегрується в щоденні цифрові звички аудиторії. Це технологічне рішення

призначене для автоматизації підрахунку балансів, використовує асинхронні REST API алгоритми обробки транзакцій для ідентифікації патернів витрат у реальному часі [9, 10]. Завдяки інтеграції модулів візуалізації та аналітики платформа має на меті запропонувати доступне та ефективне рішення, адаптоване до потреб сучасної економічно активної молоді, студентів та фрилансерів [11, 12].

Метою роботи є проєктування та програмна реалізація веб-застосунку для керування персональними фінансами «FinTrack» на базі архітектурного шаблону клієнт-серверної взаємодії, що забезпечує автоматизацію контролю особистих доходів і витрат, асинхронну візуалізацію поточного балансу та інтерактивне планування накопичень.

Для досягнення поставленої мети необхідно вирішити такі завдання:

- дослідити предметну область управління персональними фінансами та провести аналіз існуючих програмних аналогів;
- сформулювати технічні, функціональні та нефункціональні вимоги до програмної системи;
- обґрунтувати архітектурні рішення та розробити UML-моделі системи;
- спроектувати та реалізувати структуру реляційної бази даних;
- розробити програмний веб-застосунок відповідно до визначених вимог;
- реалізувати модулі візуалізації даних та побудови графіків;
- провести тестування програмного забезпечення та аналіз отриманих результатів.

Об'єктом дослідження є процес керування особистим бюджетом та накопиченнями за допомогою сучасних вебтехнологій.

Предметом дослідження є архітектурні рішення, математичні моделі та програмні інструменти розробки веб-застосунку для автоматизації контролю, прогнозування та візуалізації персональних фінансів.

Методи дослідження:

- аналіз джерел та аналогів – вивчення наукової літератури, сучасних IT-трендів та готових програмних рішень для контролю особистого бюджету;

- системний аналіз – дослідження логіки фінансових процесів користувача та чітке визначення того, які функції має виконувати майбутній застосунок;
- об'єктно-орієнтоване проєктування та UML-моделювання – розробка внутрішньої архітектури програми, побудова схем взаємодії компонентів, діаграм активності та послідовності дій;
- проєктування баз даних – створення структури таблиць у БД, їх нормалізація та налаштування зв'язків для надійного збереження інформації;
- програмна інженерія та тестування – безпосереднє написання коду та перевірка працездатності всіх модулів застосунку на практиці.

Практичне значення роботи полягає у створенні готового до впровадження веб-застосунку для керування персональними фінансами «FinTrack», який забезпечує комплексну автоматизацію контролю особистих доходів і витрат, асинхронну візуалізацію поточного балансу та інтерактивне планування накопичень. Розроблений програмний продукт може вільно використовуватися широким колом індивідуальних користувачів та всіма охочими оптимізувати власні фінансові ресурси для підвищення рівня особистої самодисципліни та раціонального розподілу бюджету в режимі реального часу.

Основні результати роботи апробовано на I Міжнародній науково-практичній конференції студентів та молодих учених «Intelligent Computing Systems and Project Management» (ICSPM 2026, м. Тернопіль), а текст тез доповіді наведено у додатку В.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Опис предметної області управління персональними фінансами

Управління персональними фінансами у сучасній науковій парадигмі розглядається як складна динамічна система, що охоплює процеси планування, накопичення, розподілу та контролю капіталу окремого індивіда або домогосподарства. У контексті цифрової трансформації економіки ця предметна область трансформувалася з пасивного спостереження за балансом рахунків у активний процес інтелектуального аналізу грошових потоків. Основний виклик у цій сфері полягає у необхідності обробки значних масивів неструктурованих даних, що генеруються внаслідок щоденної фінансової активності користувачів [1, 2].

Центральне місце в управлінні фінансами займає систематизація обліку, оскільки вона є фундаментом для забезпечення ліквідності особистих активів, мінімізації необов'язкових витрат та виконання стратегічних цілей щодо заощаджень. Історично цей процес супроводжувався низкою операційних труднощів, зокрема низкою точністю ручної фіксації транзакцій, фрагментарністю даних у різних банківських установах та відсутністю консолідованої звітності. Завдяки впровадженню алгоритмів автоматизованої класифікації та веб-технологій з'являється можливість розв'язувати ці проблеми шляхом створення єдиного інформаційного простору для фінансового контролю [4, 5].

Предметна область управління особистими коштами включає декілька критично важливих функціональних напрямів, кожен з яких вимагає специфічних інструментів автоматизації:

- моніторинг грошових потоків: цей процес передбачає реєстрацію всіх фінансових подій, включаючи активні доходи (заробітна плата, гонорари) та пасивні надходження, а також деталізований облік видатків. Важливим аспектом є фіксація не лише суми, а й метаданих транзакції: дати, категорії, методу оплати та контрагента;

- бюджетування та лімітування: встановлення фінансових рамок на певні періоди дозволяє запобігти перевищенню витрат. У межах інформаційної системи

цей процес реалізується через механізми порівняння фактичних даних із запланованими показниками у реальному часі;

- управління заощадженнями та цілепокладання: формування резервного капіталу вимагає розрахунку темпів накопичення та візуалізації прогресу щодо досягнення конкретних фінансових цілей, що є потужним мотиваційним чинником для користувача;

- аналітична звітність: перетворення сирих транзакційних даних у зрозумілі графічні форми (діаграми, графіки) дозволяє ідентифікувати патерни споживання, які раніше були приховані при ручному обліку.

Особливістю предметної області є її висока залежність від точності та оперативності даних. Навіть незначні похибки в обліку або затримки у фіксації операцій призводять до викривлення загальної фінансової картини, що робить автоматизацію не просто допоміжним, а необхідним елементом сучасного управління активами. Використання веб-орієнтованих Single Page Application (SPA) рішень забезпечує миттєвий доступ до фінансового центру з будь-якого пристрою, що відповідає мобільному способу життя сучасної цільової аудиторії [13, 14].

Окрім суто облікових задач, сучасні системи Personal Finance Management мають вирішувати питання безпеки та конфіденційності. Оскільки фінансові дані є надчутливими, архітектура системи повинна гарантувати їх цілісність та захист від несанкціонованого доступу на всіх етапах від введення до збереження у базі даних. Таким чином, розробка веб-застосунку в цій галузі вимагає збереження знань з економіки, проектування інтерфейсів (UI/UX) та веб-програмування [3, 8].

1.2 Огляд і аналіз існуючих аналогів

Попри інтенсивний розвиток сегмента фінансових технологій, більшість існуючих систем керування бюджетом характеризуються певною функціональною незбалансованістю: вони або орієнтовані на професійний сектор із надмірною кількістю інструментів, що ускладнює поріг входження для новачків, або пропонують занадто примітивні рішення без можливості глибокої аналітики. Сучасні хмарні сервіси мають потенціал змінити цей підхід, забезпечуючи продуктивний аналіз витрат у реальному часі, проте їх впровадження часто стримується питаннями безпеки та агресивною політикою монетизації [5, 14]. Для обґрунтування розробки власного програмного продукту було проведено комплексний порівняльний аналіз найбільш репрезентативних систем, представлених на світовому та локальному ринках.

Платформа Wallet by BudgetBakers є одним із лідерів у сегменті Personal Finance Management завдяки потужній системі синхронізації з банківськими рахунками через протоколи API. До переваг системи належать хмарна синхронізація даних між різними пристроями, гнучка система фінансових цілей та багаторівнева візуалізація (інтерактивні графіки та діаграми). Однак, науковий аналіз архітектури сервісу виявив суттєві недоліки: більшість критично важливих аналітичних функцій заблоковані у безкоштовній версії, а сам інтерфейс характеризується високою когнітивною складністю, що часто відштовхує користувачів, які прагнуть швидкого та простого обліку [4].

Трекер бюджету (Cleaner + Antivirus + VPN company). Цей застосунок орієнтований на швидку фіксацію транзакцій та мінімалістичний підхід до ведення бухгалтерії. Його перевагами є максимально спрощений інтерфейс та наявність базових звітів за основними категоріями витрат. Проте з точки зору системного аналізу, продукт має низьку операційну цінність для довгострокового планування: він не підтримує складні фінансові сценарії (кредити, депозити, мультивалютність), а використання сервісу супроводжується інтенсивною рекламою та нав'язуванням вбудованих покупок, що негативно впливає на користувацький досвід та концентрацію на фінансових завданнях [15].

Фінанси професіонал (Jorge de la Hoz). Рішення вирізняється високою стабільністю роботи у локальному режимі та фокусом на ручне введення даних. Система надає зручні нагадування про боргові зобов'язання та майбутні регулярні платежі, що сприяє підтримці фінансової дисципліни. Основним недоліком, що обмежує застосування цього ПЗ у сучасних умовах, є повна відсутність автоматичного імпорту транзакцій та хмарної синхронізації. Це робить базу даних вразливою до втрати при несправності пристрою та унеможливорює колективне управління бюджетом у межах домогосподарства [16].

Таблиця 1.1 узагальнює характеристики існуючих рішень за ключовими технічними та функціональними критеріями, надаючи можливість оцінити їхню ефективність у вирішенні завдань персонального бюджетування та виявити існуючі технологічні розриви.

Таблиця 1.1 – Детальна порівняльна характеристика функціональних можливостей систем-аналогів

Критерій порівняння	Wallet	Трекер бюджету	Фінанси професіонал
Архітектура бази даних	Хмарна реляційна	Гібридна	Локальна реляційна
Автоматизація імпорту	Банківська синхронізація	Сканування чеків	Відсутня
Аналітична глибина	Висока (платна)	Низька	Середня
Інтерфейс користувача	Складний / Багатофункціональний	Простий / Рекламний	Мінімалістичний
Фінансова безпека	Протоколи банків	Базова	Локальне збереження
Модель доступу	Freemium	Ad-supported	Paid App

На основі проведеного аналізу можна зробити висновок, що на ринку існує незаповнена ніша для веб-застосунку, який би поєднував безкоштовний доступ до глибокої аналітики, відсутність рекламного навантаження, автономність збереження даних та високу швидкість SPA-інтерфейсу. Проектована система

покликана розв'язати ці проблеми шляхом впровадження інтелектуальної системи цілепокладання та розрахунку індексу «фінансового здоров'я», що дозволить користувачеві не просто фіксувати витрати, а якісно змінювати свою фінансову стратегію [9, 10].

1.3 Аналіз користувачів та їхніх цифрових звичок

Актуальність розробки нової платформи інвентаризації капіталу обґрунтована специфічними потребами та цифровими звичками цільової аудиторії, до якої належать переважно представники економічно активних верств населення: студенти, фрилансери, фахівці IT-сектору та підприємці-початківці. Для зазначених груп користувачів характерна висока інтенсивність використання мобільних пристроїв та активна взаємодія з сучасними веб-застосунками у повсякденному житті, що формує запит на максимально спрощені, мінімалістичні, проте високоінформативні інтерфейси, здатні забезпечити миттєвий доступ до фінансової аналітики [13, 17].

Системний аналіз поведінкових чинників дозволив ідентифікувати основні «болі» користувачів, які перешкоджають ефективному керуванню персональним бюджетом:

- низький рівень фінансової самодисципліни: забудькуватість при фіксації дрібних витрат призводить до викривлення підсумкової статистики, що робить облік нерелевантним;
- технологічний бар'єр: надмірна складність наявних програмних продуктів та необхідність тривалого налаштування відштовхують користувачів на початкових етапах;
- відсутність візуальної мотивації: сухі цифри звітів не стимулюють людину заощаджувати, що вимагає інтеграції елементів гейміфікації процесу накопичень.

Мотиваційним ядром для використання автоматизованої системи є прагнення до повного контролю над грошовими ресурсами, що, згідно з

психологічними дослідженнями, безпосередньо корелює зі зниженням рівня загального стресу та підвищенням фінансової відповідальності особистості. Платформа розробляється як комплексна відповідь на ці виклики. Вона пропонує користувачеві не лише зручний контроль транзакцій у декілька кліків, динамічні графіки та гнучку систему фінансових цілей з інтерактивними шкалами виконання, а й інструменти аналітичного прогнозування бюджету разом із системою розумних рекомендацій для оптимізації витрат у реальному часі.

Важливою особливістю проєкту є впровадження інструментів візуалізації прогресу, що стимулює накопичення коштів через гейміфікацію процесу заощаджень [18, 14]. Аналіз цифрових звичок аудиторії демонструє схильність користувачів до отримання стислих, візуально зрозумілих звітів, тому в системі реалізовано адаптивні дашборди. Окрім того, враховуючи переважне користування месенджерами, архітектура платформи передбачає можливість подальшої інтеграції засобів сповіщень, що відповідає сучасним стандартам UX-дизайну та забезпечує безперервність фінансового моніторингу [19, 20].

Таким чином, орієнтація на специфічні потреби цільової аудиторії дозволяє не лише створити технологічно досконалий продукт, а й сформувати стійку звичку раціонального споживання, що має вагоме соціально-економічне значення в умовах сучасного ринку.

1.4 Постановка задачі дослідження

Враховуючи результати проведеного аналізу предметної області та ідентифіковані недоліки існуючих програмних рішень, формулюється завдання розробки спеціалізованого веб-застосунку. Основна концепція проєкту полягає у створенні платформи, яка б органічно поєднувала доступний безкоштовний функціонал із професійними інструментами аналітики, що раніше були притаманні лише платним аналогам. Процес розробки охоплює декілька критично важливих етапів: від наукового обґрунтування та вибору системної архітектури, що гарантує стабільність під високим навантаженням, до впровадження багаторівневих алгоритмів захисту персональних даних [7, 21].

Проектована платформа має забезпечувати аналіз транзакційних потоків у реальному часі, що дозволить реалізувати механізми розпізнавання патернів споживання та оперативну корекцію бюджетних відхилень. Це особливо актуально в умовах нестабільного надходження доходів, оскільки система дозволяє динамічно адаптувати фінансову стратегію користувача. Завдяки інтеграції з сучасними бібліотеками візуалізації, розроблюване рішення позиціонується не лише як ефективний операційний засіб обліку, а й як доступний інструмент для глибокого самоаналізу фінансового стану індивіда.

В основі функціонування системи лежить автоматизація ключових бізнес-процесів предметної області, які деталізовано нижче:

- контроль поточних активів: реалізація механізму фіксації доходів із обов'язковим розподілом за категоріями для забезпечення прозорості структури капіталу;
- категоризація та лімітування видатків: впровадження логіки групування витрат за їх функціональним призначенням (Продукти, Кафе, Здоров'я тощо), а також встановлення персональних фінансових лімітів для кожної категорії з метою уникнення перевитрат;
- інтелектуальне прогнозування та аналітика: автоматичний розрахунок очікуваних витрат до кінця поточного місяця та генерація персоналізованих рекомендацій на основі цифрових звичок користувача для оптимізації його бюджету;
- керування стратегічними цілями: розробка інструментарію для створення, поповнення та моніторингу фінансових цілей, що автоматично генерує супутні транзакції для підтримки балансової цілісності системи;
- графічний аналіз та макрозвітність: інтеграція засобів автоматичної побудови кругових, лінійних та згрупованих стовпчикових діаграм для миттєвої оцінки поточного стану фінансів та формування річної звітності в реальному часі.

На основі проведеного аналізу встановлено, що розроблювана система повинна відповідати наступному переліку технічних вимог:

- функціональні вимоги: реалізація адаптивного SPA-інтерфейсу з трьома інтерактивними вкладками («Огляд», «Аналітика», «Цілі») та модальним оверлеєм річного звіту, забезпечення швидкої фіксації транзакцій, ведення динамічного журналу історії операцій з асинхронним видаленням записів, а також створення модуля планування фінансових цілей із візуальними прогрес-барами та елементами гейміфікації;

- аналітичні вимоги: автоматичне агрегування даних за допомогою SQL-запитів до СКБД для відображення ключових показників ефективності загального балансу, усього доходів та усього витрат, побудова кругової діаграми розподілу витрат за категоріями засобами HTML5 <canvas>, хронологічного лінійного графіка динаміки зміни капіталу, а також генерація тексту персоналізованих рекомендацій від FinTrack та побудова стовпчикової діаграми річного звіту;

- нефункціональні вимоги: побудова системи на базі клієнт-серверного шаблону взаємодії (Flask REST API — JavaScript), використання автономного сховища даних SQLite (локальний файл finance.db), висока швидкість відгуку інтерфейсу на асинхронні запити за допомогою Fetch API без перезавантаження сторінок, повна адаптивність UI на основі фреймворку Tailwind CSS, а також забезпечення конфіденційності транзакцій завдяки локальному збереженню інформації.

Таким чином, комплексна реалізація зазначених вимог дозволить створити конкурентоспроможний програмний продукт, спрямований на підвищення особистої фінансової самодисципліни та раціональний розподіл бюджету користувача.

Основні бізнес-процеси предметної області:

- облік поточних активів: фіксація доходів із розподілом за джерелами;
- категоризація та лімітування видатків: групування витрат за функціональним призначенням та встановлення індивідуальних планових лімітів для запобігання перевитратам;

- інтелектуальне прогнозування та аналітика: автоматичний розрахунок фінансових ризиків до кінця календарного місяця та генерація персоналізованих підказок щодо оптимізації бюджету;
- планування накопичень: створення, поповнення та моніторинг стратегічних цілей із автоматичним підтриманням балансової цілісності системи;
- графічний аналіз та макрозвітність: асинхронне агрегування показників капіталу з миттєвою візуалізацією структури споживання за місяць та формуванням детального річного звіту.

2 АЛГОРИТМІЧНЕ ТА ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

2.1 Загальна структура проєктованої системи

Проєктована система керування персональними фінансами розробляється як сучасний веб-орієнтований застосунок, побудований на принципах багаторівневої архітектури. Такий підхід дозволяє забезпечити чітке розділення зон відповідальності між компонентами, що є критично важливим для стабільної роботи фінансового ПЗ. Вибір даної архітектурної моделі зумовлений потребою у високій масштабованості системи, спрощенні процесів тестування та легкості впровадження нових функціональних модулів без порушення цілісності існуючого коду [6, 21].

В основі архітектури лежить принцип модульності, де кожен рівень виконує специфічний набір задач, взаємодіючи з іншими через стандартизовані протоколи передачі даних. Це дозволяє оптимізувати навантаження на серверні потужності та забезпечити високу швидкість відгуку інтерфейсу користувача.

Основними складовими частинами архітектури системи є:

- Frontend-модуль: цей компонент є точкою безпосередньої взаємодії з користувачем. Він реалізує графічний інтерфейс засобами HTML5 та Tailwind CSS, забезпечує динамічну валідацію форм введення даних на стороні клієнта за допомогою JavaScript для мінімізації помилкових запитів до сервера, здійснює асинхронне виведення текстових рекомендацій та рендеринг інтерактивних графіків за допомогою бібліотеки Chart.js. Використання сучасних технологій дозволяє реалізувати механізм SPA, що забезпечує плавну навігацію та миттєве оновлення аналітичних дашбордів, модулів лімітів та вікон річної звітності без повного перезавантаження сторінок [21, 23];

- API-сервер: центральний вузол системи, який опрацьовує всю обчислювальну складову застосунку. Серверний модуль на базі мікрофреймворку Flask (Python) відповідає за виконання математичних алгоритмів розрахунку балансів, оновлення статусів фінансових цілей, а також роботу евристичних алгоритмів прогнозування витрат і генерації персоналізованих фінансових порад.

Взаємодія з клієнтською частиною відбувається через асинхронні REST API ендпоінти, що гарантує високу швидкість обміну даними у форматі JSON [24];

- Сховище даних: інформаційне ядро системи, реалізоване у вигляді реляційної структури на базі СКБД SQLite. Вибір цього рішення обґрунтований його автономністю, високою швидкодією та надійністю для локальних веб-застосунків. Сховище забезпечує логічну зв'язність даних між транзакціями, лімітами та цілями накопичення, гарантуючи цілісність інформації навіть при виконанні складних аналітичних вибірок [25].

Проектування за такою схемою дозволяє веб-застосунку ефективно обробляти транзакційні потоки, зберігаючи при цьому високу гнучкість інтерфейсу, інтелектуальну складову аналізу та надійність збереження конфіденційної інформації користувача. У наступних підрозділах буде детально розглянуто алгоритмічні рішення та даталогічну модель, що реалізують описану структуру.

2.2 Алгоритмічне забезпечення системи

В основі роботи платформи лежать алгоритми, які автоматизують процеси контролю фінансів, гарантують точність обчислень та забезпечують швидкий інтерактивний відгук інтерфейсу. Важливою частиною архітектури є наскрізна валідація даних на кожному етапі взаємодії — це повністю виключає потрапляння некоректної чи суперечливої інформації до бази даних [25-28].

2.2.1 Алгоритми функціональних модулів

Нижче представлено алгоритмічний опис процесів функціонування системи при виконанні базових фінансових сценаріїв. Завдяки використанню діаграм активності формалізовано динаміку системних станів та регламентовано умови переходів, що дозволяє детально простежити логіку опрацювання інформаційних потоків від моменту ініціації запиту користувачем до отримання кінцевого аналітичного результату.

Алгоритм наскрізного контролю та аналізу бюджету — логіка процесу починається із заповнення форми користувачем та натискання кнопки «Додати».

Після цього запускається етап вхідної валідації для перевірки полів. Якщо введені дані не є валідними, система виводить помилку на екрані та повертає користувача до коригування форми. У разі успішної валідації запускається ланцюжок серверних операцій: реєстрація та модифікація загального балансу, розрахунок показників в аналітичному контурі лімітів для поточної категорії, а також фінальне формування інтелектуального висновку з персоналізованими порадами (рисунок 2.1).

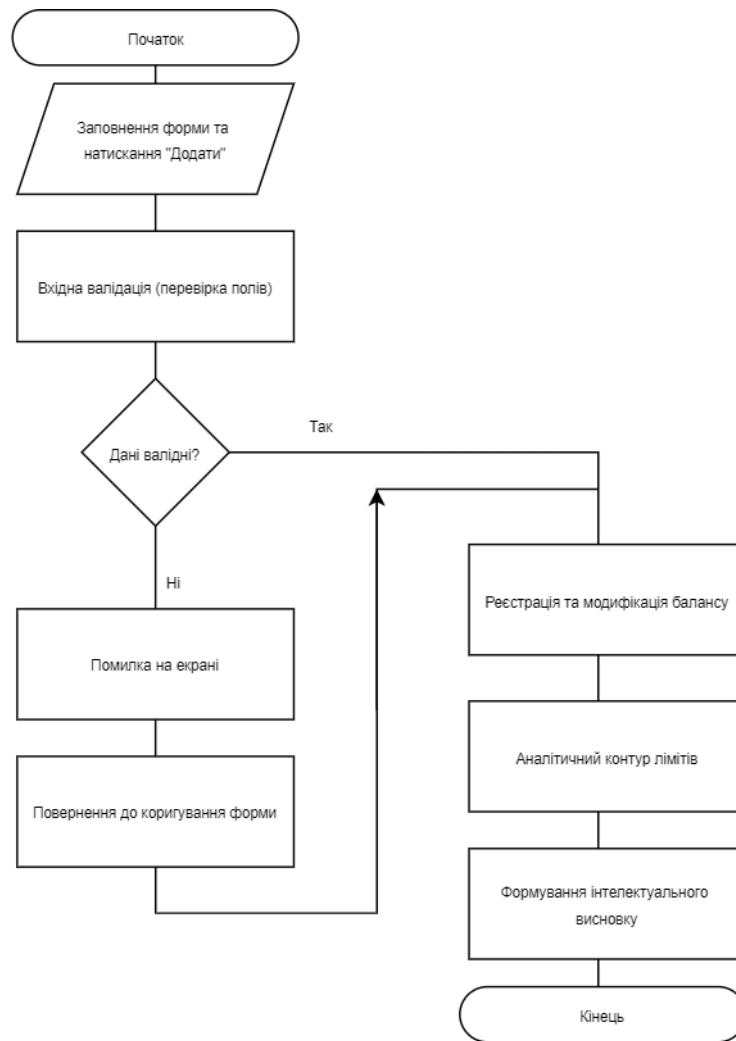


Рисунок 2.1 – Алгоритм процесу обробки та аналізу фінансової операції.

Логіка керування фінансовими цілями – алгоритм забезпечує роботу інструментарію стратегічного планування накопичень. На початковому етапі користувач створює ціль, вказуючи її назву та необхідну суму, після чого система виконує ініціалізацію об'єкта в базі даних із нульовим балансом поточних накопичень ($current = 0$). При подальшому поповненні цілі через користувацький інтерфейс програма виконує автоматичний перерахунок суми поточного прогресу та перевіряє логічну умову досягнення мети ($current \geq target?$). Якщо цільову суму

накопичено, система активує візуальні ефекти гейміфікації для заохочення користувача, в іншому випадку алгоритм одразу завершує свою роботу (рисунок 2.2).

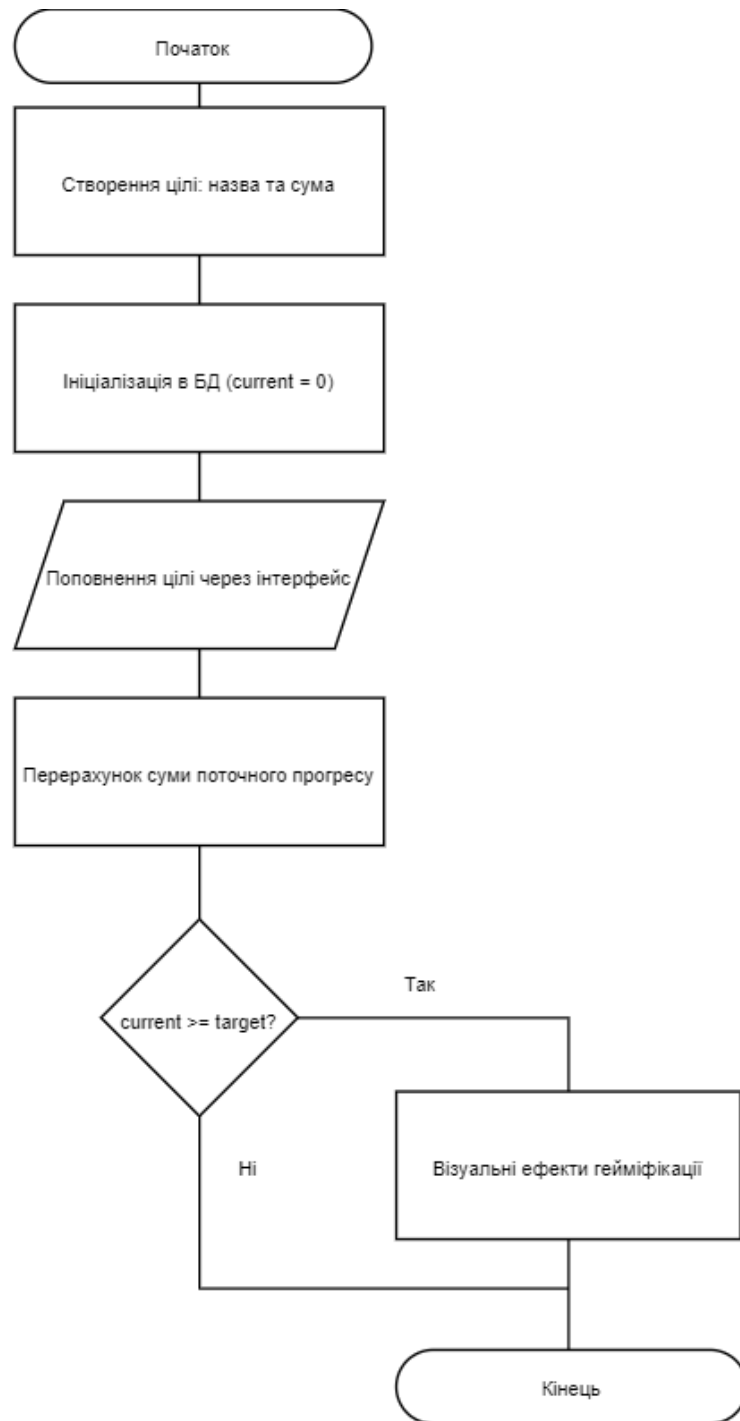


Рисунок 2.2 – Алгоритм створення та поповнення фінансової цілі

2.2.2 Алгоритми взаємодії компонентів

Для розуміння динаміки роботи системи важливо розглянути алгоритм взаємодії між клієнтською частиною (Web UI), сервером Flask та базою даних

SQLite під час реєстрації фінансових подій, що відображено на діаграмі послідовності (рисунок 2.3).

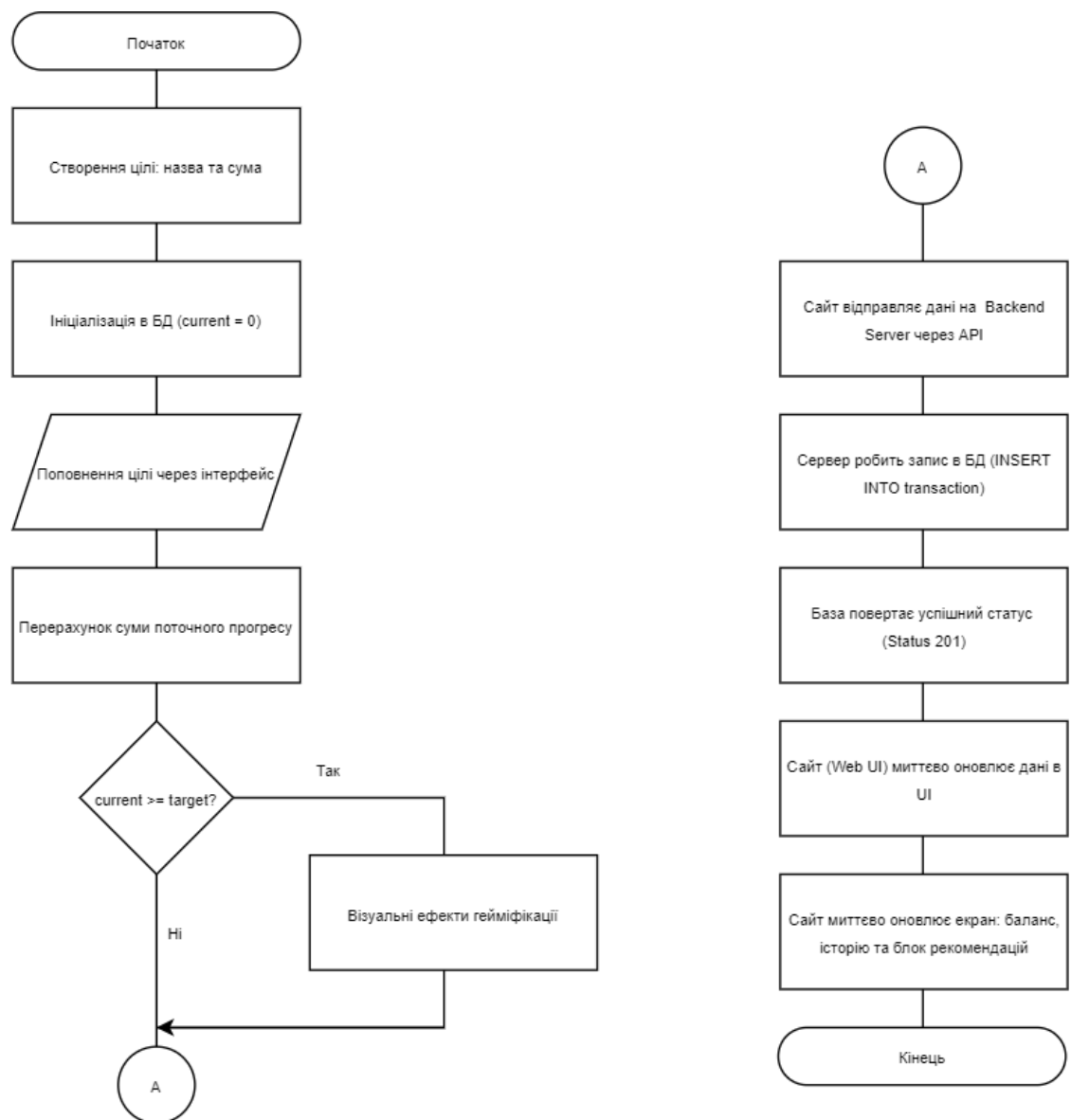


Рисунок 2.3 - Алгоритм послідовності взаємодії компонентів системи

Взаємодія компонентів та наскрізна обробка інформаційних потоків реалізується через наступні послідовні стадії:

- ініціація та збір даних: користувач на екрані «Огляд» вводить дані (тип, категорію, суму) та натискає «Додати». Сайт (Web UI) зчитує суму: якщо Сума $\leq$ 0, виводиться помилка «Заповніть поле!» і користувач залишається на тому ж екрані;
- обробка та запис: у разі коректності суми сайт відправляє дані на Backend Server через API. Сервер виконує запис у базу даних за допомогою команди INSERT INTO transaction;

- підтвердження: після фіксації даних у файлі сховища база даних повертає успішний статус (Status 201) на сервер, а сервер на клієнтську частину;
- оновлення інтерфейсу: JavaScript-модуль сайту перехоплює відповідь і миттєво, без перезавантаження сторінки, оновлює дані в UI: актуальний баланс, журнал історії та блок персоналізованих рекомендацій.

2.2.3 Математичне забезпечення обчислення прогресу цілей

Окремим інтеграційним модулем алгоритмічного забезпечення є логіка моніторингу та калькуляції фінансових цілей. Розрахунок відсоткового прогресу виконання для кожної цілі здійснюється динамічно на стороні клієнта на основі отриманих від API числових значень за наступною математичною формулою:

$$P = \left(\frac{S_{current}}{S_{target}} \right) \cdot 100\% , \quad (2.1)$$

де P – відсоток виконання;

$S_{current}$ – поточна накопичена сума;

S_{target} – цільова сума.

Цей алгоритм працює в синергії з логікою авто-витрат: у момент фінансування цілі бекенд автоматично створює супутню транзакцію із типом витрата та категорією Кошти на ціль, що дозволяє підтримувати загальну балансову стійкість системи та забезпечує високу точність фінансового моніторингу.

2.3 Інформаційне забезпечення

Інформаційне забезпечення веб-застосунку становить сукупність структурованих масивів даних, методів їх збереження та маніпулювання ними. Основою цього рівня є реляційна база даних, спроектована таким чином, щоб

мінімізувати надлишковість інформації та забезпечити високу швидкість виконання аналітичних запитів [4, 19].

2.3.1 Логічна модель бази даних

Логічна структура бази даних веб-застосунку «FinTrack» є максимально оптимізованою, компактною та повністю автономною. Сховище спроектовано за принципами нормалізації (до третьої нормальної форми включно), що дозволяє уникнути аномалій при оновленні чи видаленні записів. Фізична реалізація виконана у вигляді трьох взаємопов'язаних реляційних таблиць у межах єдиного локального файлу сховища `finance.db` (рисунок 2.4).

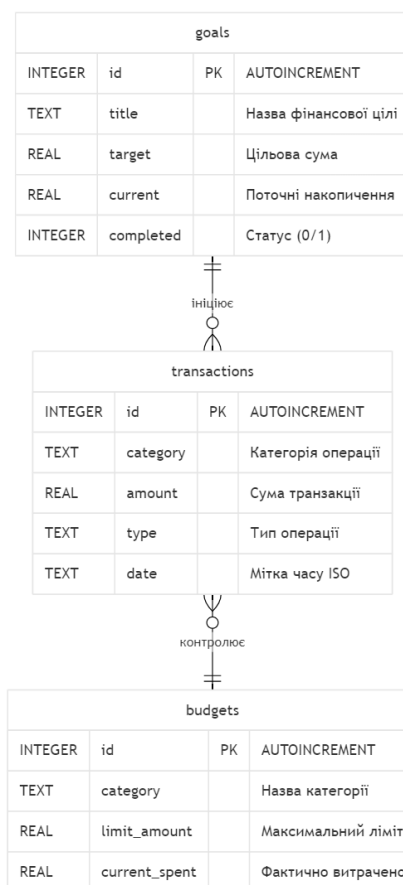


Рисунок 2.4 - Схема структури реляційної бази даних веб-застосунку

Основними компонентами структури є:

- сутність `transactions`: центральна таблиця, що фіксує кожну фінансову подію. Вона містить унікальний ідентифікатор `id` (Primary Key), текстове поле `category` для групування витрат і доходів, числовий атрибут `amount` для збереження

суми операції, маркер типу type (дохід/витрата) та текстове поле date для збереження часової мітки у форматі ISO;

- сутність goals: компонент системи планування. Зберігає назву цілі title, фінансові орієнтири target, поточний рівень заощаджень current, а також логічний прапорець completed (0 або 1), який виступає в ролі тригера для активації інтерфейсних скриптів заохочення;

- сутність budgets: інструмент аналітичного контуру програми, що відповідає за калькулятор лімітів та прогнозування бюджету. Таблиця містить унікальний ключ id, назву категорії category, ліміт максимальних витрат limit_amount, а також розрахункове поле current_spent для асинхронного порівняння фактичних видатків із встановленими обмеженнями та генерації інтелектуальних рекомендацій.

2.3.2 Забезпечення цілісності та безпеки даних

Для гарантування надійності інформаційного забезпечення на рівні локальної бази даних SQLite впроваджено наступні механізми:

- транзакційність (ACID): кожна операція зміни балансу, внесення коштів на ціль або встановлення лімітів виконується в межах єдиної транзакційної сесії додатка. Якщо запис транзакції не вдасться зберегти на рівні файлової системи, загальний баланс дашборду не буде змінено, що виключає ризик розсинхронізації чи пошкодження даних;

- типізація даних: для запобігання похибкам округлення грошових сум, характерних для динамічних мов програмування, всі числові поля фінансових показників (amount, target, current, limit_amount, current_spent) на рівні сховища суворо обмежені реляційним типом REAL;

- індексація: для первинних ключів id за замовчуванням активовано індексацію B-Tree, що дозволяє веб-застосунку миттєво здійснювати складні SQL-вибірки історії операцій, асинхронно агрегувати суми для звітних періодів аналітики та розраховувати прогнози в реальному часі.

Для деталізації фізичного рівня архітектури даних та забезпечення однозначності інтерпретації структурних елементів системи було розроблено специфікацію ключових сутностей. У таблиці 2.1 наведено розширений опис атрибутів основних таблиць бази даних із зазначенням відповідних типів даних та їхнього функціонального призначення в межах розробленого веб-застосунку.

Таблиця 2.1 — Опис ключових полів бази даних та їх типів

Назва таблиці	Поле	Тип даних	Опис / Функція
transactions	id	INTEGER	Унікальний номер транзакції
transactions	category	TEXT	Назва фінансової категорії для групування
transactions	amount	REAL	Сума фінансової операції
transactions	type	TEXT	Класифікатор операції (дохід / витрата)
transactions	date	TEXT	Часова мітка створення запису
goals	id	INTEGER	Унікальний номер фінансової цілі
goals	target	REAL	Цільова сума накопичення
goals	completed	INTEGER	Статус досягнення цілі (0 – в процесі, 1 – виконано)

3 ПРОГРАМНО-ТЕХНОЛОГІЧНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

3.1 Реалізація програмного забезпечення

3.1.1 Структурна схема програмної системи та її опис

Розроблювана інформаційна система «FinTrack» спроектована та реалізована на основі класичного архітектурного шаблону клієнт-серверної взаємодії із чітким розділенням системних обов'язків. Ключовою технологічною особливістю створеного програмного комплексу є застосування модульного SPA підходу на стороні клієнта у поєднанні з легковаговою сервіс-орієнтованою архітектурою на стороні сервера. Таке інженерне рішення дозволяє мінімізувати навантаження на мережеві канали зв'язку, забезпечує високу швидкість відгуку інтерфейсу користувача та ізолює логіку збереження даних від механізмів їх візуального представлення.

Відповідно до принципів багат шарової архітектури програмного забезпечення, загальну структурну схему інформаційної системи «FinTrack» декомпоновано на три автономні, логічно відокремлені, але інформаційно інтегровані рівні, кожен з яких виконує строго визначений набір функцій.

Рівень представлення або клієнтська частина функціонує безпосередньо в середовищі веб-браузера кінцевого користувача і є повністю відповідальним за формування графічного інтерфейсу, валідацію первинних дій та забезпечення інтерактивної взаємодії. Технологічний стек цього рівня включає:

- HTML5: використовується для побудови семантичної структури веб-документа, розмітки інформаційних карток та елементів керування;
- Tailwind CSS v4: утилітарний CSS-фреймворк нового покоління, який забезпечує гнучку, адаптивну та продуктивну стилізацію інтерфейсу. Завдяки вбудованій системі констант на рівні представлення реалізовано преміальну мінімалістичну палітру кольорів із точним позиціонуванням компонентів сітки, що забезпечує сучасний зовнішній вигляд застосунку. Використання фреймворку сприяє скороченню обсягу CSS-коду, пришвидшує процес розробки та гарантує коректне відображення інтерфейсу на різних пристроях;

- Lucide Icons: бібліотека векторних іконок, інтегрована для забезпечення інтуїтивно зрозумілої візуальної навігації (маркування типів транзакцій, фінансових цілей тощо);
- Chart.js: спеціалізована JavaScript-бібліотека, що використовує HTML5-елемент `<canvas>` для побудови динамічних графіків на льоту. На її основі функціонують компоненти кругової діаграми розподілу витрат (Doughnut Chart) та ретроспективного лінійного графіка капіталу (Line Chart) та стовпчикової діаграми порівняльного аналізу місяців у модальному вікні річного макрозвіту.

Головне завдання, перехоплення подій користувача (наприклад, клік на кнопку «Додати запис», перемикання вкладок або встановлення планових бюджетних обмежень), динамічна зміна станів DOM-дерева без перезавантаження сторінки, асинхронне виведення текстових порад від системи та ініціалізація асинхронних запитів до серверної частини. Повний лістинг вихідного коду структури та клієнтської логіки цього рівня наведено у додатку А.

Рівень бізнес-логіки або серверна частина розгорнута на базі високорівневої мови програмування Python та мікрофреймворку Flask. Він виступає в ролі центрального комунікаційного та обчислювального хаба системи. Flask функціонує як REST API сервер, який ізольований від прямого формування HTML-структур, що підвищує гнучкість системи. Основні функції включають:

- маршрутизація: обробка вхідних HTTP-запитів за строго визначеними ендпоінтами (`/api/transactions`, `/api/goals` тощо);
- обробка та валідація даних: сервер зчитує передані клієнтом дані, здійснює їх перетворення (наприклад, приведення рядкових значень сум до типу `float`) та перевіряє коректність параметрів перед відправкою у сховище;
- реалізація бізнес-правил: саме на цьому рівні виконується логіка фінансового обліку. Наприклад, при поповненні цілі контролер Flask не просто оновлює її поточний стан, а автоматично ініціює створення супутньої транзакції витрати, підтримуючи цим балансову цілісність системи. Паралельно з цим серверний рівень виконує математичний прорахунок прогнозних ризиків перевищення лімітів до кінця календарного місяця та запускає евристичний модуль

генерації індивідуальних текстових рекомендацій для оптимізації витрат. Програмна реалізація серверних REST API маршрутизаторів мовою Python представлена у додатку Б.

Рівень даних або сховище відповідає за довготривале персистентне зберігання фінансової інформації, журналів транзакцій, параметрів щомісячних лімітів та параметрів планування. Для оптимізації швидкодії та забезпечення автономності додатка використано безсерверну реляційну систему керування базами даних SQLite3. База даних локалізована в єдиному файлі `finance.db`, що знаходиться безпосередньо в кореневому каталозі проєкту.

Взаємодія сервера з СКБД реалізована за допомогою низькорівневих оптимізованих SQL-запитів. Для забезпечення гнучкості маніпулювання даними в Python-коді активовано режим `conn.row_factory = sqlite3.Row`, який трансформує стандартні кортежі реляційних рядків у структуровані словники, що значно прискорює їх подальшу серіалізацію.

Опис механізму міжрівневої взаємодії: Комунікація між Рівнем представлення та Рівнем бізнес-логіки реалізована за технологією AJAX (Asynchronous JavaScript and XML) із використанням сучасного JavaScript інструменту Fetch API. Процес передачі інформації відбувається повністю асинхронно у фоновому режимі, що дозволяє користувачеві безперешкодно взаємодіяти з інтерфейсом під час виконання запитів до бази даних (рисунк 3.1).

Стандартизованим форматом обміну даними є JSON (JavaScript Object Notation). Наприклад, коли користувач вносить нову операцію:

- 1) клієнтська функція `sendData()` збирає значення з полів форми, пакує їх у JSON-об'єкт та надсилає HTTP POST-запит на сервер;

- 2) Flask-сервер приймає цей об'єкт за допомогою методу `request.get_json()`, обробляє його, відкриває транзакційну сесію з файлом `finance.db` та виконує SQL-команду `INSERT INTO transactions`;

- 3) після успішного збереження даних у СКБД, сервер ініціює перерахунок лімітів, формує новий прогноз із блоком рекомендацій та повертає статус відповіді у форматі JSON;

4) клієнтський скрипт отримує цю відповідь, автоматично викликає функцію оновлення інтерфейсу `fetchDashboard()`, яка робить повторний GET-запит, забирає оновлені агреговані суми актуальний текст порад від FinTrack і моментально перемальовує графіки Chart.js, картки KPI та індикатори шкал виконання лімітів на екрані користувача.

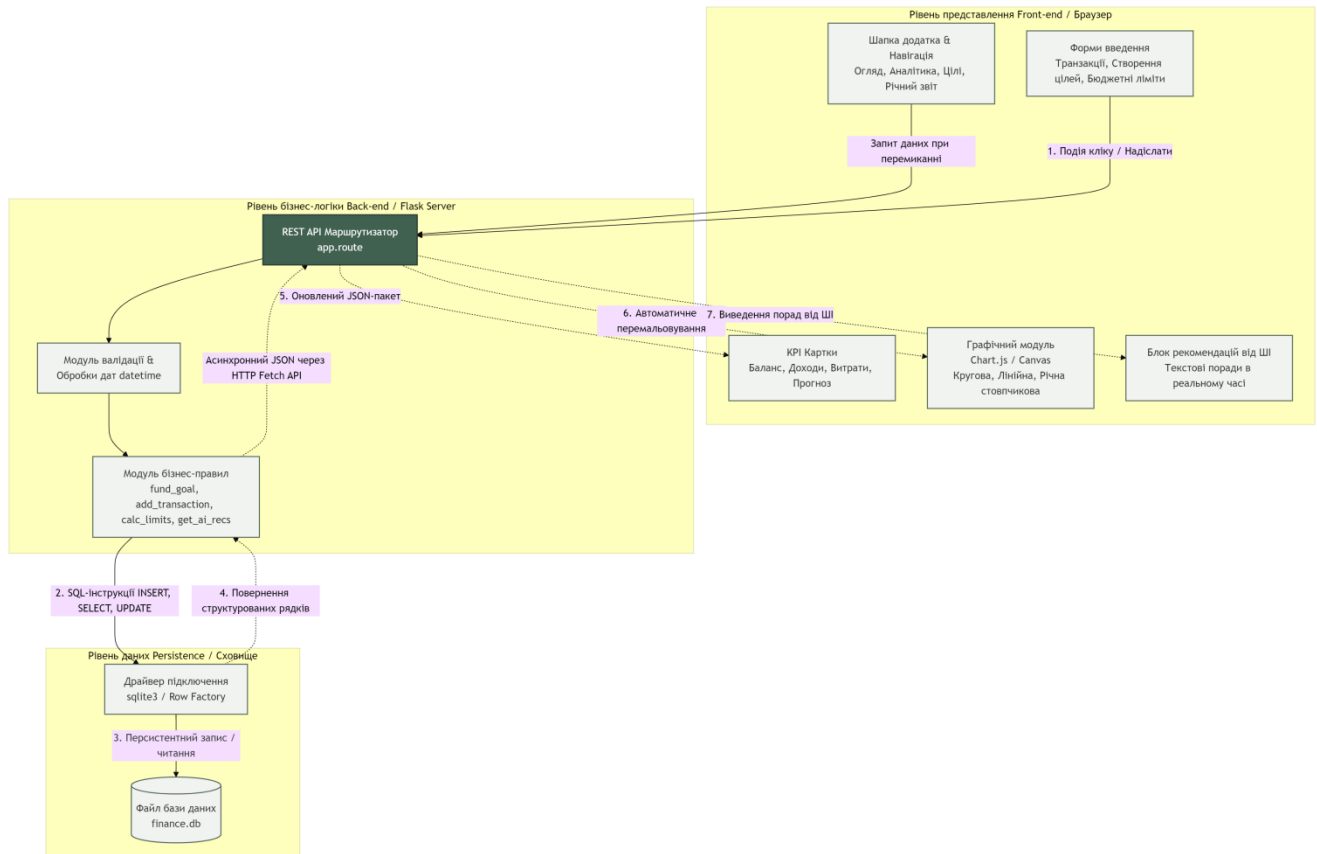


Рисунок 3.1 – Структурна схема інформаційної системи «FinTrack»

3.1.2 UML-діаграма класів, що реалізують основну бізнес-логіку

Оскільки обраний стек технологій Flask у поєднанні з драйвером `sqlite3` використовує процедурно-модульний підхід до проектування REST API, архітектура серверної частини відображається через взаємодію контролерів та персистентних сутностей бази даних. Логічна структура таблиць і функцій, які забезпечують життєвий цикл транзакцій та фінансових цілей, представлена на діаграмі класів (рисунок 3.2).

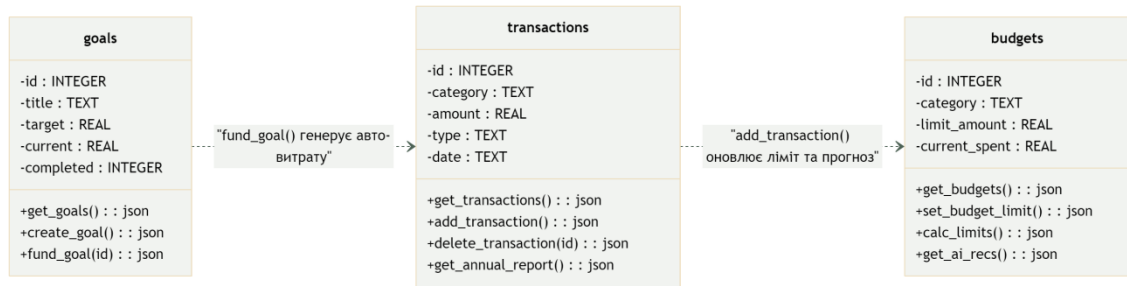


Рисунок 3.2 – UML-діаграма сутностей та REST API контролерів системи

Опис функціонального призначення основних методів бізнес-логіки сервера (app.py):

- `init_db()` – системна функція первинного розгортання інформаційного сховища. Вона перевіряє наявність локального файлу `finance.db` та асинхронно ініціює створення реляційних таблиць `transactions`, `goals` та `budgets` із суворим дотриманням нативної типізації даних (зокрема, використання типу `REAL` для забезпечення високої точності фінансових обчислень);

- `add_transaction()` – контролер обробки HTTP POST-запитів, що відповідає за реєстрацію фінансових операцій. Метод валідує вхідні параметри, зчитує системний час сервера за допомогою бібліотеки `datetime` та формує текстовий штамп часової мітки у форматі ISO перед безпосереднім записом у базу даних. Паралельно функція запускає тригери перерахунку лімітів;

- `fund_goal(id)` – модуль управління накопиченнями на фінансові цілі. Він приймає суму поповнення, додає її до атрибута `current` та перевіряє логічну умову досягнення мети (якщо `current >= target`, статус `completed` змінюється на 1). Для забезпечення балансової цілісності системи метод автоматично генерує та записує супутню транзакцію витрати із категорією «Кошти на ціль»;

- `calc_limits()` – аналітичний модуль калькулятора лімітів. Функція динамічно агрегує суми витрат за кожною категорією в таблиці `transactions` за поточний звітний період, порівнює їх із встановленими обмеженнями в таблиці `budgets`, оновлює розрахункове поле `current_spent` та формує математичний прогноз динаміки витрат до кінця календарного місяця;

- `get_ai_recs()` – евристичний модуль генерації інтелектуальних рекомендацій від системи. На основі розрахованих показників виконання

бюджетних обмежень та прогнозних ризиків перевитрат, алгоритм у реальному часі формує персоналізовану текстову пораду для виведення на дашборд користувача;

– `get_annual_report()` – контролер агрегації історичних даних для формування річного макрозвіту. Метод виконує складну SQL-вибірку за останні 12 місяців, групує доходи й витрати та повертає структурований масив даних у форматі JSON для подальшої візуалізації у вигляді стовпчикової діаграми засобами бібліотеки `Chart.js`.

Повний синтаксис та взаємодія зазначених функцій у межах файлу сервера `app.py` міститься у лістингу додатку Б.

3.1.3 UML-діаграма станів елементів графічного інтерфейсу

Користувацький інтерфейс побудовано як SPA, де перемикання між функціональними модулями («Огляд», «Аналітика», «Цілі») здійснюється через перехоплення подій кліків за допомогою JavaScript та маніпуляцію властивістю `classList.remove('hidden') / classList.add('hidden')` (рисунк 3.3).

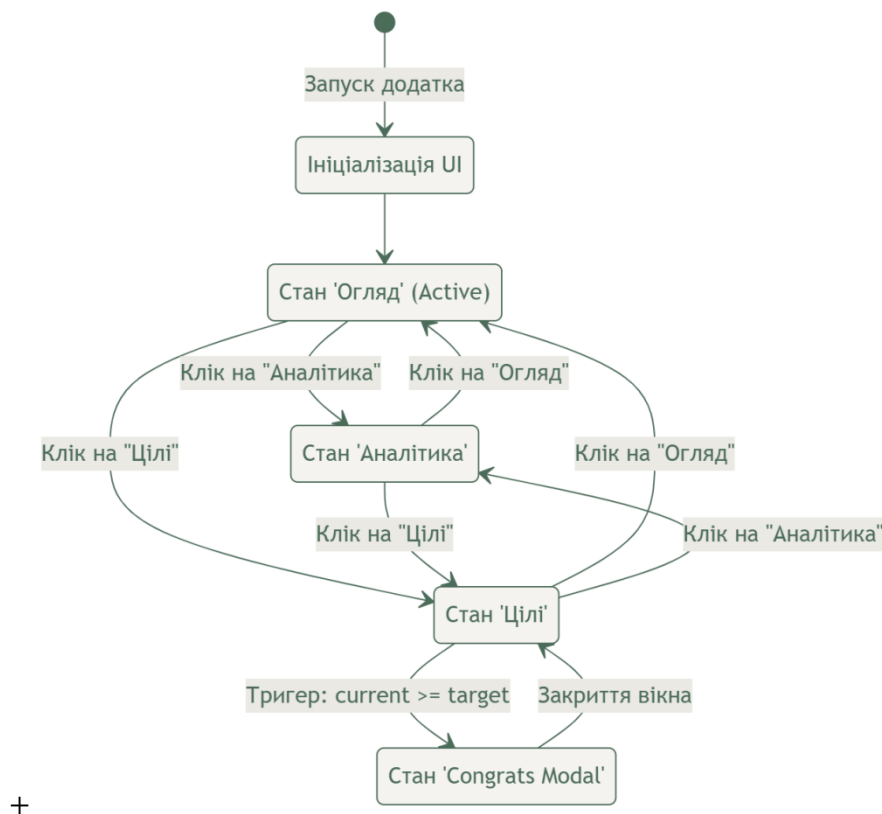


Рисунок 3.3 – Схема станів користувацького інтерфейсу системи

3.1.4 Обґрунтування вибору програмних засобів реалізації

Вибір технологічного стеку для розробки системи «FinTrack» обґрунтовано критеріями кросплатформності, швидкодії та незалежності від сторонніх важких серверних служб:

- Python та Flask: забезпечують розробку надійної серверної логіки. Flask обрано за його мікро-архітектуру, що дозволяє створювати чисті REST API ендпоінти без генерації зайвих системних файлів та накладних витрат продуктивності. Окрім цього, багата екосистема мови Python дозволила безшовно інтегрувати математичні методи прогнозування капіталу та евристичні алгоритми аналізу фінансових ризиків безпосередньо в ядро бекенду;
- SQLite: реляційна база даних, яка не потребує встановлення окремого сервера чи налаштування фіксованих портів, що повністю виключає помилки зв'язку та таймаутів при розгортанні на будь-якому комп'ютері. Варто зазначити, що початкова концепція системи передбачала використання СКБД PostgreSQL, проте в ході оптимізації архітектури задля забезпечення повної автономності та спрощення локального розгортання застосунку було прийнято рішення про перехід на автономну СКБД SQLite3;
- Tailwind CSS v4: дозволив створити сучасну палітру кольорів (#4C6C5A — акцентний зелений, #BC7C6C — пастельно-червоний витрат, #F4F3EF — колір карток), забезпечуючи високу швидкість рендерингу та адаптивність;
- Chart.js: використовується для побудови інтерактивних діаграм через HTML5 елемент <canvas>. Вона забезпечує високу продуктивність при фоновому перемальовуванні кругових діаграм розподілу категорій, лінійних графіків балансу та стовпчикових діаграм порівняльних зрізів у модулі річного макрозвіту;
- Canvas-confetti: бібліотека гейміфікації для стимулювання користувача при досягненні фінансових цілей.

Практична синергія обраного технологічного стеку та асинхронна взаємодія між компонентами системи детально відображена у вихідних кодах розробленого веб-застосунку (додаток А, додаток Б).

3.2 Інтерфейс користувача

Розроблений інтерфейс користувача системи «FinTrack» є повністю адаптивним та орієнтованим на сучасні стандарти UI/UX дизайну.

У верхній частині кожної сторінки розташована статична шапка (Header) із логотипом, назвою системи та девізом «Ваш естетичний фінансовий помічник». У центрі шапки розміщено таб-перемикач навігаційних кнопок («Огляд», «Аналітика», «Цілі»). Справа від них інтегровано акцентну кнопку «Звіт за рік» з аналітичною графічною іконкою, яка слугує для виклику модуля макроаналізу капіталу. Поруч розташовано кнопку швидкого видалення (іконка кошика) для повного обнулення локальної бази даних. Під шапкою розміщено три блоки головних фінансових індикаторів: «Поточний баланс», «Доходи місяця» та «Витрати місяця» (рисунок 3.4).

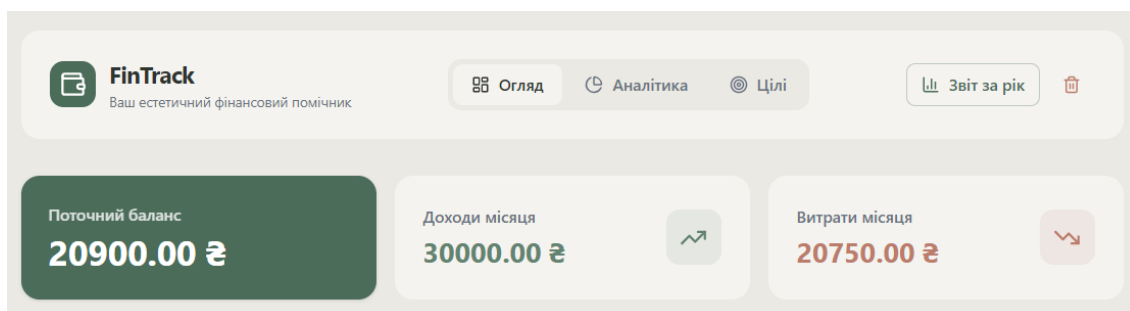


Рисунок 3.4 - Панель навігації та картки ключових показників ефективності

Інтерфейс програми має три основні функціональні стани (вкладки):

- вкладка «Огляд» (головний дашборд) (рисунок 3.5): візуально розділена на два вертикальні стовпчики. Лівий робочий простір містить форму «Нова операція» з інтерактивними кнопками перемикання типів транзакцій («Дохід»/«Витрата»), випадаючим динамічним списком категорій та числовим полем введення суми. Під формою реалізовано інтелектуальний блок «Рекомендації від FinTrack», який за допомогою алгоритмів аналізу даних генерує персоналізовані текстові підказки щодо оптимізації витрат, відсоткової частки

категорій у бюджеті та стабільності фінансового резерву. Правий стовпчик містить картку «Історія операцій» із вертикальним скролбаром. Кожен запис містить іконку напрямку руху коштів, колірне підсвічування суми (зелене/червоне), назву категорії, часову мітку та кнопку видалення [x], що ініціює миттєвий асинхронний перерахунок балансів та рекомендацій без перезавантаження сторінки;

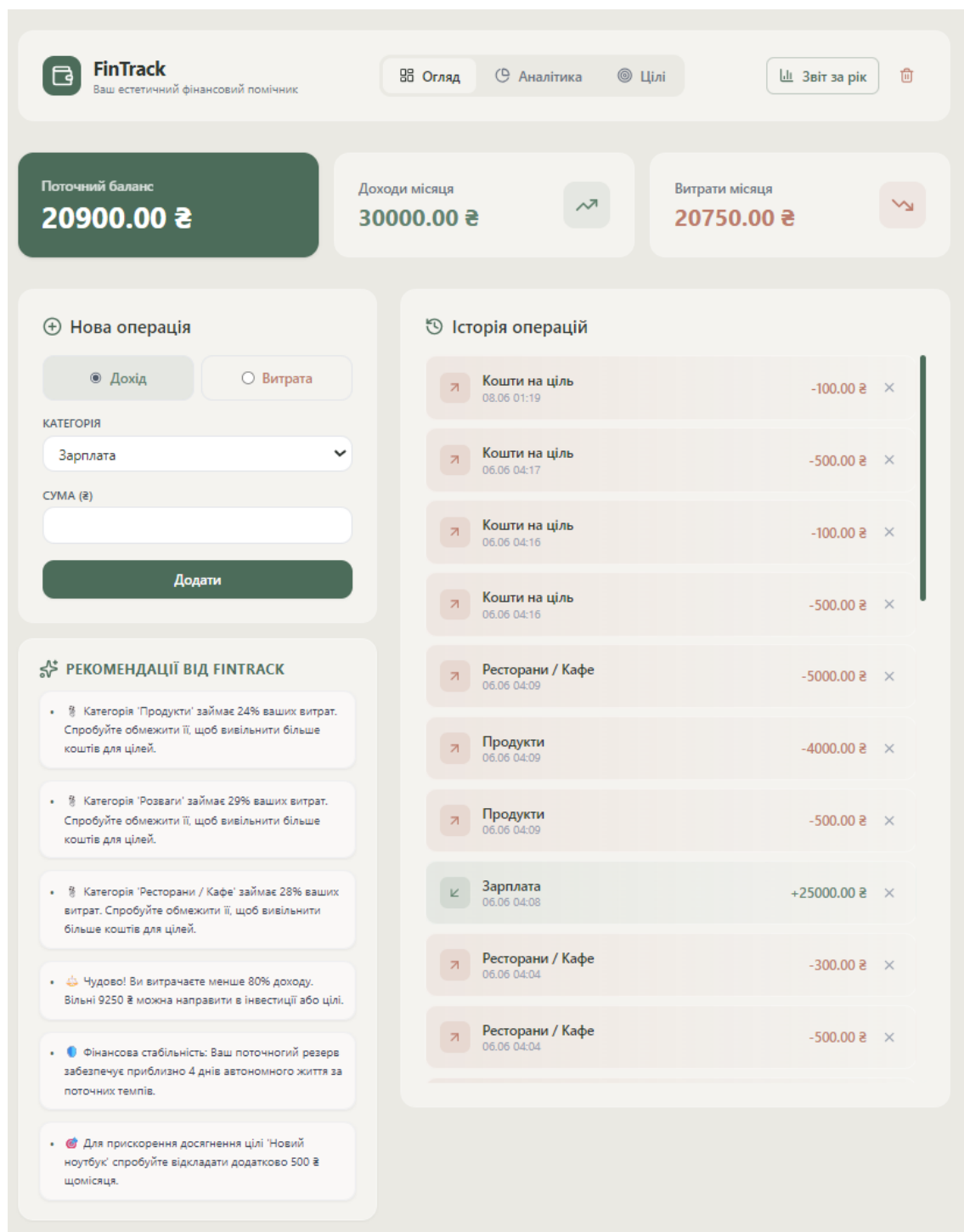
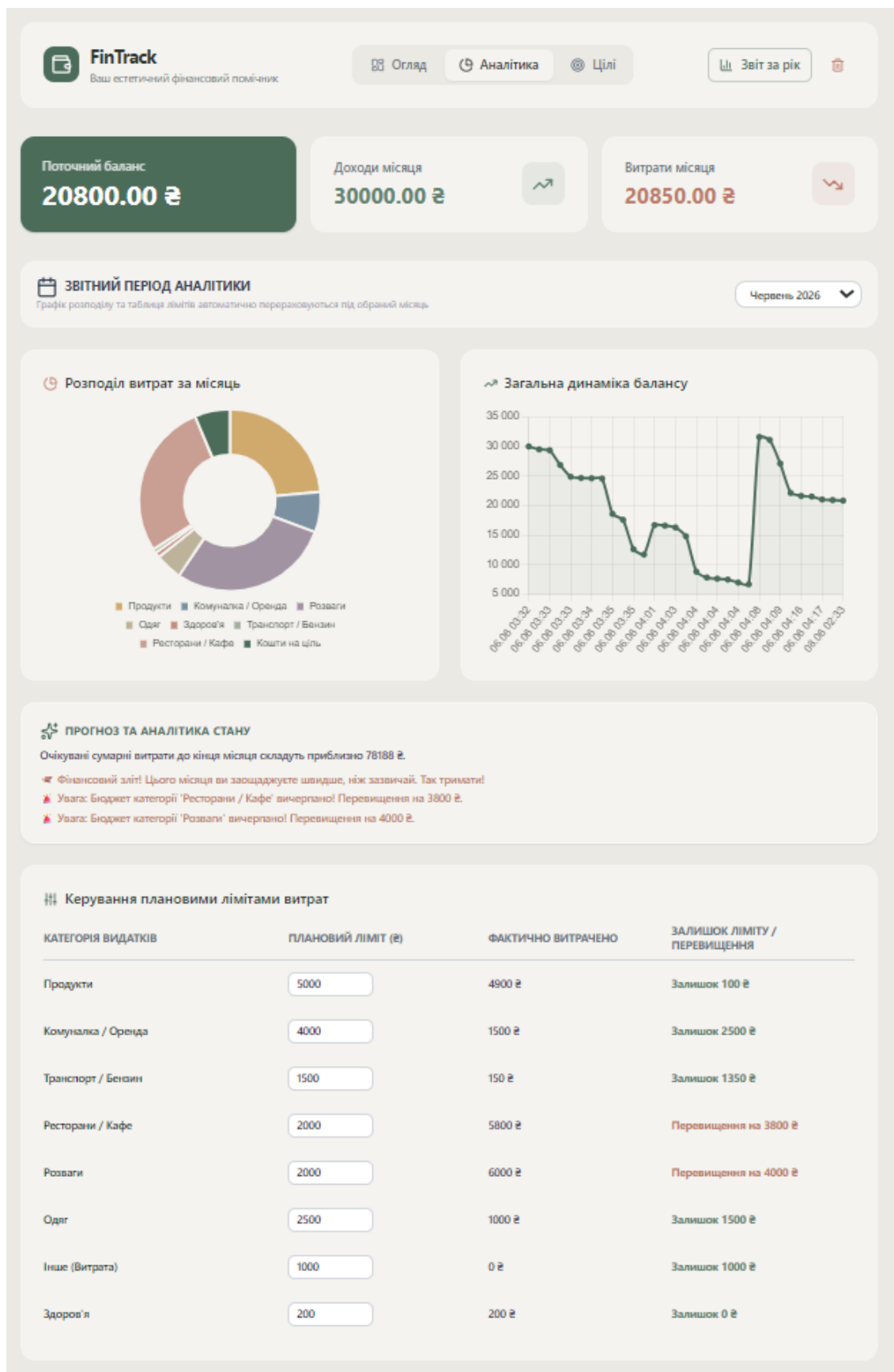


Рисунок 3.5 – Головний екран вкладки «Огляд»

– вкладка «Аналітика» (рисунок 3.6): спеціалізований аналітичний та прогностичний контур системи, призначений за для комплексного моніторингу руху капіталу, стратегічного планування витрат та автоматизованого контролю бюджетних обмежень у реальному часі. У верхній частині модуля інтегровано керуючий елемент «Звітний період аналітики» у вигляді випадаючих списків вибору місяця та року, при зміні яких JavaScript-модуль асинхронно ініціює HTTP GET-запит до серверних ендпоінтів Flask, моментально адаптуючи всі графічні та табличні компоненти вкладки під обраний період без перезавантаження вебсторінки. Центральну область займає спарений блок інтерактивної графічної звітності на базі елементів `<canvas>` бібліотеки Chart.js: кругова діаграма «Розподіл витрат за місяць» (Doughnut Chart) для наочного пропорційного маркування видатків із можливістю приховування окремих секторів, та лінійний графік «Загальна динаміка балансу» (Line Chart), що відображає ретроспективну криву коливання капіталу за ISO-мітками часу. Нижче інтегровано інтерактивний віджет «Прогноз та аналітика стану», який на основі аналізу поточної частоти та середнього чека транзакцій прораховує очікувані сумарні витрати до кінця календарного місяця, виводячи результат у вигляді KPI-індикаторів, текстових порад від ШІ-модуля та попереджувальних кольірних тригерів Tailwind CSS про ризики перевищення лімітів. Завершує інтерфейсну структуру вкладки функціональна таблиця «Керування плановими лімітами витрат» для прямого адміністрування лімітів через індивідуальні поля введення (Input Fields), при коригуванні яких система через Fetch API асинхронно оновлює реляційну таблицю budgets у файлі finance.db, після чого калькулятор сервера миттєво повертає фактично витрачені суми, а клієнтський скрипт динамічно підсвічує залишок бюджету або величину перевитрати відповідним кольором.



– вкладка «Цілі» (рисунок 3.7 та 3.8): візуалізує процес стратегічного планування та моніторингу особистих заощаджень. Панель розділена на дві функціональні зони: ліва містить форму створення нової фінансової мети із введенням назви (title) та цільової суми (target), а права — динамічно генерує картки вже створених цілей за допомогою JavaScript. Кожна картка відображає поточний прогрес у відсотках та оснащена плавною горизонтальною шкалою (прогрес-баром), що візуалізується засобами Tailwind CSS. Для швидкого фінансування цілей у структуру картки інтегровано індивідуальне поле введення та акцентну кнопку «Внести». При досягненні 100% виконання алгоритм автоматично блокує елементи керування, змінює статус об'єкта на завершений та ініціює запуск анімаційного салюту за допомогою бібліотеки canvas-confetti для гейміфікації процесу накопичень.

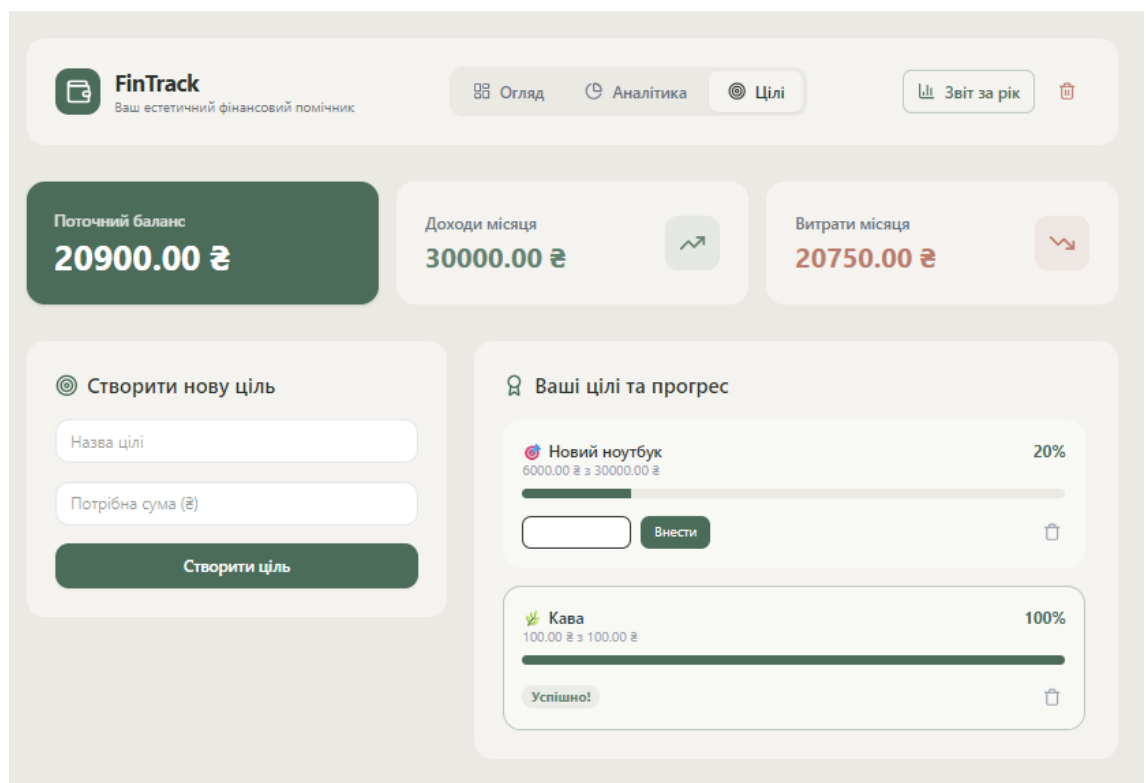


Рисунок 3.7 – Екран вкладки «Цілі»

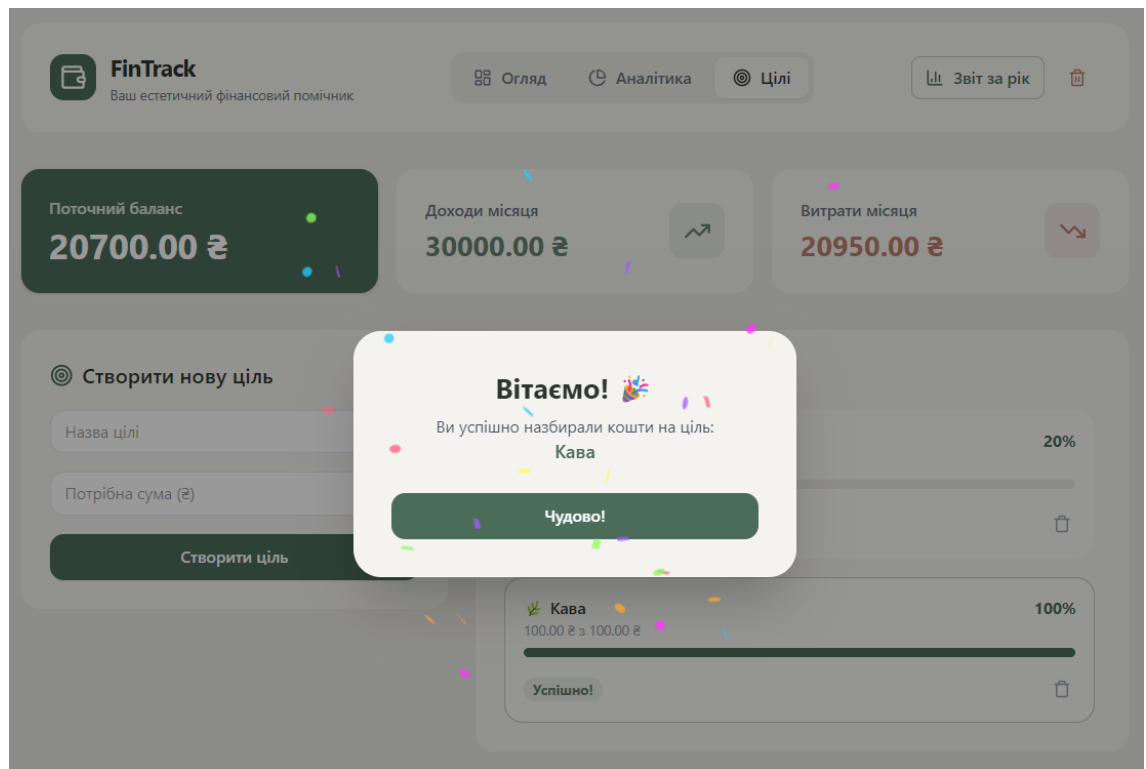


Рисунок 3.8 – Момент успішного досягнення фінансової цілі

Окремим аналітичним компонентом користувацького інтерфейсу є модальне вікно річного звіту (рисунок 3.9), яке з'являється у вигляді оверлея поверх поточного екрана без перезавантаження сторінки. Вікно «Річний аналітичний звіт» містить верхню інформаційну панель із трьома інтегрованими картками макропоказників («Річний дохід», «Річні витрати» та «Виконано цілей»).

Центральну область модального вікна займає двовимірною згрупована стовпчикова діаграма (Bar Chart) на базі HTML5-елемента `<canvas>` та графічного рушія Chart.js, яка реалізує помісячне зіставлення доходів та витрат користувача від січня до грудня. Оновлення стовпчиків діаграми здійснюється асинхронно через метод `chart.update()`. Для закриття аналітичного оверлея та повернення до робочого стану будь-якої з вкладок у верхній правій частині форми передбачено інтерактивний елемент «хрестик».

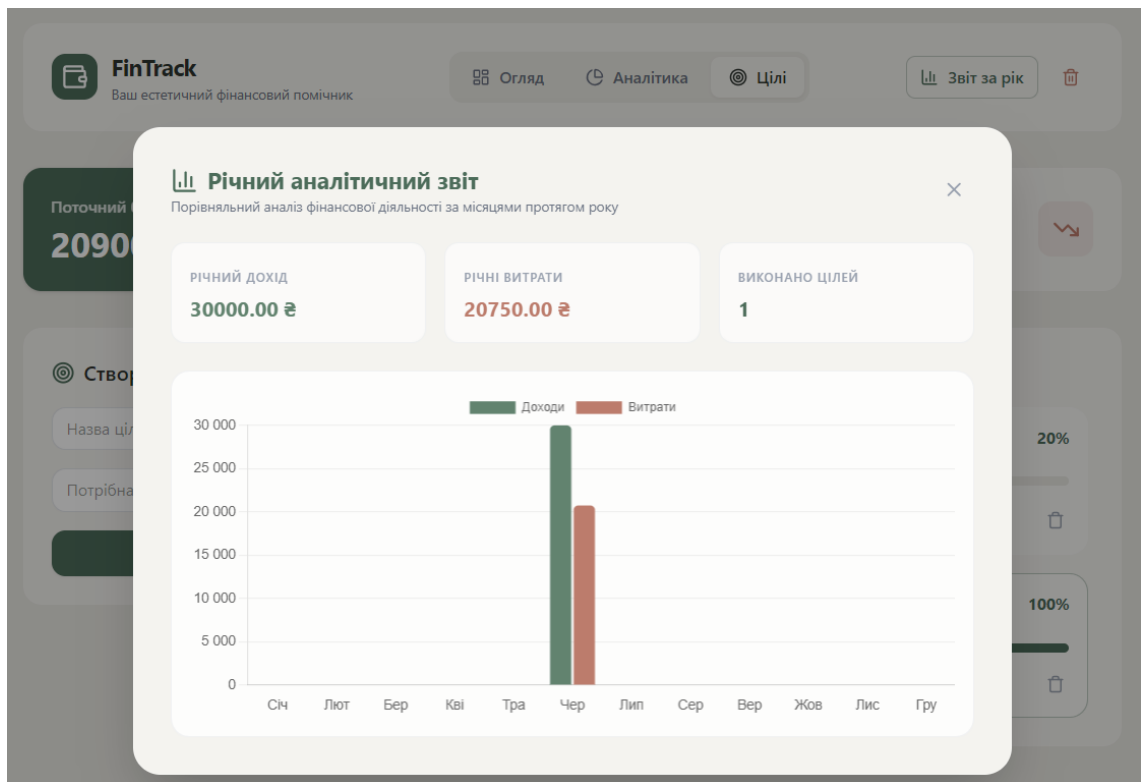


Рисунок 3.9 - Модальне вікно «Річний аналітичний звіт»

3.3 Тестування розробленої програми

Для перевірки коректності роботи бізнес-процесів, точності математичних обчислень та стабільності зв'язку через REST API було проведено функціональне тестування за методологією «чорної скриньки». Результати тестування на основі реальних даних наведено в таблиці 3.1.

Таблиця 3.1 – Результати функціонального тестування «FinTrack»

№ тесту	Опис тестового сценарію	Вхідні дані	Очікуваний результат	Фактичний результат	Статус
1	Перемикання списку категорій форми	Клік на радіокнопку Витрата	Завантаження видаткових категорій у селектор	Доступні категорії: Продукти, Комуналка, Ресторани, Здоров'я тощо	Успішно
2	Додавання нової витрати	Category: Ресторани / Кафе, Amount: 200.00	Зменшення балансу, додавання рядка в історію зі штампом часу	Баланс зменшився на 200 ₴. В історії з'явився новий верхній запис	Успішно

Продовження таблиці 3.1

№ тесту	Опис тестового сценарію	Вхідні дані	Очікуваний результат	Фактичний результат	Статус
3	Рендеринг кругової діаграми витрат	Перехід на вкладку Аналітика	Побудова секторів. Сектор "Кошти на ціль" має бути найбільшим	Графік побудовано. Розраховано та відображено частки всіх категорій	Успішно
4	Побудова лінійного графіка динаміки капіталу	Розрахунок масиву транзакцій	Відображення кривої лінії балансу, що спадає в хронологічному у порядку	Побудовано лінію з точками транзакцій, що відображає падіння до 4800 ₴	Успішно
5	Створення фінансової цілі	Title: Новий ноутбук, Target: 40000.00	Створення картки на вкладці Цілі, встановлення прогресу на 0%	Картка створена. Початковий прогрес зафіксовано	Успішно
6	Поповнення цілі з картки	ID цілі, Amount: 7000.00	Розрахунок прогресу (18%), оновлення прогрес-бару, створення авто-витрати	Прогрес оновився до 18% (7000 з 40000 ₴). В історії з'явився запис "Кошти на ціль"	Успішно
7	Досягнення цілі 100%	Ціль "Кава", Сума 100 з 100	Успішний REST-запит, конфеті-салют	Запущено візуальне привітання, ціль заблоковано для поповнення	Успішно
8	Видалення транзакції	Клік на [x] біля запису Здоров'я	Видалення рядка з БД за id, миттєвий перерахунок KPI карток	Запис зник із журналу. Поточний баланс та витрати перераховані автоматично	Успішно

Проведене тестування підтвердило повну працездатність усіх заявлених модулів програми, точність синхронізації даних між вкладками «Огляд» та «Аналітика», а також стабільність транзакцій у СКБД SQLite.

ВИСНОВКИ

У процесі виконання кваліфікаційної роботи було:

1. Досліджено предметну область керування особистим бюджетом та виконано порівняльний аналіз чинних ринкових аналогів (Wallet, Трекер бюджету, Фінанси професіонал). Це дозволило виявити функціональні розриви в існуючих програмах та обґрунтувати доцільність розробки автономного рішення для контролю власних коштів.

2. Сформульовано комплекс технічних, функціональних та нефункціональних вимог до системи. Визначено логіку роботи модульного SPA-інтерфейсу, параметри фіксації транзакцій, критерії встановлення бюджетних лімітів, а також вимоги до забезпечення повної конфіденційності та високої швидкодії веб-застосунку.

3. Обґрунтовано архітектурні рішення та побудовано відповідні UML-моделі системи. Створено діаграми активності та послідовності, які відображають логіку взаємодії користувача з інтерфейсом, а також алгоритмічне забезпечення для асинхронного обміну даними між клієнтською частиною та сервером.

4. Спроектовано та реалізовано структуру реляційної бази даних на базі СКБД SQLite. Розроблено логічну та фізичну моделі даних, де таблиці транзакцій, лімітів та фінансових цілей нормалізовано до третьої нормальної форми (3NF), що забезпечує стабільну роботу з інформацією, транзакційність (ACID) та унеможливорює появу аномалій.

5. Розроблено програмний веб-застосунок «FinTrack» відповідно до визначених вимог. Реалізовано серверну REST API логіку мовою Python на базі мікрофреймворку Flask та адаптивний користувацький інтерфейс за допомогою JavaScript і Tailwind CSS. Інтегровано інтелектуальний блок рекомендацій та алгоритми прогнозування витрат до кінця місяця на основі встановлених лімітів.

6. Реалізовано модулі візуалізації даних та побудови графіків. За допомогою бібліотеки Chart.js та елемента <canvas> впроваджено кругову діаграму розподілу витрат, лінійний графік динаміки капіталу та модальне вікно річного аналітичного звіту зі згрупованою стовпчиковою діаграмою, які оновлюються

асинхронно в режимі реального часу. Також реалізовано елементи гейміфікації за допомогою ефектів бібліотеки canvas-confetti.

7. Проведено функціональне тестування розробленого програмного забезпечення за методологією «чорної скриньки». Перевірка базових сценаріїв підтвердила повну коректність обробки фінансових даних, стабільність виконання деструктивних операцій видалення чи очищення сховища, а також адаптивність інтерфейсу під різні роздільні здатності екранів.

8. Розроблений веб-застосунок має високу практичну цінність як безкоштовний, автономний та захищений від витоку даних інструмент для раціонального розподілу бюджету та підвищення особистої самодисципліни. Продукт є повністю завершеним і може бути доопрацьований у майбутньому шляхом додавання модуля багатокористувацької автентифікації для сімейного бюджету або інтеграції засобів сповіщень у месенджери.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ginanjar I. K. N. Personal Finance Transformation in the Digital Era: The Impact of Financial Technology on People's Economic Behavior [Електронний ресурс]. IMEISJ. 2024. URL: <https://jurnal.stmik-mi.ac.id/imeisj/article/view/48> (дата звернення: 04.02.2026).
2. Prakash J., et al. Personal Finance Transformation in the Digital Era [Електронний ресурс]. ResearchGate, 2024. URL: https://www.researchgate.net/publication/392776574_Personal_Finance_Transformation_in_the_Digital_Era (дата звернення: 04.02.2026).
3. Abirami S., et al. NeuralBudget: An AI-Powered Personal Finance Management System [Електронний ресурс]. ResearchGate, 2025. URL: https://www.researchgate.net/publication/397260603_NeuralBudget_An_AI-Powered_Personal_Finance_Management_System (дата звернення: 04.02.2026).
4. Patil V., et al. A Critical Review of Personal Finance Tracker Technologies Among Gen Z [Електронний ресурс]. ResearchGate, 2024. URL: https://www.researchgate.net/publication/396484259_A_Critical_Review_of_Personal_Finance_Tracker_Technologies_Among_Gen_Z (дата звернення: 04.02.2026).
5. Kumar P., et al. Personal Finance Management Application [Електронний ресурс]. ResearchGate, 2024. URL: https://www.researchgate.net/publication/383469539_Personal_Finance_Management_Application (дата звернення: 04.02.2026).
6. Personal Finance Management System [Електронний ресурс]. ResearchGate, 2024. URL: https://www.researchgate.net/publication/392730244_Personal_Finance_Management (дата звернення: 04.02.2026).
7. Bashovyi V. M., Statsenko V. V., Statsenko D. V. Development of adaptive web interface for personal finance management application. Technologies and Engineering. 2022. URL: <https://doi.org/10.30857/2786-5371.2022.5.1> (дата звернення: 04.02.2026).

8. Singh T., et al. Personal Finance Tracker [Электронный ресурс]. ResearchGate, 2025. URL: https://www.researchgate.net/publication/397523239_PERSONAL_FINANCE_TRACKER (дата звернения: 04.02.2026).
9. Kaur M., et al. AI-Based Personal Finance Management System [Электронный ресурс]. International Journal of Engineering Research & Technology. 2024. URL: <https://www.ijert.org/ai-based-personal-finance-management-system> (дата звернения: 04.02.2026).
10. Verma S. Smart Finance Management Using Machine Learning [Электронный ресурс]. International Journal of Innovative Research in Technology. 2024. URL: https://ijirt.org/publishedpaper/IJIRT177688_PAPER.pdf (дата звернения: 04.02.2026).
11. Bhosale S., et al. Finance Tracker and Advisor [Электронный ресурс]. ResearchGate, 2025. URL: https://www.researchgate.net/publication/399247203_Finance_Tracker_and_Advisor (дата звернения: 04.02.2026).
12. Iyer S., et al. Smart Personal Finance Tracker for Efficient Money Management [Электронный ресурс]. ResearchGate, 2024. URL: https://www.researchgate.net/publication/392602422_Smart_Personal_Finance_Tracker_for_Efficient_Money_Management (дата звернения: 04.02.2026).
13. Nadeem M., et al. Fintrack: Personal Finance Tracker [Электронный ресурс]. ResearchGate, 2024. URL: https://www.researchgate.net/publication/380980910_Fintrack_Personal_Finance_Tracker (дата звернения: 04.02.2026).
14. Saini H., et al. The Influence of Technology in Personal Finance Management [Электронный ресурс]. ResearchGate, 2024. URL: https://www.researchgate.net/publication/381974842_THE_INFLUENCE_OF_TECHNOLOGY_IN_PERSONAL_FINANCE_MANAGEMENT (дата звернения: 04.02.2026).
15. Money manager & expenses [Электронный ресурс] / Cleaner + Antivirus + VPN company. Google Play. URL:

https://play.google.com/store/apps/details?id=ru.innim.my_finance (дата звернення: 04.02.2026).

16. Finance Pro: Expense control [Електронний ресурс] / Jorge de la Hoz. Google Play. URL: <https://play.google.com/store/apps/details?id=com.financepro> (дата звернення: 04.02.2026).

17. Personal Finance Tracker: Finance Management Application [Електронний ресурс]. International Journal of Advanced Research in Science, Communication and Technology. 2023. URL: <https://ijarsct.co.in/Paper29347.pdf> (дата звернення: 04.02.2026).

18. Personal Finance Management and Budgeting System [Електронний ресурс]. UNIMAS Institutional Repository. 2023. URL: <https://ir.unimas.my/id/eprint/44070/> (дата звернення: 04.02.2026).

19. Agrawal P., et al. Artificial Intelligence Powered Personal Finance Management System [Електронний ресурс]. ResearchGate, 2025. URL: https://www.researchgate.net/publication/401913715_Artificial_Intelligence_Powered_Personal_Finance_Management_System (дата звернення: 04.02.2026).

20. Gupta R., et al. Finance Tracker FINTRA: A Cross-Platform System for Personal Finance Tracking and Budgeting [Електронний ресурс]. ResearchGate, 2025. URL: https://www.researchgate.net/publication/397800911_Finance_Tracker_FINTRA_A_Cross-Platform_System_for_Personal_Finance_Tracking_and_Budgeting (дата звернення: 04.02.2026).

21. Tiwari N., Singh P., Mishra K. Design and Implementation of Expense Management System Using Web Technologies. International Journal of Computer Science and Mobile Computing. 2021.

22. Dinesh K., et al. CacheBook: A Smart Personal Finance Tracker with OCR-Based Expense Logging and Visual Reporting [Електронний ресурс]. ResearchGate, 2025. URL: https://www.researchgate.net/publication/399490009_CacheBook_A_Smart_Personal_Finance_Tracker_with_OCR-Based_Expense_Logging_and_Visual_Reporting (дата звернення: 04.02.2026).

23. Sharma A. Expense Tracker App: Design and Implementation [Електронний ресурс]. ResearchGate, 2024. URL: https://www.researchgate.net/publication/379444046_Expense_Tracker_App (дата звернення: 04.02.2026).

24. Mishra A., et al. Implementation of an Expense Tracker using Python [Електронний ресурс]. ResearchGate, 2024. URL: https://www.researchgate.net/publication/393738793_Implementation_of_an_Expense_Tracker_using_Python (дата звернення: 04.02.2026).

25. Hipp R. SQLite [Електронний ресурс]. SQLite Consortium. URL: <https://www.sqlite.org/index.html> (дата звернення: 04.02.2026).

26. Комар М. П., Васильків Н. М., Гладій Г. М., Коваль В. С. Програма переддипломної практики здобувачів першого (бакалаврського) рівня вищої освіти спеціальності 122 «Комп'ютерні науки» освітньо-професійної програми «Комп'ютерні науки». Тернопіль : ЗУНУ, 2024. 38 с.

27. Комар М. П., Саченко А. О., Васильків Н. М., Гладій Г. М., Коваль В. С., Ліп'яніна-Гончаренко Х. В. Методичні рекомендації до виконання кваліфікаційної роботи з освітньо-професійної програми «Комп'ютерні науки» спеціальності 122 «Комп'ютерні науки» за першим (бакалаврським) рівнем вищої освіти. Тернопіль : ЗУНУ, 2024. 52 с.

28. Гурняк С., Биковий П. Є. Веб-застосунок для керування персональними фінансами // Intelligent Computing Systems and Project Management (ICSPM 2026) : матеріали I Міжнародної науково-практичної конференції студентів та молодих учених. Тернопіль : Західноукраїнський національний університет, 2026. С. 177-179

Додаток А

Лістинг програмного коду клієнтської частини (index.html)

```
<!DOCTYPE html>
<html lang="uk">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>FinTrack</title>
  <script
src="https://cdn.jsdelivr.net/npm/@tailwindcss/browser@4"></script>
  <script>
    tailwind.config = {
      theme: {
        extend: {
          colors: { brand: { bg: '#EAE9E3', accent: '#4C6C5A', card:
'#F4F3EF', income: '#62836F', expense: '#BC7C6C' } }
        }
      }
    }
  </script>
  <script src="https://unpkg.com/lucide@latest"></script>
  <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/canvas-
confetti@1.6.0/dist/confetti.browser.min.js"></script>
  <style>
    input[type="number"]::-webkit-outer-spin-button,
    input[type="number"]::-webkit-inner-spin-button {
      -webkit-appearance: none;
      margin: 0;
    }
    input[type="number"] {
      -moz-appearance: textfield;
    }

    .custom-scroll::-webkit-scrollbar {
      width: 6px;
      height: 6px;
    }
    .custom-scroll::-webkit-scrollbar-track {
      background: rgba(244, 243, 239, 0.5);
      border-radius: 10px;
    }
    .custom-scroll::-webkit-scrollbar-thumb {
      background-color: #4C6C5A;
      border-radius: 10px;
      transition: background-color 0.2s ease;
    }
    .custom-scroll::-webkit-scrollbar-thumb:hover {
      background-color: #3D5748;
    }
  </style>
</head>
<body class="bg-[#EAE9E3] text-[#2D332F] font-sans antialiased min-h-screen">

  <div id="congrats-modal" class="hidden fixed inset-0 bg-black/40 flex
items-center justify-center z-50 p-4">
    <div class="bg-[#F4F3EF] rounded-3xl p-8 max-w-sm w-full text-center
shadow-2xl border border-white/40">
      <h3 class="text-2xl font-bold mb-2">Біраємо! 🎉</h3>
```

```

        <p class="text-gray-600 text-sm mb-6">Ви успішно назбирали кошти
на ціль: <br><span id="congrats-goal-name" class="font-semibold text-[#4C6C5A] text-
base"></span></p>
        <button onclick="document.getElementById('congrats-
modal').classList.add('hidden')" class="w-full bg-[#4C6C5A] text-white font-medium
py-2.5 rounded-xl text-sm cursor-pointer">Чудово!</button>
    </div>
</div>

    <div id="report-modal" class="hidden fixed inset-0 bg-black/40 flex items-
center justify-center z-50 p-4 overflow-y-auto">
        <div class="bg-[#F4F3EF] rounded-3xl p-6 sm:p-8 max-w-3xl w-full
shadow-2xl border border-white/40 my-auto">
            <div class="flex justify-between items-center mb-6">
                <div>
                    <h3 class="text-xl font-bold flex items-center gap-2 text-
[#4C6C5A]"><i data-lucide="bar-chart-3"></i> Річний аналітичний звіт</h3>
                    <p class="text-xs text-gray-500">Порівняльний аналіз
фінансової діяльності за місяцями протягом року</p>
                </div>
                <button onclick="document.getElementById('report-
modal').classList.add('hidden')" class="text-gray-400 hover:text-gray-700 p-2"><i
data-lucide="x" class="w-5 h-5"></i></button>
            </div>

            <div class="grid grid-cols-1 md:grid-cols-3 gap-4 mb-6">
                <div class="bg-white/60 p-4 rounded-xl border border-gray-
200/50">
                    <span class="text-[11px] uppercase tracking-wider text-
gray-400 font-medium">Річний дохід</span>
                    <p id="rep-year-income" class="text-lg font-bold text-
[#62836F] mt-0.5">0.00 ₾</p>
                </div>
                <div class="bg-white/60 p-4 rounded-xl border border-gray-
200/50">
                    <span class="text-[11px] uppercase tracking-wider text-
gray-400 font-medium">Річні витрати</span>
                    <p id="rep-year-expense" class="text-lg font-bold text-
[#BC7C6C] mt-0.5">0.00 ₾</p>
                </div>
                <div class="bg-white/60 p-4 rounded-xl border border-gray-
200/50">
                    <span class="text-[11px] uppercase tracking-wider text-
gray-400 font-medium">Виконано цілей</span>
                    <p id="rep-year-goals" class="text-lg font-bold text-
[#4C6C5A] mt-0.5">0</p>
                </div>
            </div>

            <div class="bg-white/80 p-4 rounded-2xl border border-gray-200/50
h-[320px]">
                <canvas id="yearComparisonChart"></canvas>
            </div>
        </div>
    </div>

    <div class="max-w-5xl mx-auto p-4 sm:p-6 lg:p-8">
        <!-- Шапка -->
        <header class="flex flex-col lg:flex-row lg:items-center justify-
between gap-4 mb-8 bg-[#F4F3EF] p-6 rounded-2xl border border-white/20">
            <div class="flex items-center gap-3">
                <div class="bg-[#4C6C5A] text-white p-2 rounded-xl"><i data-
lucide="wallet" class="w-6 h-6"></i></div>
            </div>

```

```

<h1 class="text-xl font-bold tracking-tight text-
[#2D332F]">FinTrack</h1>
<p class="text-xs text-gray-500">Ваш естетичний фінансовий
помічник</p>
</div>
</div>
<div class="flex flex-row items-center bg-[#EAE9E3] p-1 rounded-
xl overflow-x-auto max-w-full gap-1">
<button id="tab-dashboard" class="px-4 py-2 text-sm font-
medium rounded-lg bg-[#F4F3EF] text-[#2D332F] cursor-pointer flex items-center gap-
2 transition whitespace-nowrap">
<i data-lucide="layout-dashboard" class="w-4 h-4"></i>
Огляд
</button>
<button id="tab-analytics" class="px-4 py-2 text-sm font-
medium rounded-lg text-gray-500 hover:text-[#2D332F] cursor-pointer flex items-
center gap-2 transition whitespace-nowrap">
<i data-lucide="pie-chart" class="w-4 h-4"></i> Аналітика
</button>
<button id="tab-goals" class="px-4 py-2 text-sm font-medium
rounded-lg text-gray-500 hover:text-[#2D332F] cursor-pointer flex items-center gap-
2 transition whitespace-nowrap">
<i data-lucide="target" class="w-4 h-4"></i> Цілі
</button>
</div>
<div class="flex items-center gap-2">
<button onclick="openYearReport()" class="text-sm text-[#4C6C5A]
hover:bg-[#4C6C5A]/10 border border-[#4C6C5A]/30 px-3 py-2 rounded-lg font-medium
transition flex items-center gap-2 cursor-pointer whitespace-nowrap">
<i data-lucide="bar-chart-3" class="w-4 h-4"></i> Звіт за
рік
</button>
<button id="clear-all" class="text-sm text-[#BC7C6C] hover:bg-
[#BC7C6C]/10 px-3 py-2 rounded-lg font-medium transition flex items-center gap-2
cursor-pointer">
<i data-lucide="trash-2" class="w-4 h-4"></i>
</button>
</div>
</header>
<!-- Картки балансу -->
<div class="grid grid-cols-1 md:grid-cols-3 gap-4 mb-8">
<div class="bg-[#4C6C5A] text-white p-6 rounded-2xl shadow-sm">
<p class="text-sm opacity-80 font-medium">Поточний баланс</p>
<p id="total-balance" class="text-3xl font-bold mt-1">0.00
€</p>
</div>
<div class="bg-[#F4F3EF] p-6 rounded-2xl border border-white/20
flex items-center justify-between">
<div>
<p class="text-sm text-gray-500 font-medium">Доходи
місяця</p>
<p id="total-income" class="text-2xl font-bold text-
[#62836F] mt-1">0.00 €</p>
</div>
<div class="bg-[#62836F]/10 text-[#62836F] p-3 rounded-xl">
<i data-lucide="trending-up" class="w-6 h-6"></i>
</div>
</div>
<div class="bg-[#F4F3EF] p-6 rounded-2xl border border-white/20
flex items-center justify-between">
<div>

```

```

        <p class="text-sm text-gray-500 font-medium">Витрати
місяця</p>
        <p id="total-expense" class="text-2xl font-bold text-
[#BC7C6C] mt-1">0.00 ₪</p>
        </div>
        <div class="bg-[#BC7C6C]/10 text-[#BC7C6C] p-3 rounded-xl">
            <i data-lucide="trending-down" class="w-6 h-6"></i>
        </div>
    </div>
</div>

<!-- ВКЛАДКА 1: ОБГЛЯД -->
<div id="content-dashboard" class="grid grid-cols-1 md:grid-cols-5
gap-6 items-start">
    <div class="md:col-span-2 space-y-4">
        <section class="bg-[#F4F3EF] p-6 rounded-2xl border border-
white/20 h-fit">
            <h2 class="text-lg font-semibold mb-4 flex items-center
gap-2">
                <i data-lucide="plus-circle" class="w-5 h-5 text-
[#4C6C5A]"></i> Нова операція
            </h2>
            <form id="transaction-form" class="space-y-4">
                <div>
                    <div class="grid grid-cols-2 gap-2">
                        <label class="border border-gray-200 rounded-
xl p-3 flex items-center justify-center gap-2 cursor-pointer has-[:checked]:bg-
[#62836F]/10 transition">
                            <input type="radio" name="type"
value="income" checked class="text-[#62836F]">
                            <span class="text-sm font-medium text-
[#62836F]">Дохід</span>
                        </label>
                        <label class="border border-gray-200 rounded-
xl p-3 flex items-center justify-center gap-2 cursor-pointer has-[:checked]:bg-
[#BC7C6C]/10 transition">
                            <input type="radio" name="type"
value="expense" class="text-[#BC7C6C]">
                            <span class="text-sm font-medium text-
[#BC7C6C]">Витрата</span>
                        </label>
                    </div>
                </div>
                <div>
                    <label class="block text-xs font-medium text-gray-
600 uppercase mb-1">Категорія</label>
                    <select id="category" required class="w-full px-
3 py-2 border border-gray-200 rounded-xl text-sm bg-white"></select>
                </div>
                <div>
                    <label class="block text-xs font-medium text-gray-
600 uppercase mb-1">Сума (₪)</label>
                    <input type="number" id="amount" step="0.01"
required class="w-full px-3 py-2 border border-gray-200 rounded-xl text-sm bg-white">
                </div>
                <button type="submit" class="w-full bg-[#4C6C5A]
hover:bg-[#3D5748] text-white font-medium py-2.5 rounded-xl text-sm cursor-pointer
shadow-xs">Додати</button>
            </form>
        </section>

        <section class="bg-[#F4F3EF] border border-white/40 p-5
rounded-2xl shadow-xs">
            <div class="flex items-center gap-2 mb-3 text-[#4C6C5A]">
                <i data-lucide="sparkles" class="w-5 h-5"></i>

```

```

        <h4 class="text-sm font-bold uppercase tracking-
wider">Рекомендації від FinTrack</h4>
        </div>
        <ul id="ui-recommendations-list" class="text-xs text-gray-
700 space-y-3 list-none"></ul>
        </section>
    </div>

    <section class="md:col-span-3 bg-[#F4F3EF] p-6 rounded-2xl border
border-white/20 flex flex-col min-h-[400px]">
        <h2 class="text-lg font-semibold mb-4 flex items-center gap-
2">
            <i data-lucide="history" class="w-5 h-5 text-[#4C6C5A]"></i>
Історія операцій
        </h2>
        <div class="overflow-y-auto flex-1 max-h-[750px] pr-1 custom-
scroll">
            <ul id="transaction-list" class="space-y-2"></ul>
            <div id="empty-state" class="hidden text-center py-12
text-gray-400">
                <i data-lucide="layers" class="w-12 h-12 mx-auto mb-2
opacity-30"></i>
                <p class="text-sm">Тут поки немає жодної операції</p>
            </div>
        </div>
    </section>
</div>

<!-- ВКЛАДКА 2: АНАЛІТИКА -->
<div id="content-analytics" class="hidden space-y-6">

    <div class="bg-[#F4F3EF] p-4 rounded-2xl border border-white/20
flex flex-col sm:flex-row justify-between items-start sm:items-center gap-3">
        <div>
            <h3 class="text-sm font-bold text-gray-700 uppercase flex
items-center gap-2"><i data-lucide="calendar"></i> Звітний період аналітики</h3>
            <p class="text-[11px] text-gray-400">Графік розподілу та
таблиця лімітів автоматично перераховуються під обраний місяць</p>
        </div>
        <select id="ui-month-selector" onchange="onMonthChanged()"
class="px-3 py-1.5 border border-gray-300 rounded-xl bg-white text-xs font-semibold
focus:outline-hidden focus:ring-1 focus:ring-[#4C6C5A]">
            <option value="2026-06" selected>Червень 2026</option>
            <option value="2026-05">Травень 2026</option>
            <option value="2026-04">Квітень 2026</option>
            <option value="2026-03">Березень 2026</option>
            <option value="2026-02">Лютий 2026</option>
            <option value="2026-01">Січень 2026</option>
        </select>
    </div>

    <div class="grid grid-cols-1 md:grid-cols-2 gap-6">
        <div class="bg-[#F4F3EF] p-6 rounded-2xl border border-white/20
flex flex-col items-center">
            <h3 class="text-base font-semibold mb-4 self-start flex
items-center gap-2">
                <i data-lucide="pie-chart" class="w-4 h-4 text-
[#BC7C6C]"></i> Розподіл витрат за місяць
            </h3>
            <div class="w-full max-w-[280px] aspect-square relative
flex items-center justify-center">
                <canvas id="expenseChart"></canvas>
                <p id="no-expense-data" class="hidden text-sm text-
gray-400 absolute text-center">Немає даних про витрати</p>
            </div>
        </div>
    </div>

```

```

        </div>
        <div class="bg-[#F4F3EF] p-6 rounded-2xl border border-white/20
flex flex-col">
            <h3 class="text-base font-semibold mb-4 flex items-center
gap-2">
                <i data-lucide="trending-up" class="w-4 h-4 text-
[#4C6C5A]"></i> Загальна динаміка балансу
            </h3>
            <div class="w-full flex-1 min-h-[250px] relative flex
items-center justify-center">
                <canvas id="balanceChart"></canvas>
            </div>
        </div>
    </div>

    <!-- ЗМІНЕНО КОЛІР КАРТКИ: Тепер вона така ж кремово-естетична,
як і блок рекомендацій -->
    <div class="bg-[#F4F3EF] border border-white/40 p-5 rounded-2xl
shadow-xs">
        <div class="flex items-center gap-2 mb-2 text-[#4C6C5A]">
            <i data-lucide="sparkles" class="w-5 h-5"></i>
            <h4 class="text-xs font-bold uppercase tracking-
wider">Прогноз та аналітика стану</h4>
        </div>
        <div>
            <p id="ui-forecast-text" class="text-xs text-gray-700
font-medium leading-relaxed">Завантаження розрахунків...</p>
            <div id="ui-notifications-container" class="mt-2 space-y-
1.5 text-xs text-[#BC7C6C] font-semibold"></div>
        </div>

        <div class="bg-[#F4F3EF] p-6 rounded-2xl border border-white/20">
            <h3 class="text-base font-semibold mb-4 flex items-center gap-
2"><i data-lucide="sliders" class="w-4 h-4 text-[#4C6C5A]"></i> Керування плановими
лімітами витрат</h3>
            <div class="overflow-x-auto custom-scroll">
                <table class="w-full text-left text-xs table-fixed min-w-
[600px]">
                    <thead>
                        <tr class="border-b border-gray-300 text-gray-500
uppercase">
                            <th class="pb-2 w-1/3">Категорія видатків</th>
                            <th class="pb-2 w-56">Плановий ліміт (€)</th>
                            <th class="pb-2 w-1/4">Фактично витрачено</th>
                            <th class="pb-2 w-1/4">Залишок ліміту /
Перевищення</th>
                        </tr>
                    </thead>
                    <tbody id="ui-deviations-table-body" class="divide-y
divide-gray-200"></tbody>
                </table>
            </div>
        </div>
    </div>

    <!-- ВКЛАДКА 3: ЦІЛІ -->
    <div id="content-goals" class="hidden grid grid-cols-1 md:grid-cols-
5 gap-6">
        <section class="md:col-span-2 bg-[#F4F3EF] p-6 rounded-2xl border
border-white/20 h-fit">
            <h2 class="text-lg font-semibold mb-4 flex items-center gap-
2"><i data-lucide="target" class="w-5 h-5 text-[#4C6C5A]"></i> Створити нову
ціль</h2>
            <form id="goal-form" class="space-y-4">

```

```

        <input type="text" id="goal-title" required
placeholder="Назва цілі" class="w-full px-3 py-2 border border-gray-200 rounded-xl
text-sm bg-white">
        <input type="number" id="goal-target" step="0.01" required
placeholder="Потрібна сума (€)" class="w-full px-3 py-2 border border-gray-200
rounded-xl text-sm bg-white">
        <button type="submit" class="w-full bg-[#4C6C5A] text-
white font-medium py-2.5 rounded-xl text-sm cursor-pointer shadow-xs">Створити
ціль</button>

    </form>
</section>
<section class="md:col-span-3 bg-[#F4F3EF] p-6 rounded-2xl border
border-white/20 flex flex-col min-h-[350px]">
    <h2 class="text-lg font-semibold mb-4 flex items-center gap-
2"><i data-lucide="award" class="w-5 h-5 text-[#4C6C5A]"></i> Ваші цілі та
прогрес</h2>
    <div id="goals-list" class="space-y-4 overflow-y-auto max-h-
[500px] pr-1 custom-scroll"></div>
    <div id="goals-empty-state" class="hidden text-center py-12
text-gray-400 text-xs">Цілей немає.</div>
</section>
</div>
</div>

<script>
    const API_URL = "http://127.0.0.1:5000/api";
    lucide.createIcons();

    const categories = {
        income: ['Зарплата', 'Фріланс', 'Інвестиції', 'Подарунок', 'Інше
(Дохід)'],
        expense: ['Продукти', 'Комуналка / Оренда', 'Транспорт / Бензин',
'Ресторани / Кафе', 'Розваги', 'Одяг', 'Здоров'я', 'Інше (Витрата)']
    };

    const categoryColors = {
        'Продукти': '#D0A96C', 'Комуналка / Оренда': '#7B91A1', 'Транспорт
/ Бензин': '#A3B19B',
        'Ресторани / Кафе': '#C99E93', 'Розваги': '#A293A3', 'Одяг':
'#BDB29A', 'Здоров'я': '#C28D80',
        'Кошти на цілі': '#4C6C5A', 'Інше (Витрата)': '#8A928E'
    };

    const tabs = {
        dashboard: { btn: document.getElementById('tab-dashboard'), content:
document.getElementById('content-dashboard') },
        analytics: { btn: document.getElementById('tab-analytics'), content:
document.getElementById('content-analytics') },
        goals: { btn: document.getElementById('tab-goals'), content:
document.getElementById('content-goals') }
    };

    let expenseChartInstance = null, balanceChartInstance = null,
yearChartInstance = null, localTransactions = [];

    function displayCategoryName(name) {
        return name === 'Здоров'я' ? "Здоров'я" : name;
    }

    function formatDisplayDate(isoDateStr) {
        if (!isoDateStr || !isoDateStr.includes('-')) return isoDateStr;
        const parts = isoDateStr.split(' ');
        const dateParts = parts[0].split('-');
        return `${dateParts[2]}.${dateParts[1]} ${parts[1]}`;
    }

```

```

function updateCategorySelect() {
    const selectedType =
document.querySelector('input[name="type"]:checked').value;
    const select = document.getElementById('category');
    select.innerHTML = '';
    categories[selectedType].forEach(cat => {
        const opt = document.createElement('option'); opt.value = cat;
    opt.textContent = displayCategoryName(cat);
        select.appendChild(opt);
    });
}

document.querySelectorAll('input[name="type"]').forEach(r =>
r.addEventListener('change', updateCategorySelect));

function switchTab(activeKey) {
    Object.keys(tabs).forEach(key => {
        if (key === activeKey) {
            tabs[key].btn.className = "px-4 py-2 text-sm font-medium
rounded-lg bg-[#F4F3EF] text-[#2D332F] shadow-xs cursor-pointer flex items-center
gap-2";
            tabs[key].content.classList.remove('hidden');
        } else {
            tabs[key].btn.className = "px-4 py-2 text-sm font-medium
rounded-lg text-gray-500 hover:text-[#2D332F] cursor-pointer flex items-center gap-
2";
            tabs[key].content.classList.add('hidden');
        }
    });
    if (activeKey === 'analytics') { loadIntelligence();
renderCharts(); }
    if (activeKey === 'goals') loadGoals();
}

Object.keys(tabs).forEach(key =>
tabs[key].btn.addEventListener('click', () => switchTab(key)));

function onMonthChanged() {
    loadIntelligence();
    renderCharts();
}

async function loadData() {
    const res = await fetch(`${API_URL}/transactions`);
    localTransactions = await res.json();
    updateUI(localTransactions);
    loadIntelligence();
}

async function loadIntelligence() {
    const currentMonth = document.getElementById('ui-month-
selector').value;
    const res = await
fetch(`${API_URL}/intelligence?month=${currentMonth}`);
    const data = await res.json();

    document.getElementById('ui-forecast-text').textContent =
        `Очікувані сумарні витрати до кінця місяця складуть приблизно
${data.forecast.toFixed(0)} ₴`;

    document.getElementById('total-income').textContent =
`${data.month_income.toFixed(2)} ₴`;
    document.getElementById('total-expense').textContent =
`${data.month_expense.toFixed(2)} ₴`;

```

```

const recContainer = document.getElementById('ui-recommendations-
list');
recContainer.innerHTML = '';
data.recommendations.forEach(r => {
  const li = document.createElement('li');
  li.className = "p-2.5 rounded-xl bg-white/50 border border-
gray-200/50 flex items-start gap-2.5 leading-relaxed shadow-xs text-gray-700";
  li.innerHTML = `<span class="text-[#4C6C5A] font-semibold
select-none">•</span><span>${r.replace(/Здоровя/g, "Здоров'я")}</span>`;
  recContainer.appendChild(li);
});

const notifContainer = document.getElementById('ui-notifications-
container');
notifContainer.innerHTML = '';
data.notifications.forEach(n => {
  const p = document.createElement('p'); p.textContent =
n.replace(/Здоровя/g, "Здоров'я"); notifContainer.appendChild(p);
});

const tableBody = document.getElementById('ui-deviations-table-
body');
tableBody.innerHTML = '';
data.deviations.forEach(d => {
  const tr = document.createElement('tr');
  tr.className = "border-b border-gray-100 text-xs";
  const isOverspent = d.deviation < 0;

  tr.innerHTML = `
    <td class="py-3 font-medium text-gray-
900">${displayCategoryName(d.category)}</td>
    <td class="py-3 flex items-center gap-2">
      <input type="number"
                                id="input-limit-
${encodeURIComponent(d.category)}"
                                value="${d.plan.toFixed(0)}"
                                onkeydown="if(event.key==='Enter')
saveLimit('${encodeURIComponent(d.category)}') "
                                onblur="saveLimit('${encodeURIComponent(d.cat
egory)}') "
                                class="w-24 px-2 py-1 border border-gray-300
rounded-lg bg-white text-left font-semibold text-gray-800 focus:outline-hidden
focus:ring-1 focus:ring-[#4C6C5A]">
      <span id="status-${encodeURIComponent(d.category)}"
class="text-xs text-[#62836F] font-bold opacity-0 transition-opacity duration-300
flex items-center gap-0.5">✓ <span class="text-[10px]">Збережено</span></span>
    </td>
    <td class="py-3 font-medium text-gray-
700">${d.fact.toFixed(0)} €</td>
    <td class="py-3 font-bold ${isOverspent ? 'text-[#BC7C6C]' :
'text-[#62836F]'}">
      ${isOverspent ? 'Перевищення на ' +
Math.abs(d.deviation).toFixed(0) : 'Залишок ' + d.deviation.toFixed(0)} €
    </td>
  `;
  tableBody.appendChild(tr);
});

window.saveLimit = async function(encodedCategory) {
  const category = decodeURIComponent(encodedCategory);
  const inputVal = document.getElementById(`input-limit-
${encodeURIComponent(category)}`).value;
  await fetch(`${API_URL}/budgets/update`, {
    method: 'POST',

```

```

        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({ category: category, limit: inputVal })
    });
    const statusEl = document.getElementById(`status-${
    ${encodedCategory}`);
    statusEl.className = "text-xs text-[#62836F] font-bold opacity-
    100 transition-opacity duration-300 flex items-center gap-0.5";
    setTimeout(() => {
        statusEl.className = "text-xs text-[#62836F] font-bold opacity-
    0 transition-opacity duration-300 flex items-center gap-0.5";
        loadData();
    }, 1200);
};

window.openYearReport = async function() {
    document.getElementById('report-modal').classList.remove('hidden');
    lucide.createIcons();

    const res = await fetch(`${API_URL}/report/year`);
    const data = await res.json();

    document.getElementById('rep-year-income').textContent =
    `${data.year_income.toFixed(2)} ₾`;
    document.getElementById('rep-year-expense').textContent =
    `${data.year_expense.toFixed(2)} ₾`;
    document.getElementById('rep-year-goals').textContent =
    data.completed_goals;

    const ukrMonths = ["Січ", "Лют", "Бер", "Кві", "Тра", "Чер", "Лип",
    "Сер", "Вер", "Жов", "Лис", "Гру"];
    const incomeDataset = [];
    const expenseDataset = [];

    for(let i=1; i<=12; i++) {
        const key = String(i).padStart(2, '0');
        incomeDataset.push(data.months[key].income);
        expenseDataset.push(data.months[key].expense);
    }

    if (yearChartInstance) yearChartInstance.destroy();
    yearChartInstance = new
    Chart(document.getElementById('yearComparisonChart').getContext('2d'), {
        type: 'bar',
        data: {
            labels: ukrMonths,
            datasets: [
                { label: 'Доходи', data: incomeDataset, backgroundColor:
    '#62836F', borderRadius: 4 },
                { label: 'Витрати', data: expenseDataset,
    backgroundColor: '#BC7C6C', borderRadius: 4 }
            ]
        },
        options: {
            responsive: true,
            maintainAspectRatio: false,
            scales: { y: { beginAtZero: true, grid: { color:
    'rgba(0,0,0,0.05)' } }, x: { grid: { display: false } } },
            plugins: { legend: { labels: { font: { size: 11 } } } }
        }
    });
};

function updateUI(transactionsList) {
    const container = document.getElementById('transaction-list');
    container.innerHTML = '';

```

```

        if (transactionsList.length === 0)
        { document.getElementById('empty-state').classList.remove('hidden'); return; }
        document.getElementById('empty-state').classList.add('hidden');

        let balance = 0;
        [...transactionsList].reverse().forEach(t => {
            const isInc = t.type === 'income';
            balance += isInc ? t.amount : -t.amount;

            const li = document.createElement('li');
            li.className = `flex items-center justify-between p-3.5
rounded-xl border border-white/40 bg-linear-to-r ${isInc ? 'from-[#62836F]/10 to-
transparent' : 'from-[#BC7C6C]/10 to-transparent'}`;
            li.innerHTML = `
                <div class="flex items-center gap-3">
                    <div class="p-2 rounded-lg ${isInc ? 'bg-[#62836F]/15
text-[#62836F]' : 'bg-[#BC7C6C]/15 text-[#BC7C6C]}'">
                        <i data-lucide="${isInc ? 'arrow-down-left' :
'arrow-up-right'}" class="w-4 h-4"></i>
                    </div>
                    <div>
                        <p class="font-medium text-sm text-
[#2D332F]">${displayCategoryName(t.category)}</p>
                        <p class="text-xs text-gray-
400">${formatDisplayDate(t.date)}</p>
                    </div>
                </div>
                <div class="flex items-center gap-3">
                    <span class="font-semibold text-sm ${isInc ? 'text-
[#62836F]' : 'text-[#BC7C6C]}'">${isInc ? '+' : '-'}${t.amount.toFixed(2)} €</span>
                    <button onclick="deleteTransaction(${t.id})"
class="text-gray-400 hover:text-[#BC7C6C] p-1 rounded-md hover:bg-white/40
transition cursor-pointer"><i data-lucide="x" class="w-4 h-4"></i></button>
                </div>
            `;
            container.appendChild(li);
        });

        document.getElementById('total-balance').textContent =
`$${balance.toFixed(2)} €`;
        lucide.createIcons();
    }

    document.getElementById('transaction-form').addEventListener('submit',
async (e) => {
        e.preventDefault();
        const amount = document.getElementById('amount').value;
        const type =
document.querySelector('input[name="type"]:checked').value;
        const category = document.getElementById('category').value;
        await fetch(`${API_URL}/transactions`, { method: 'POST', headers:
{ 'Content-Type': 'application/json' }, body: JSON.stringify({ category, amount,
type }) });
        document.getElementById('amount').value = '';
        loadData();
    });

    window.deleteTransaction = async function(id) { await
fetch(`${API_URL}/transactions/${id}`, { method: 'DELETE' }); loadData(); };

    async function loadGoals() {
        const res = await fetch(`${API_URL}/goals`); const goals = await
res.json();
        const container = document.getElementById('goals-list');
        container.innerHTML = '';
    }

```

```

        if (goals.length === 0) { document.getElementById('goals-empty-
state').classList.remove('hidden'); return; }
        document.getElementById('goals-empty-
state').classList.add('hidden');

        goals.forEach(g => {
            const pct = Math.min((g.current / g.target) * 100, 100);
            const isDone = g.completed;
            const div = document.createElement('div');
            div.className = `p-4 border rounded-2xl bg-white/40 ${isDone ?
'border-[#62836F]/40 bg-[#62836F]/5' : 'border-white/20'}`;
            div.innerHTML = `
                <div class="flex justify-between mb-2">
                    <div><h4 class="font-semibold text-sm text-
[#2D332F]">${isDone ? '🍀' : '🎯'} ${escapeHTML(g.title)}</h4><p class="text-xs
text-gray-400">${g.current.toFixed(2)} из ${g.target.toFixed(2)} 📊</p></div>
                    <span class="text-sm font-bold text-
[#4C6C5A]">${pct.toFixed(0)}%</span>
                </div>
                <div class="w-full bg-[#EAE9E3] h-2 rounded-full overflow-
hidden mb-4"><div class="h-full bg-[#4C6C5A]" style="width: ${pct}%></div></div>
                <div class="flex justify-between items-center">
                    ${!isDone ? `<div class="flex gap-2"><input
type="number" id="fund-${g.id}" class="w-24 px-2 py-1 text-xs border rounded-lg bg-
white"><button onclick="fundGoal(${g.id})" class="bg-[#4C6C5A] text-white text-xs
px-3 py-1.5 rounded-lg cursor-pointer">Внести</button></div>` : `<span class="text-
xs font-bold text-[#62836F] bg-[#62836F]/10 px-2 py-0.5 rounded-
full">Успешно!</span>`}
                    <button onclick="deleteGoal(${g.id})" class="text-
gray-400 hover:text-[#BC7C6C] p-1"><i data-lucide="trash" class="w-4 h-
4"></i></button>
                </div>
            `;
            container.appendChild(div);
        });
        lucide.createIcons();
    }

    document.getElementById('goal-form').addEventListener('submit', async
(e) => {
        e.preventDefault();
        const title = document.getElementById('goal-title').value;
        const target = document.getElementById('goal-target').value;
        await fetch(`${API_URL}/goals`, { method: 'POST', headers:
{ 'Content-Type': 'application/json' }, body: JSON.stringify({ title, target }) });
        document.getElementById('goal-title').value = '';
        document.getElementById('goal-target').value = '';
        loadGoals(); loadData();
    });

    window.fundGoal = async function(id) {
        const amt = document.getElementById(`fund-${id}`).value;
        const res = await fetch(`${API_URL}/goals/${id}/fund`, { method:
'POST', headers: { 'Content-Type': 'application/json' }, body:
JSON.stringify({ amount: amt }) });
        const data = await res.json();
        if (data.is_completed) {
            document.getElementById('congrats-goal-name').textContent =
data.title;
            document.getElementById('congrats-
modal').classList.remove('hidden');
            confetti({ particleCount: 30, spread: 60, origin: { y: 0.8 } });
        }
        loadGoals(); loadData();
    };

```

```

        window.deleteGoal = async function(id) { if(confirm('Видалити ціль?'))
{   await   fetch(`${API_URL}/goals/${id}`,   {   method:   'DELETE'   });   loadGoals();
loadData(); } };

        document.getElementById('clear-all').addEventListener('click', async
() => {
                if(confirm('Очистити все?')) { await fetch(`${API_URL}/clear`,
{ method: 'POST' }); loadData(); }
                });

        function renderCharts() {
                const currentMonth = document.getElementById('ui-month-
selector').value;
                const monthExpenses = localTransactions.filter(t => t.type ===
'expense' && t.date.startsWith(currentMonth));

                const expData = {};
                monthExpenses.forEach(t => { expData[t.category] =
(expData[t.category] || 0) + t.amount; });
                if (expenseChartInstance) expenseChartInstance.destroy();

                if (Object.keys(expData).length > 0) {
                        document.getElementById('no-expense-
data').classList.add('hidden');
                        expenseChartInstance = new
Chart(document.getElementById('expenseChart').getContext('2d'), {
                                type: 'doughnut',
                                data: { labels: Object.keys(expData).map(k =>
displayCategoryName(k)), datasets: [{ data: Object.values(expData), backgroundColor:
Object.keys(expData).map(k => categoryColors[k] || '#8A928E'), borderWidth: 3,
borderColor: '#F4F3EF' }] },
                                options: { responsive: true, maintainAspectRatio: false,
plugins: { legend: { position: 'bottom', labels: { boxWidth: 10, font: { size: 11 },
color: '#2D332F' } } } }
                                });
                        } else { document.getElementById('no-expense-
data').classList.remove('hidden'); }

                if (balanceChartInstance) balanceChartInstance.destroy();
                let cur = 0; const history = [], labels = [];
                localTransactions.forEach((t) => {
                        cur += t.type === 'income' ? t.amount : -t.amount;
                        history.push(cur);
                        labels.push(formatDisplayDate(t.date));
                });

                if (history.length > 0) {
                        balanceChartInstance = new
Chart(document.getElementById('balanceChart').getContext('2d'), {
                                type: 'line',
                                data: { labels: labels, datasets: [{ data: history,
borderColor: '#4C6C5A', backgroundColor: 'rgba(76, 108, 90, 0.06)', fill: true,
tension: 0.2, borderWidth: 3, pointRadius: 2 }] },
                                options: { responsive: true, maintainAspectRatio: false,
plugins: { legend: { display: false } } }
                                });
                }
        }

        function escapeHTML(s) { return s.replace(/&/g, "&amp;").replace(/</g,
"&lt;").replace(/>/g, "&gt;"); }

        updateCategorySelect();
        loadData();

```

```
    </script>
</body>
</html>
```

Додаток Б

Лістинг програмного коду серверної частини (app.py)

```
import os
import sqlite3
from datetime import datetime
from flask import Flask, jsonify, request

app = Flask(__name__)
DATABASE = 'fintrack.db'

def get_db():
    conn = sqlite3.connect(DATABASE)
    conn.row_factory = sqlite3.Row
    return conn

def init_db():
    with get_db() as db:
        db.execute('''
            CREATE TABLE IF NOT EXISTS transactions (
                id INTEGER PRIMARY KEY AUTOINCREMENT,
                category TEXT NOT NULL,
                amount REAL NOT NULL,
                type TEXT NOT NULL,
                date TEXT NOT NULL
            )
        ''')
        db.execute('''
            CREATE TABLE IF NOT EXISTS budgets (
                category TEXT PRIMARY KEY,
                max_limit REAL NOT NULL
            )
        ''')
        db.execute('''
            CREATE TABLE IF NOT EXISTS goals (
                id INTEGER PRIMARY KEY AUTOINCREMENT,
                title TEXT NOT NULL,
                target REAL NOT NULL,
                current REAL DEFAULT 0.0,
                completed INTEGER DEFAULT 0
            )
        ''')

        default_limits = [
            ('Продукти', 5000.0),
            ('Комуналка / Оренда', 4000.0),
            ('Транспорт / Бензин', 1500.0),
            ('Ресторани / Кафе', 2000.0),
            ('Розваги', 2000.0),
            ('Одяг', 2500.0),
            ('Здоров'я', 2000.0),
            ('Інше (Витрата)', 1000.0)
        ]
        for cat, lim in default_limits:
            db.execute('INSERT OR IGNORE INTO budgets (category, max_limit) VALUES
            (?, ?)', (cat, lim))
            db.commit()

@app.after_request
def after_request(response):
    response.headers.add('Access-Control-Allow-Origin', '*')
    response.headers.add('Access-Control-Allow-Headers', 'Content-Type,Authorization')
```

```

        response.headers.add('Access-Control-Allow-Methods',
'GET,PUT,POST,DELETE,OPTIONS')
        return response

@app.get('/api/transactions')
def get_transactions():
    with get_db() as db:
        rows = db.execute('SELECT * FROM transactions ORDER BY id ASC').fetchall()
        return jsonify([dict(r) for r in rows])

@app.post('/api/transactions')
def add_transaction():
    data = request.get_json()
    category = data.get('category')
    amount = float(data.get('amount', 0))
    ttype = data.get('type')
    date_str = datetime.now().strftime('%Y-%m-%d %H:%M')

    with get_db() as db:
        db.execute('INSERT INTO transactions (category, amount, type, date) VALUES
(?, ?, ?, ?)',
                    (category, amount, ttype, date_str))
        db.commit()
    return jsonify({"status": "success"})

@app.delete('/api/transactions/<int:tid>')
def delete_transaction(tid):
    with get_db() as db:
        db.execute('DELETE FROM transactions WHERE id = ?', (tid,))
        db.commit()
    return jsonify({"status": "success"})

@app.post('/api/budgets/update')
def update_budget():
    data = request.get_json()
    category = data.get('category')
    limit_val = float(data.get('limit', 0))
    with get_db() as db:
        db.execute('INSERT OR REPLACE INTO budgets (category, max_limit) VALUES
(?, ?)', (category, limit_val))
        db.commit()
    return jsonify({"status": "success"})

@app.get('/api/goals')
def get_goals():
    with get_db() as db:
        rows = db.execute('SELECT * FROM goals').fetchall()
        return jsonify([dict(r) for r in rows])

@app.post('/api/goals')
def create_goal():
    data = request.get_json()
    title = data.get('title')
    target = float(data.get('target', 0))
    with get_db() as db:
        db.execute('INSERT INTO goals (title, target) VALUES (?, ?)', (title,
target))
        db.commit()
    return jsonify({"status": "success"})

@app.post('/api/goals/<int:gid>/fund')
def fund_goal(gid):
    data = request.get_json()
    amount = float(data.get('amount', 0))
    is_completed = False

```

```

with get_db() as db:
    goal = db.execute('SELECT * FROM goals WHERE id = ?', (gid,)).fetchone()
    if goal:
        new_current = goal['current'] + amount
        completed = 1 if new_current >= goal['target'] else 0
        if completed == 1 and goal['completed'] == 0: is_completed = True
        db.execute('UPDATE goals SET current = ?, completed = ? WHERE id = ?',
(new_current, completed, gid))

        date_str = datetime.now().strftime('%Y-%m-%d %H:%M')
        db.execute('INSERT INTO transactions (category, amount, type, date)
VALUES (?, ?, ?, ?)',
                    ('Кошти на ціль', amount, 'expense', date_str))
        db.commit()
        return jsonify({"status": "success", "is_completed": is_completed,
"title": goal['title']})
    return jsonify({"status": "error"}), 404

@app.delete('/api/goals/<int:gid>')
def delete_goal(gid):
    with get_db() as db:
        db.execute('DELETE FROM goals WHERE id = ?', (gid,))
        db.commit()
    return jsonify({"status": "success"})

@app.post('/api/clear')
def clear_all():
    with get_db() as db:
        db.execute('DELETE FROM transactions')
        db.execute('DELETE FROM goals')
        db.execute('DROP TABLE IF EXISTS budgets')
        db.commit()
    init_db()
    return jsonify({"status": "success"})

@app.get('/api/intelligence')
def get_intelligence():
    month_filter = request.args.get('month', datetime.now().strftime('%Y-%m'))

    with get_db() as db:
        all_txs = db.execute('SELECT * FROM transactions').fetchall()
        budgets_rows = db.execute('SELECT * FROM budgets').fetchall()
        budgets = {b['category']: b['max_limit'] for b in budgets_rows}
        goals_rows = db.execute('SELECT * FROM goals').fetchall()

    month_txs = [t for t in all_txs if t['date'].startswith(month_filter)]

    total_income = sum(t['amount'] for t in month_txs if t['type'] == 'income')
    total_expense = sum(t['amount'] for t in month_txs if t['type'] == 'expense')
    balance = total_income - total_expense

    cats_fact = {}
    for t in month_txs:
        if t['type'] == 'expense':
            cats_fact[t['category']] = cats_fact.get(t['category'], 0.0) +
t['amount']

    deviations = []
    for cat, plan in budgets.items():
        fact = cats_fact.get(cat, 0.0)
        deviations.append({
            "category": cat,
            "plan": plan,
            "fact": fact,

```

```

        "deviation": plan - fact
    })

    forecast = total_expense
    day_of_month = datetime.now().day
    if month_filter == datetime.now().strftime('%Y-%m') and day_of_month > 0 and
total_expense > 0:
        forecast = (total_expense / day_of_month) * 30

    recommendations = []
    notifications = []

    # 1. Порада на основі структури витрат (> 20%)
    has_structure_advice = False
    if total_expense > 0:
        for cat, fact in cats_fact.items():
            percentage = (fact / total_expense) * 100
            if percentage > 20 and cat != 'Кошти на ціль':
                recommendations.append(f"👛 Категорія '{cat}' займає
{percentage:.0f}% ваших витрат. Спробуйте обмежити її, щоб вивільнити більше
коштів для цілей.")
                has_structure_advice = True

    if not has_structure_advice:
        recommendations.append("🍏 Поточний розподіл витрат збалансований, явних
критичних перекосів не виявлено.")

    # 2. Порада на основі правила 50/30/20 (Миттєве відновлення тексту)
    if total_income > 0:
        expense_pct = (total_expense / total_income) * 100
        if expense_pct <= 80:
            free_money = total_income - total_expense
            recommendations.append(f"👛 Чудово! Ви витрачаєте менше 80% доходу.
Вільні {free_money:.0f} ₴ можна направити в інвестиції або цілі.")
        else:
            recommendations.append(f"👛 Ви витрачаєте {expense_pct:.0f}% свого
доходу. Згідно з правилом 50/30/20, рекомендується заощаджувати хоча б 20%
бюджету.")
        else:
            recommendations.append("💡 Порада: Внесіть ваші щомісячні доходи
(Зарплата, Фріланс тощо), щоб увімкнути аналіз пропорцій за правилом 50/30/20.")

    # 3. Аналіз фінансової подушки безпеки
    if balance > 0 and total_expense > 0:
        daily_rate = total_expense / day_of_month if day_of_month > 0 else 1
        days_surv = balance / daily_rate
        recommendations.append(f"🛡️ Фінансова стабільність: Ваш поточний резерв
забезпечує приблизно {days_surv:.0f} днів автономного життя за поточних темпів.")
    elif balance <= 0 and total_expense > 0:
        recommendations.append("🚨 Ваші витрати перевищують доходи. Рекомендується
скоротити необов'язкові витрати на Розваги та Ресторани на 15%.")

    # 4. Аналіз фінансових цілей
    active_goals = [g for g in goals_rows if not g['completed']]
    if active_goals:
        recommendations.append(f"🎯 Для прискорення досягнення цілі
'{active_goals[0]['title']}' спробуйте відкладати додатково 500 ₴ щомісяця.")

    # Психологічні сповіщення для підтримки духу + Сповідання про вичерпання
лімітів
    if total_income > 0 and balance > (total_income * 0.3):
        notifications.append("🚀 Фінансовий зліт! Цього місяця ви заощаджуєте
швидше, ніж зазвичай. Так тримати!")

```

```

overspent_any = any(d['deviation'] < 0 for d in deviations)
if total_expense > 0 and not overspent_any:
    notifications.append("🚨 Залізний контроль! Жодного перевищення лімітів за останні дні. Ви супер-стратег!")

for d in deviations:
    if d['deviation'] < 0:
        notifications.append(f"⚠️ Увага: Бюджет категорії '{d['category']}' вичерпано! Перевищення на {abs(d['deviation']):.0f} ₴.")

return jsonify({
    "forecast": forecast,
    "recommendations": recommendations,
    "notifications": notifications,
    "deviations": deviations,
    "month_income": total_income,
    "month_expense": total_expense
})

@app.route('/api/report/year', methods=['GET'])
def get_year_report():
    current_year = datetime.now().strftime('%Y')
    with get_db() as db:
        txs = db.execute("SELECT * FROM transactions WHERE date LIKE ?",
(f"{current_year}-%",)).fetchall()
        goals = db.execute("SELECT * FROM goals").fetchall()

    months_data = {f"{i:02d}": {"income": 0.0, "expense": 0.0} for i in range(1,
13)}
    for t in txs:
        m = t['date'].split('-')[1]
        if m in months_data:
            if t['type'] == 'income': months_data[m]['income'] += t['amount']
            else: months_data[m]['expense'] += t['amount']

    year_income = sum(m['income'] for m in months_data.values())
    year_expense = sum(m['expense'] for m in months_data.values())
    completed_goals = sum(1 for g in goals if g['completed'])

    return jsonify({
        "year": current_year,
        "months": months_data,
        "year_income": year_income,
        "year_expense": year_expense,
        "completed_goals": completed_goals
    })

if __name__ == '__main__':
    init_db()
    app.run(debug=True, port=5000)

```

Додаток В

Копія опублікованих результатів

Міністерство освіти і науки України
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління



Матеріали

I Міжнародної науково-практичної конференції студентів і молодих вчених
«ICSPM 2026: Intelligent Computing Systems and Project Management /
Інтелектуальні обчислювальні системи та управління проєктами»
21–22 травня 2026 р.



м. Тернопіль - 2026

УДК 004.8:005.8](063)

Присвячено 60-річчю Західноукраїнського національного університету

ICSPM 2026: Інтелектуальні обчислювальні системи та управління проєктами : збірник тез доповідей I Міжнародної науково-практичної конференції студентів і молодих вчених (Тернопіль, 21–22 травня 2026 року). – Тернопіль : ЗУНУ, 2026. – 190 с.

До збірника увійшли тези доповідей учасників I Міжнародної науково-практичної конференції студентів і молодих вчених «ICSPM 2026: Intelligent Computing Systems and Project Management / Інтелектуальні обчислювальні системи та управління проєктами», що відбулася на базі кафедри інформаційно-обчислювальних систем і управління факультету комп'ютерних інформаційних технологій Західноукраїнського національного університету.

Матеріали відображають результати досліджень за такими напрямками:

1. Інтелектуальні обчислення та програмні системи.
2. Проєктно-орієнтоване управління та процеси прийняття рішень.
3. Штучний інтелект, аналітика даних і кібернетика.
4. Прикладні технології та інформаційно-комунікаційні системи.
5. Сучасні інформаційні технології в економіці, праві та освіті.

Рекомендовано до друку на засіданні кафедри інформаційно-обчислювальних систем і управління Західноукраїнського національного університету (протокол №12 від 26.05.2026 р.).

За зміст наукових праць, точність наведених цитувань, перекладу та достовірність фактологічних матеріалів відповідальність несуть автори публікацій. У збірнику збережено авторську стилістику, пунктуацію та орфографію.

© Кафедра інформаційно-обчислювальних систем і управління ЗУНУ, 2026
© Західноукраїнський національний університет, 2026

Хархаліс В.Ю., Турченко І.В.	
ВЕБ-ЗАСТОСУНОК ДЛЯ ІНТЕРВАЛЬНОГО ЗАПАМ'ЯТОВУВАННЯ.....	165
Шабатько А.В., Турченко І.В.	
ВЕБ-ЗАСТОСУНОК ДЛЯ АНАЛІЗУ ТА ВІЗУАЛІЗАЦІЇ ОСОБИСТИХ ФІНАНСІВ	168
 <i>СЕКЦІЯ - СУЧАСНІ ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ В ЕКОНОМІЦІ, ПРАВІ ТА ОСВІТІ / MODERN INFORMATION TECHNOLOGIES IN ECONOMICS, LAW, AND EDUCATION</i>	
I.Tsegelnyy, N.Dziubanovska	
QUALIMETRIC MODEL OF DIGITAL EVALUATION OF INVESTMENT ATTRACTIVENESS OF ECONOMIC RECOVERY PROJECTS IN UKRAINE	173
Гурняк С.С., Биковий П.Є.	
ВЕБ-ЗАСТОСУНОК ДЛЯ КЕРУВАННЯ ПЕРСОНАЛЬНИМИ ФІНАНСАМИ	177
Климчук А.Б., Турченко І.В.	
МЕТОД ПАРАЛЕЛЬНОЇ СЕМАНТИЧНОЇ ОБРОБКИ МЕДІА-РЕСУРСІВ ДЛЯ КОНТЕКСТНО-АДАПТИВНОЇ ГЕНЕРАЦІЇ НАВЧАЛЬНОГО КОНТЕНТУ .	180
Котрубенко В.І., Васильків Н.М.	
ПРОГРАМНА СИСТЕМА ОНЛАЙН-ЗАМОВЛЕНЬ ЮВЕЛІРНИХ ПРИКРАС	184
Овчарук А.І., Васильків Н.М.	
МОДЕЛЮВАННЯ ФІНАНСОВИХ ПРОЦЕСІВ ЗАСОБАМИ EXCEL ТА PYTHON	187

УДК 004.9:336.1

ВЕБ-ЗАСТОСУНОК ДЛЯ КЕРУВАННЯ ПЕРСОНАЛЬНИМИ ФІНАНСАМИ

Гурняк Сніжана Степанівна, студентка,
snizhanahurniak@gmail.com

Науковий керівник: Биковий Павло Євгенович,
к.т.н., доцент, доцент кафедри інформаційно-
обчислювальних систем і управління,
Західноукраїнський національний університет,
pb@wunu.edu.ua,
ORCID: 0000-0002-5705-5702

У роботі представлено результати проектування та реалізації веб-застосунку для керування персональними фінансами. Розглянуто архітектуру системи, алгоритми обробки фінансових транзакцій, механізми бюджетного планування та засоби забезпечення безпеки даних. Особливу увагу приділено інтеграції аналітичних модулів для моніторингу фінансового стану користувача та візуалізації статистичних показників.

Цифровізація фінансового сектору суттєво змінила підходи до управління персональними коштами. Використання кількох банківських рахунків, електронних платіжних сервісів та цифрових гаманців призводить до фрагментації фінансової інформації та ускладнює оперативний контроль витрат і доходів. Традиційні методи ведення обліку, зокрема електронні таблиці або ручна фіксація транзакцій, не забезпечують достатнього рівня автоматизації, аналітичної гнучкості та інтегрованого представлення фінансового стану користувача [1-3].

Сучасні системи Personal Finance Management (PFM) дозволяють автоматизувати процеси фінансового моніторингу, однак значна частина наявних рішень характеризується закритою архітектурою, обмеженим функціоналом безкоштовних версій або високою складністю інтерфейсу. Це знижує доступність таких платформ для широкого кола користувачів та обмежує можливість персоналізованого аналізу фінансової активності [4].

Метою дослідження є розробка веб-застосунку для управління персональними фінансами, який поєднує адаптивний інтерфейс, ефективні механізми збереження даних та алгоритмічні засоби аналізу фінансових операцій. Особливістю запропонованого рішення є поєднання механізмів бюджетного планування, інтерактивної фінансової аналітики та модулів моніторингу фінансових цілей у межах єдиного веб-орієнтованого середовища.

Архітектура системи реалізована за принципом багаторівневої моделі та включає клієнтський інтерфейс, серверну частину та реляційну базу даних. Такий підхід забезпечує масштабованість системи, спрощує інтеграцію

додаткових функціональних модулів та підвищує надійність роботи програмного забезпечення [5].

Серверна логіка реалізована мовою Python, що дозволяє ефективно обробляти фінансові транзакції, виконувати перевірку коректності даних та формувати статистичні показники в реальному часі. Для збереження інформації використано систему керування базами даних PostgreSQL, яка підтримує принципи ACID та забезпечує цілісність фінансових записів [4, 6].

Розроблена система забезпечує ведення обліку доходів і витрат користувача, підтримує категоризацію фінансових операцій та дозволяє працювати з декількома рахунками одночасно. Функціональність застосунку передбачає встановлення бюджетних лімітів для окремих категорій витрат, моніторинг фінансових цілей користувача, а також автоматичне формування статистичних звітів і графічної аналітики для оцінювання фінансового стану.

Алгоритм додавання транзакцій передбачає перевірку валідності введених даних, фіксацію запису в базі даних та автоматичне оновлення фінансової статистики. Після успішного виконання операції система перераховує поточний баланс рахунку та оновлює аналітичні індикатори користувача.

Для взаємодії між клієнтською та серверною частинами використано REST API [7]. Передача даних здійснюється асинхронно, що дозволяє мінімізувати затримки при оновленні інтерфейсу та забезпечує високу швидкість системи навіть при значній кількості транзакцій. На рисунку 1 представлено послідовність взаємодії між користувачем, клієнтським веб-інтерфейсом, серверною частиною та базою даних під час додавання фінансової операції. Діаграма демонструє асинхронний обмін даними та механізм оновлення статистики після успішного збереження транзакції.

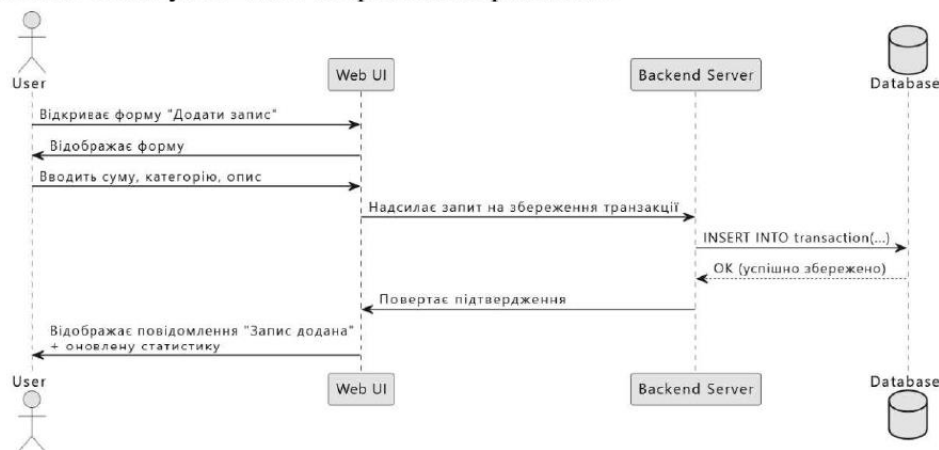


Рисунок 1 – Діаграма послідовності обробки фінансових транзакцій

Окремим компонентом системи є модуль фінансових цілей. Користувач може задавати бажану суму накопичення та часовий інтервал її досягнення. Прогрес виконання цілі визначається як співвідношення поточних накопичень

до цільового значення та динамічно оновлюється після кожної зміни фінансового стану.

Для забезпечення безпеки персональних даних реалізовано механізми JWT-авторизації та хешування паролів. Використання токенів доступу дозволяє захистити API від несанкціонованих запитів, а збереження паролів у вигляді хешів відповідає сучасним вимогам інформаційної безпеки [4].

Для оцінки ефективності запропонованого рішення проведено порівняльний аналіз із популярними системами управління персональними фінансами. Результати дослідження показали, що розроблений веб-застосунок забезпечує базовий набір функцій, необхідних для персонального фінансового моніторингу, підтримує інтерактивний аналіз фінансових даних та не містить рекламного навантаження, характерного для багатьох безкоштовних платформ.

Практичне значення отриманих результатів полягає у створенні доступного інструменту для автоматизації фінансового обліку та підтримки прийняття рішень під час планування персонального бюджету. Розроблений підхід може бути використаний як основа для подальшого розвитку інформаційних систем фінансового моніторингу.

Перспективи подальших досліджень полягають у впровадженні модулів прогнозування витрат на основі аналізу часових рядів, інтеграції з банківськими API та розширенні функціональності системи засобами персоналізованих фінансових рекомендацій.

Список використаних джерел

1. Agarwal, V., Ray, R., & Varghese, N. A privacy-driven data management model for smart personal assistants. *ResearchGate*. 2024. <https://www.researchgate.net/publication/317172126>.
2. Prakash, J., Sharma, V., Patel, N. Personal finance transformation in the digital era: The impact of financial technology on people's economic behavior. *ResearchGate*. 2024. <https://www.researchgate.net/publication/392776574>.
3. Gomber, P., Koch, J.-A., Siering, M. Digital finance and FinTech: Current research and future research directions. *Journal of Business Economics*, 2017. Vol. 87. P. 537–580. <https://doi.org/10.1007/s11573-017-0852-x>
4. Kumar, A., Gupta, S. Security and privacy issues: A survey on FinTech. *ResearchGate*. 2020. <https://www.researchgate.net/publication/312326195>.
5. Bashovyi, V. M., Statsenko, V. V., Statsenko, D. V. Development of adaptive web interface for personal finance management application. *Technologies and Engineering*, 2022. Vol.5. <https://doi.org/10.30857/2786-5371.2022.5.1>.
6. Elmasri, R., Navathe, S. B. *Fundamentals of Database Systems* (7th ed.). Pearson, 2016.
7. Fielding, R. T. *Architectural styles and the design of network-based software architectures*. University of California, Irvine, 2000.