Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет центр післядипломної освіти
(повна назва)

Кафедра програмної інженерії
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Програмна система управління фінансами домогосподарства

(тема)

Виконав:
студент 2 курсу, групи ПЗПІп-22-2

Іванічев В.О.
(прізвище, ініціали)

Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Програмна інженерія
(повна назва освітньої програми)

Керівник доц. кафедри ПІ Лановий О.Ф.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри _____ З.В.Дудар
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет центр післядипломної освіти
Кафедра програмної інженерії
Рівень вищої освіти перший (бакалаврський)
Спеціальність 121 – Інженерія програмного забезпечення
Тип програми Освітньо-професійна
Освітня програма Програмна Інженерія
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«___» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Іванічеву Володимиру Олександровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Програмна система управління фінансами домогосподарства

Затверджена наказом по університету від 17.06. 2024р. № 588 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 19.07.2024

3. Вихідні дані до роботи Розробити програмну систему управління фінансами домогосподарства, яка дозволяє отримувати та зберігати дані про витрати та доходи домогосподарства .

4. Перелік питань, що потрібно опрацювати в роботі

Вступ, аналіз предметної галузі, формування вимог до програмної системи, архітектура та проектування програмного забезпечення, опис прийнятих програмних рішень, тестування розробленого програмного забезпечення, висновки, додатки .

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз предметної галузі	18.06.2024	*виконано*
2	Створення специфікації ПЗ	20.06.2024	*виконано*
3	Проектування ПЗ	24.06.2024	*виконано*
4	Розробка ПЗ	26.06.2024	*виконано*
5	Тестування ПЗ	01.07.2024	*виконано*
6	Оформлення пояснювальної записки	05.07.2024	*виконано*
7	Підготовка презентації та доповіді	12.07.2024	*виконано*
8	Попередній захист	13.07.2024	*виконано*
9	Нормоконтроль, рецензування	15.07.2024	*виконано*
10	Здача роботи у електронний архів	17.07.2024	*виконано*
11	Допуск до захисту у зав. кафедри	19.07.2024	*виконано*

Дата видачі завдання 18 червня 2024р.

Студент (ка)_____ Іванічев В.О.
(підпис)

Керівник роботи _____ доц. кафедри ПІ Лановий О.Ф.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Пояснювальна записка до кваліфікаційної роботи бакалавра, 67 стор., 30 рис., 1 табл., 17 джерел.

МОВА ПРОГРАМУВАННЯ JAVASCRIPT, РЕЛЯЦІЙНА БАЗА ДАНИХ, СИСТЕМА УПРАВЛІННЯ БАЗАМИ ДАНИХ, ANGULAR, MYSQL SERVER, NODE.JS

Об’єкт розробки – система управління фінансами домогосподарства.

Мета розробки – розробка веб-додатку «Програмна система управління фінансами домогосподарства» з використанням реляційних баз даних та логіки об’єктно-орієнтованого програмування.

Метод рішення – середа розробки Webstrom, фреймворк Angular та Express.js, мови програмування Javascript та Typescript а також система управління базами даних My SQL Server.

В результаті роботи був створений веб-додаток, який дозволяє отримувати та зберігати дані про витрати та доходи домогосподарства. Також програма дозволяє шукати, фільтрувати інформацію, отримувати статистики по обігу грошових коштів, підтримує задачу автоматизації з формування бюджету домогосоподарства по фактичним витратам.

JAVASCRIPT PROGRAMMING LANGUAGE, RELATIONAL DATABASE, DATABASE MANAGEMENT SYSTEM, ANGULAR, MYSQL SERVER, NODE.JS

The object of development is a household finance management system.

The purpose of the development is to develop a web application "Household Finance Management Software System" using relational databases and object-oriented programming logic.

The solution method is the Webstrom development environment, the Angular and Express.js frameworks, the Javascript and Typescript programming languages, and the My SQL Server database management system.

As a result of the work, a web application was created that allows you to receive and store data on household expenses and income. Also, the program allows you to search and filter information, receive statistics on the circulation of money, supports the task of automating the formation of a household budget based on actual expenses.

Я, Іванічев Володимир Олександрович, студент гр. ПЗППп-22-2, здобувач вищої освіти на першому (бакалаврському) рівні кафедри «Програмна інженерія», заявляю: моя кваліфікаційна робота на тему «Програмна система управління фінансами домогосподарства», що буде представлена до екзаменаційної комісії для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElAr KhNURE. Усі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлений з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови до допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

СУБД – Система Управління Базами Даних

HTTP – Hypertext Transfer Protocol

SDK – Serial Development Kit

ВСТУП

На поточному етапі розвитку фінансової системи України одним із головних пріоритетів є удосконалення та розвиток фінансів домогосподарств. Фінанси домогосподарств постійно взаємодіють з іншими сегментами фінансової системи держави, що об'єктивно сприяє створенню необхідних умов для подальшого розвитку як фінансового, так і нефінансового секторів економіки, формуванню та розподілу валового внутрішнього продукту (ВВП), а також державних і місцевих бюджетів.

Особливо важлива роль домогосподарств в умовах військового стану та економічної нестабільності, коли їхні раціональні рішення та поведінка безпосередньо впливають на стабільність основних параметрів рівноваги соціально-економічної та фінансової системи. Вони визначають напрями розподілу доходів і їх використання, а також формування заощаджень та їх інвестиційне спрямування [1].

Сума доходу, необхідного для повного покриття витрат домогосподарства, залежить від потреб його членів, які можуть задовольнятися як матеріальними благами, так і набором послуг. Фінансові ресурси, що здобуваються окремими членами домогосподарства, можуть тимчасово належати індивіду, але згодом використовуються для задоволення потреб усіх членів родини [2].

Координація економічних інтересів усіх членів домогосподарства є важливим завданням для ефективного функціонування фінансів [3]. Завдяки фінансовим ресурсам, домогосподарства можуть перерозподіляти загальний дохід між своїми членами, а також змінювати частку доходу, що припадає на кожного конкретного члена родини.

Темою кваліфікаційної роботи є розробка додатку, який дозволяє здійснювати ефективне управління фінансами домогосподарства та надає користувачам інтуїтивно зрозумілий інструмент для контролю над їхнім бюджетом та фінансовим станом. Основні функціональні можливості додатку включають ведення обліку доходів та витрат, аналіз фінансового стану, планування бюджету. Також програма дозволяє шукати, фільтрувати інформацію, отримувати

статистики по обігу грошових коштів, підтримує задачу автоматизації з формування бюджету домогосподарства по фактичним витратам.

Додаток є інтуїтивно зрозумілим і легко адаптованим, тому будь-яка особа, незалежно від рівня досвіду, зможе легко користуватися ним та швидко адаптуватися до його функціоналу.

В ході створення веб-додатку були використані середа розробки Webstrom, фреймворк Angular та Express.js, мови програмування Javascript та Typescript а також система управління базами даних MySQL Server.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Аналіз предметної галузі

Домогосподарство – це група людей, які живуть разом і спільно управляють своїми фінансами та ресурсами, щоб задовольнити свої потреби і забезпечити комфортне життя. Термін "домогосподарство" може використовуватися для опису родини, побуту або сім'ї, які проживають разом і діляться витратами та обов'язками [4].

Таким чином, важливим аспектом діяльності домогосподарства є контроль та планування витрат. Контроль витрат допомагає домогосподарству ефективно використовувати свої фінансові ресурси і уникати непотрібних витрат. Планування витрат дозволяє заздалегідь визначити, на що будуть витрачені кошти, і створити бюджет, який враховує поточні та майбутні фінансові потреби домогосподарства.

Розрізняють три види бюджету домогосподарства: спільний, частковий і роздільний[3].

1) Спільний бюджет. Це найпоширеніший вид бюджету домогосподарства. При такому способі розподілу коштів, усі доходи, зароблені членами домогосподарства, складаються разом, і потім подружжя спільно вирішує, як розподілити отриману суму на певний проміжок часу (зазвичай - на місяць). Найбільший плюс такого підходу - у відчутті єдності. Чоловік та дружина разом обговорюють майбутні витрати, разом відповідають за розрахунок коштів. Спільний тип бюджету, або «загальний гаманець», зазвичай використовують подружжя з приблизно рівними доходами або пари, де дружина частково або повністю знаходиться на утриманні чоловіка. Цей варіант практично неминучий у випадку, коли жінка повністю присвячує себе догляду за дитиною, а єдиним годувальником залишається чоловік. Тобто фактично бюджет стає одноосібним, але психологічно він все ж є спільним - гроші лежать у певному місці, подружжя разом вирішує як ними розпорядитися. Основа такого підходу в довірі один до одного, взаємній відповідальності та вмінні знаходити компроміс.

2) Частковий бюджет (Спільно-роздільний). Спільно-роздільний бюджет наразі набуває все більшої актуальності. Цей принцип працює найкраще у випадку, якщо різниця між зарплатами подружжя незначна. Для цього спочатку потрібно розрахувати, скільки грошей домогосподарство витрачає щомісяця на харчування, комунальні платежі, господарські витрати та інші потреби. Далі ця сума розподіляється між членами домогосподарства або порівну, або у співвідношенні, яке домогосподарство вважає справедливим, залежно від зарплати. Таким чином, у кожного залишаються особисті гроші, які можна витратити на свій розсуд. Позитивна сторона такого планування полягає в унікальному поєднанні відчуття спільності в домогосподарстві (як і у випадку «загального гаманця») та елементу фінансової незалежності один від одного. Частковий бюджет - досить універсальний і підходить майже всім, але тільки за умови, що обоє подружжя працюють.

3) Роздільний бюджет. Роздільний бюджет, як такий, рідко застосовується в чистому вигляді. Такий тип розподілу грошей більше прийнятий серед пар, у яких обоє подружжя мають достатньо високий дохід. Кожен забезпечує себе сам тим, у чому має потребу, гроші при цьому, як правило, знаходяться на різних банківських рахунках а їжа купується спільно. Деякі пари, що ведуть роздільний бюджет, просто рахують, скільки грошей у них йде на їжу щомісяця, і розділяють цю сумму порівну. Коли в когось одного гроші закінчуються, він позичає у другого, з умовою обов'язкового повернення боргу. Плюси такого типу бюджету в матеріальній незалежності один від одного, яка допомагає уникати конфліктів на фінансовому ґрунті і дає можливість кожному планувати свої придбання, ні перед ким не звітуючи.

Планування бюджету домогосподарства – це прогнозування змін доходів і витрат домогосподарства на наступний період, визначення організаційно-економічних та фінансових заходів щодо збалансованості доходів і витрат, отримання та ефективного використання накопичень домогосподарства[2].

Планування бюджету домогосподарства здійснюється в такому порядку:

1. прогнозування доходів домогосподарства;

2. прогнозування витрат домогосподарства;

3. співставлення майбутніх доходів і витрат, їх балансування та регулювання шляхом пошуку додаткових джерел доходів і визначення заходів щодо скорочення витрат домогосподарства;

4. визначення та розподіл очікуваних накопичень домогосподарства.

Усі накопичення домогосподарства за своїм призначенням можна розділити на резерви непередбачених витрат і цільові планові накопичення.

Резерви непередбачених витрат домогосподарства включають в себе резерв непередбачених поточних витрат і резерв компенсації втрат від нещасних випадків. Резерв непередбачених поточних витрат призначається для покриття незапланованих витрат, викликаних несподіваним підвищенням цін на споживчі товари, аварійними ремонтами та заміною вибулого домашнього майна, придбанням необхідних речей та іншими поточними витратами, не передбаченими у витратній частині бюджету домогосподарства. Резерв компенсації втрат від нещасних випадків формується для оплати непередбачених витрат, викликаних тривалими хворобами або смертю членів домогосподарства, стійкою втратою їх працездатності, безробіттям, ліквідацією наслідків стихійних лих, пожеж та інших нещасних випадків. Розмір цього резерву залежить від складу домогосподарства, місць роботи, віку та стану здоров'я членів домогосподарства, від підвищеності регіону до землетрусів, повеней та інших стихійних лих. При визначенні планової величини цього резерву слід мати на увазі ті державні допомоги та інші страхові виплати, які будуть належати домогосподарству у випадку зазначених нещасних випадків.

Цільові планові накопичення складаються з короткострокових і довгострокових накопичень, що різняться термінами, розмірами і своєю значимістю для домогосподарства [5]. Цільові короткострокові накопичення призначені для фінансового забезпечення вирішення тактичних завдань домогосподарства щодо придбання нових, додаткових речей та інших витрат, не включених до поточних витрат домогосподарства, і для яких потрібні значні

накопичення грошових коштів протягом терміну, що перевищує період поточного плану.

Цільові довгострокові накопичення призначені для фінансового забезпечення досягнення стратегічних цілей домогосподарства, що вимагають великих довгострокових накопичень протягом декількох років (на придбання автомобіля, дорогої сільськогосподарської та іншої техніки, на придбання або будівництво житла, дачі тощо). Розміри і терміни всіх цільових планових накопичень залежать від величини необхідних витрат, що забезпечують досягнення поставленої мети, і від матеріальних можливостей домогосподарства [6].

Контроль боргів, накопичень та управління кредитами також є важливими аспектами фінансового планування та управління домогосподарством. Давайте розглянемо кожен з цих аспектів окремо.

Контроль боргів: Борги можуть включати кредитні картки, позики, іпотеку, автомобільні кредити та інші зобов'язання, які домогосподарство має перед кредиторами. Контроль боргів передбачає регулярний моніторинг стану боргу, включаючи розмір боргу, відсоткову ставку, строк погашення та мінімальні щомісячні платежі. Важливо дотримуватись розрахункових термінів та розуміти загальну суму боргу, щоб уникнути занадто великого навантаження на фінансовий бюджет домогосподарства.

Контроль накопичень: Накопичення грошей важливо для фінансової стабільності домогосподарства і може включати створення екстреного фонду, пенсійного накопичення, освітніх фондів та інших фінансових цілей. Контроль накопичень передбачає визначення цілей накопичень, встановлення мінімальних щомісячних внесків, моніторинг стану накопичень та використання стратегій зростання, таких як інвестування.

Управління кредитами: Управління кредитами охоплює раціональне використання кредитних можливостей та ефективне погашення кредитів. При виборі кредиту важливо аналізувати умови кредитного договору, включаючи відсоткову ставку, строк кредиту та комісійні платежі. Розуміння погашення

кредиту та відповідне планування платежів допоможе уникнути просрочень та зберегти гарну кредитну історію.

Всі ці аспекти вимагають дисципліни, розуміння основних фінансових понять та уміння вести точний фінансовий облік.

Інформаційні потреби домогосподарства зазвичай пов'язані з ефективним управлінням фінансами, бюджетуванням, плануванням витрат і контролем над фінансовими ресурсами. Основні інформаційні потреби домогосподарства включають:

Фінансовий стан: Домогосподарство потребує інформації про свій фінансовий стан, яка включає загальну суму доступних коштів, банківський рахунок, інвестиції, зобов'язання та борги. Ця інформація допомагає визначити фінансові можливості та обмеження домогосподарства.

Бюджетування: Домогосподарство має інформацію про свій місячний або річний бюджет. Це включає інформацію про доходи, витрати, розподіл коштів на основні категорії (наприклад, їжа, житло, транспорт, освіта) та планування резерву на непередбачені витрати.

Витрати: Домогосподарство має інформацію про свої витрати, що дозволяє контролювати та аналізувати, на що саме витрачаються кошти.

1.2 Аналіз існуючих аналогів

На ринку існує кілька додатків та платформ, призначених для управління фінансами домогосподарства. Перед розробкою даного додатку було проведено докладний аналіз існуючих аналогів з метою визначення їхніх переваг та недоліків [7].

Mint – це інтегрована фінансова веб-платформа, доступна через веб-браузер, яка автоматично вивантажує транзакції з банківських рахунків, надає гнучкий інструмент бюджетування та графіки витрат для зручного управління особистими фінансами. Завдяки своїй забезпеченій системі та функціоналу, Mint допомагає користувачам ефективно вести облік та аналізувати свої фінанси.

Переваги:

- інтегрована забезпечена система обліку та аналізу фінансів;
- автоматичне вивантаження транзакцій з банківських рахунків;
- гнучкий бюджетний інструмент та графіки витрат.

Недоліки:

- залежність від підтримки конкретних банків;
- можливість затримок у вивантаженні транзакцій.

You Need A Budget (YNAB) – це програма для планування бюджету, спрямована на те, щоб допомагати користувачам ефективно розподіляти свої фінансові ресурси, приводячи до більшого контролю над власними витратами та фінансовим станом. З центральним акцентом на категоризації та прозорості, YNAB сприяє формуванню фінансової дисципліни та відповідальності. You Need A Budget є веб-платформою, а також має додатки для різних операційних систем, включаючи Windows, macOS, iOS та Android. Користувачі можуть використовувати YNAB як через веб-браузер, так і завантажити його додатки для використання на різних пристроях[8].

Переваги:

- орієнтована на планування та контроль над кожним доларом;
- інтерактивний бюджет та освітні матеріали щодо фінансової грамотності;
- можливість працювати з ручним введенням транзакцій.

Недоліки:

- платна підписка;
- можливість вимагати більше часу для введення та аналізу даних.

PocketGuard – це мобільний додаток для управління фінансами, який дозволяє користувачам легко вести облік своїх доходів та витрат, надаючи швидкий огляд їхнього фінансового стану. Забезпечуючи інтерактивний аналіз та автоматичні рекомендації, PocketGuard допомагає користувачам ефективно керувати своїм бюджетом та забезпечує простий і зручний інтерфейс для використання на різних мобільних пристроях [9].

Переваги:

- швидке і просте встановлення та використання;

- автоматичний аналіз фінансового стану та рекомендації;
- можливість працювати з різними фінансовими інструментами.

Недоліки:

- обмежені можливості у розширенні функціоналу;
- не завжди точна інтерпретація деяких транзакцій.

Аналіз існуючих аналогів показав, що на ринку вже існують ефективні рішення для управління фінансами. Однак наш додаток ставить своїми основними перевагами легкість використання, швидкість пристосування та високий рівень інтуїції, щоб забезпечити максимальну доступність та комфорт користувачів будь-якого рівня досвіду.

1.3 Постановка задачі

Необхідно спроектувати базу даних для збереження інформації стосовно доходів та витрат домогосподарства, та створити інформаційну систему, що буде забезпечувати наступні функції:

1) система повинна відображати дані:
    1) безпосередньо про основні поняття предметної області (ПО): списки статей доходів та видатків з ієрархією, список бюджетів, гаманців та рахунків користувача, список валют;
    2) про пов’язані поняття ПО: інформацію про отримані доходи і видатки, інформацію про заплановані доходи і видатки відповідно до бюджету;
2) система повинна підтримувати арифметичну обробку даних у вигляді обчислювальних полів: стосовно залишків грошових коштів на гаманцях;
3) система повинна підтримувати сортування, пошук та фільтрацію даних:
    1) сортувати прибутки та видатки за датою операції;
    2) здійснювати пошук статті прибутку та статті видатку за повною або частковою назвою;
    3) здійснювати фільтрацію операції доходу чи видатку по даті операції та гаманцем;

4) система повинна підтримувати додавання нових даних про користувачів, гаманці, бюджети, прибутки та видатки;

5) система повинна підтримувати можливості редагування інформації про користувачів, гаманці, бюджети, прибутки та видатки;

6) система повинна підтримувати можливості вилучення інформації про прибутки та видатки з підтримкою режиму підтвердження користувачем видалення інформації про поточний об’єкт;

7) система повинна підтримувати формування наступних статистик:

1) отримати статистику структури прибутків за обраний період з урахуванням ієрархії статей витрат:
   - стаття доходів;
   - дата витрати;
   - сумма операції;
2) отримати статистику структури витрат за період з урахуванням ієрархії статей витрат:
   - группа витрат;
   - стаття витрат;
   - дата витрати;
   - сумма операції;
3) отримати статистику по операціям доходу та видатку по гаманцю:
   - валюта;
   - гаманець;
   - сумма операції;
4) отримати статистику по операціям доходу та видатку по користувачу:
   - користувач;
   - дата операції;
   - сумма операції;

8) система повинна підтримувати можливість формування довільного запиту до БД на мові SQL;

9) система повинна підтримувати підготовку та друк наступних звітів:

1) фінансовий стан помісячно (зміна фінансового стану домогосподарства згідно курсу валюти):
   - звітний період;
   - валюта;
   - сума коштів та заощаджень на кінець періоду;
2) аналіз план-факт бюджету (порівняння запланованих та фактично здійснених операцій доходів та видатків):
   - бюджетний період;
   - стаття доходів/витрат;
   - сума запланованих операцій;
   - сума фактичних операцій;

10) система повинна реалізовувати наступні задачі автоматизації:
    1) формування бюджету на підставі статистики доходів та витрат в попередніх періодах.

2 ФОРМУВАННЯ ВИМОГ ДО ПРОГРАМНОЇ СИСТЕМИ

Для побудови програмної системи управління фінансами домогосподарства розроблено концепт-документ, який наведено у даному розділі [10].

Опис функціонального призначення ІС, що створюється наведено на рисунку 2.1.

Опис концептів програмної області, їх властивостей та зв'язків між ними зобразимо у вигляді загальної діаграми класів представлено на рисунку 2.2.

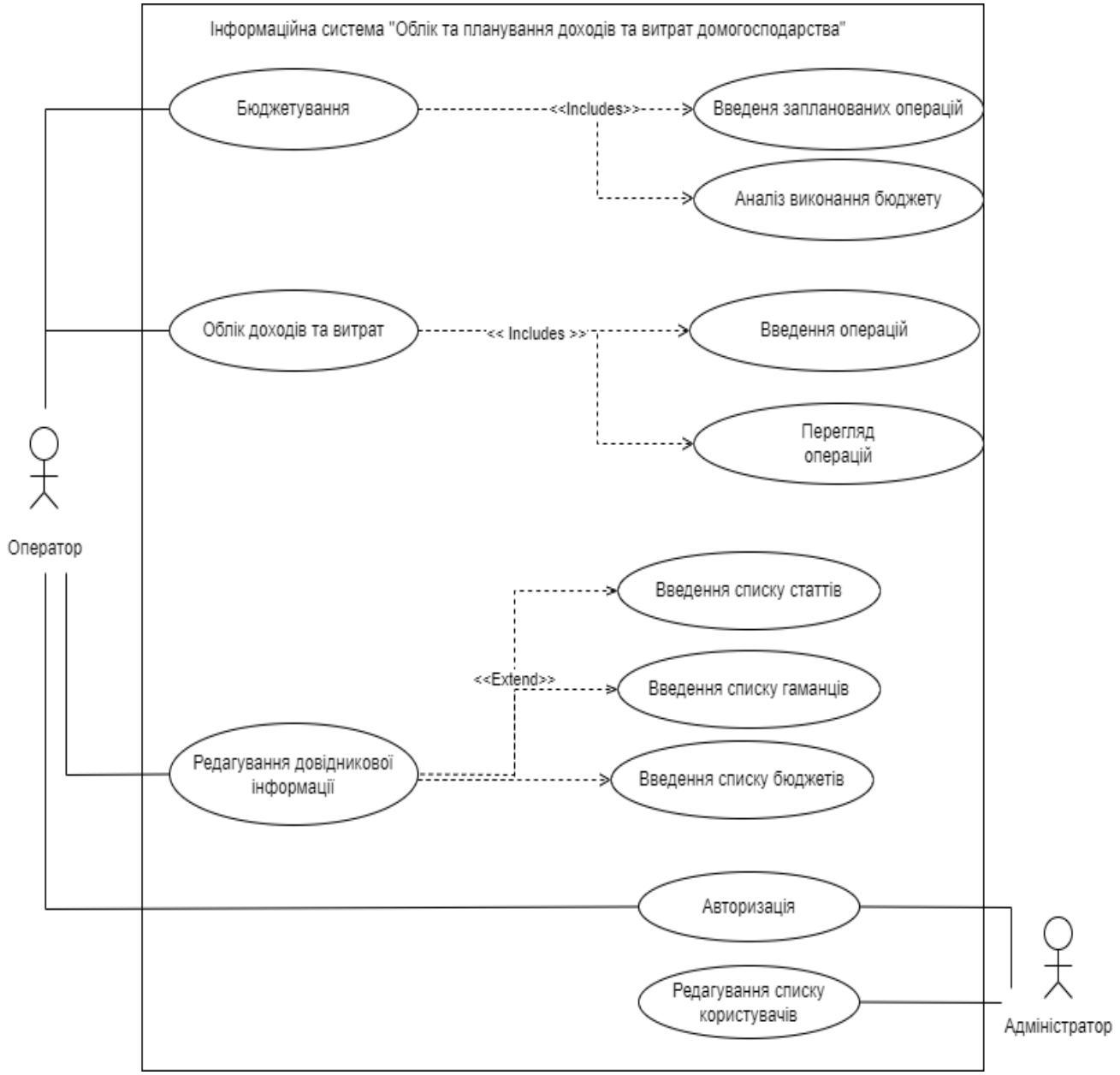


Рисунок 2.1 – Загальна USE-CASE діаграма

Рисунок 2.2 – Загальна діаграма класів

Зазначимо перелік характеристик для кожного з наведених понять.

Поняття “Гаманець” має в собі інформацію про власника в виді поняття “Користувач” та інформацію про валюту гаманця у вигляді поняття “Валюта”. “Користувач” має ім’я, “Валюта” має інформацію про назву.

Поняття “Стаття доходу/витрати” має ієрархічну структуру, і відповідно назву та тип статті. Поняття “Операція” має назву, тип, інформацію про гаманець, дату проведення та містить в собі список статей та суми по ним.

Поняття “Бюджет” має атрибути про дату початку та завершення періоду. Зв’язок між бюджетом та статтями доходів та витрат формують заплановані витрати/доходи в періоді

Для спрощення роботи користувачів під час роботи з інформацією вони повинні мати можливість:

- сортувати операції за датою, сумою, типом операції операції;

– здійснювати пошук інформації про операцію за назвою статей що включені в операцію;

– здійснювати фільтрацію інформації по гаманцям, валютам.

Користувачі для мають мати можливість отримання наступної статистики:

– отримати статистику структури прибутків за обраний період з урахуванням ієрархії статей витрат;

– отримати статистику структури витрат за період з урахуванням ієрархії статей витрат;

– отримати статистику по операціям доходу та видатку по гаманцю;

– отримати статистику по операціям доходу та видатку по користувачу.

Схематично зобразимо інформаційні потреби користувача (див. рис. 2.3).

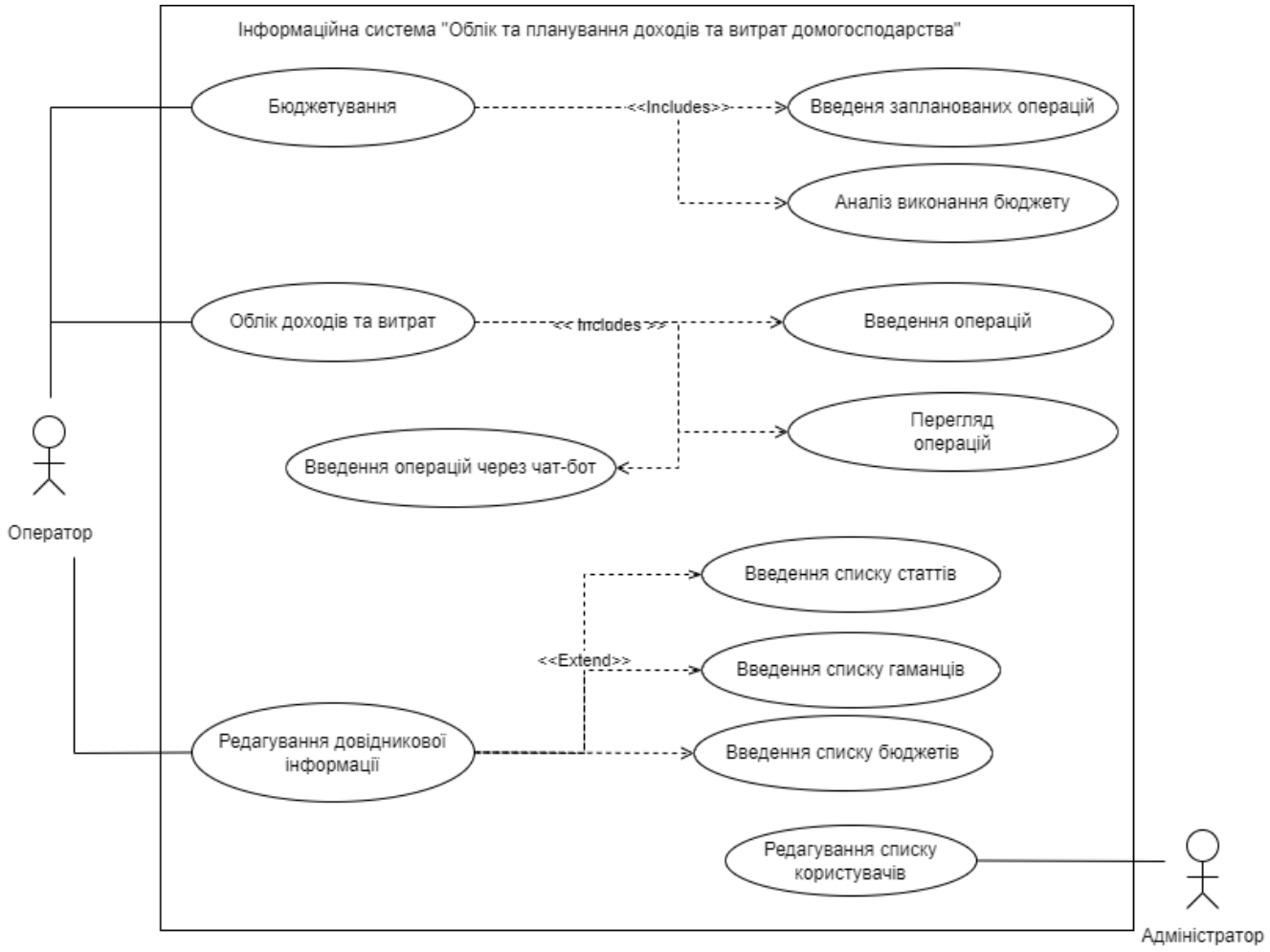


Рисунок 2.3 – Детальна USE-CASE діаграма для опису інформаційних потреб

Інформацію про операції можно вносити за допомогою чат-боту, то необхідна автоматизація з імпорту цієї інформації в БД. Також існує потреба в формуванні бюджету на підставі існуючої інформації з попередніх періодів, оскільки інформаційна база містить інформацію про заплановані та фактичні доходи та витрати в минулих періодах.

Таким чином, формування нового бюджету повинно проходити автоматично з можливістю подальшого корегування їх користувачем Бюджет після формування має бути затверджений користувачем, але в поточному періоді користувач має мати можливість корегування. Також має бути реалізовано оповіщення користувача у випадку якщо бюджет перевищено.

Запропоновану автоматизацію зображено за допомогою діаграми потоків даних (див. рис. 2.4). Це може бути або діаграма послідовностей, або діаграма видів діяльності, або діаграма станів.

Описання алгоритмічних залежностей показників в обліку складається з наступних залежностей:

- залишок коштів на гаманців дорівнює різниці між надходженням коштів до гаманця та витратами з гаманця;
- сума доходів або витрат за статтею в періоді дорівнює сумі підпорядкованих статей в періоді.

В проблемній області існує рід обмежень, які можна віднести до обмежень цілісності стосовно ідентифікації:

- для ідентифікації валют потрібно використовувати міжнародний класифікатор валют;
- для інших сутностей в інформаційній системі треба використовувати сурогатні ідентифікатори.

В проблемній області існує рід обмежень, які можна віднести до обмежень цілісності стосовно зв’язків:

Рисунок 2.4 – Діаграма станів роботи з бюджетом

- у користувача може бути декілька гаманців;
- гаманець зберігає кошти в одній валюті;
- операція може містити декілька статей доходів/витрат;

- бюджети не мають пересікатись між собою з урахуванням їх дат початку та завершення.

З урахуванням ліцензійного ПЗ, необхідно дотримуватися наступних обмежень:

- для реалізації функцій чат-боту використовувати месенджер Telegram;
- використовувати СУБД MySql.

Лінгвістичні відносини, що існують в ПЗ:

бюджет – період планування доходів та затрат в домогосподарстві стаття витрат – це окрема категорія витрат, яка використовується для групування і класифікації витратних позицій в бюджеті. Кожна стаття затрат представляє собою конкретну категорію витрат, наприклад, їжа, транспорт, освіта, розваги тощо. Використання статей затрат допомагає вести облік і контроль за розподілом та витратами грошових коштів у домогосподарстві.

Гаманець: це фізичний або електронний засіб для зберігання грошових коштів. У контексті домогосподарства, гаманець використовується для управління грошовими потоками та витратами. Кожен гаманець може бути пов'язаний з конкретним бюджетом або статтею затрат, що дозволяє відстежувати, на що саме витрачаються гроші.

3 АРХІТЕКТУРА ТА ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Побудова ER-діаграми

ER-діаграма (діаграма сутностей-зв'язків) є потужним інструментом для моделювання взаємозв'язків між різними сутностями у базі даних. У контексті фінансового додатку для управління домогосподарством, побудова ER-діаграми є ключовим етапом для визначення структури даних та їхніх взаємозв'язків.[11]

Першим кроком є визначення сутностей, які представляють об'єкти або поняття, що мають важливість для системи. У фінансовому додатку можливі такі сутності:

Користувач: з атрибутами, такими як ім'я, електронна адреса та пароль для авторизації.

Операція: з атрибутами, такими як сума, дата та тип (дохід чи витрата).

Стаття: для класифікації транзакцій, наприклад, їжа, транспорт, розваги.

Бюджет: для класифікації періодів планування

Визначаємо взаємозв'язки між сутностями.

Гаманець може мати багато операцій, але кожна операція належить лише одному гаманцю (відношення один-до-багатьох).

Операції можуть відноситися до багатьох статей, і кожна стаття може мати багато операцій (відношення багато-до-багатьох).

Далі визначаємо ключі для кожної сутності. У сутності "Користувач" ключем може бути ідентифікатор користувача (UserID).

На даному етапі проектування БД доцільно за основу взяти загальну діаграму класів, яку наведено у розділі 2. Було розроблена ER-діаграма за нотацією Баркера (див. рис. 3.1).

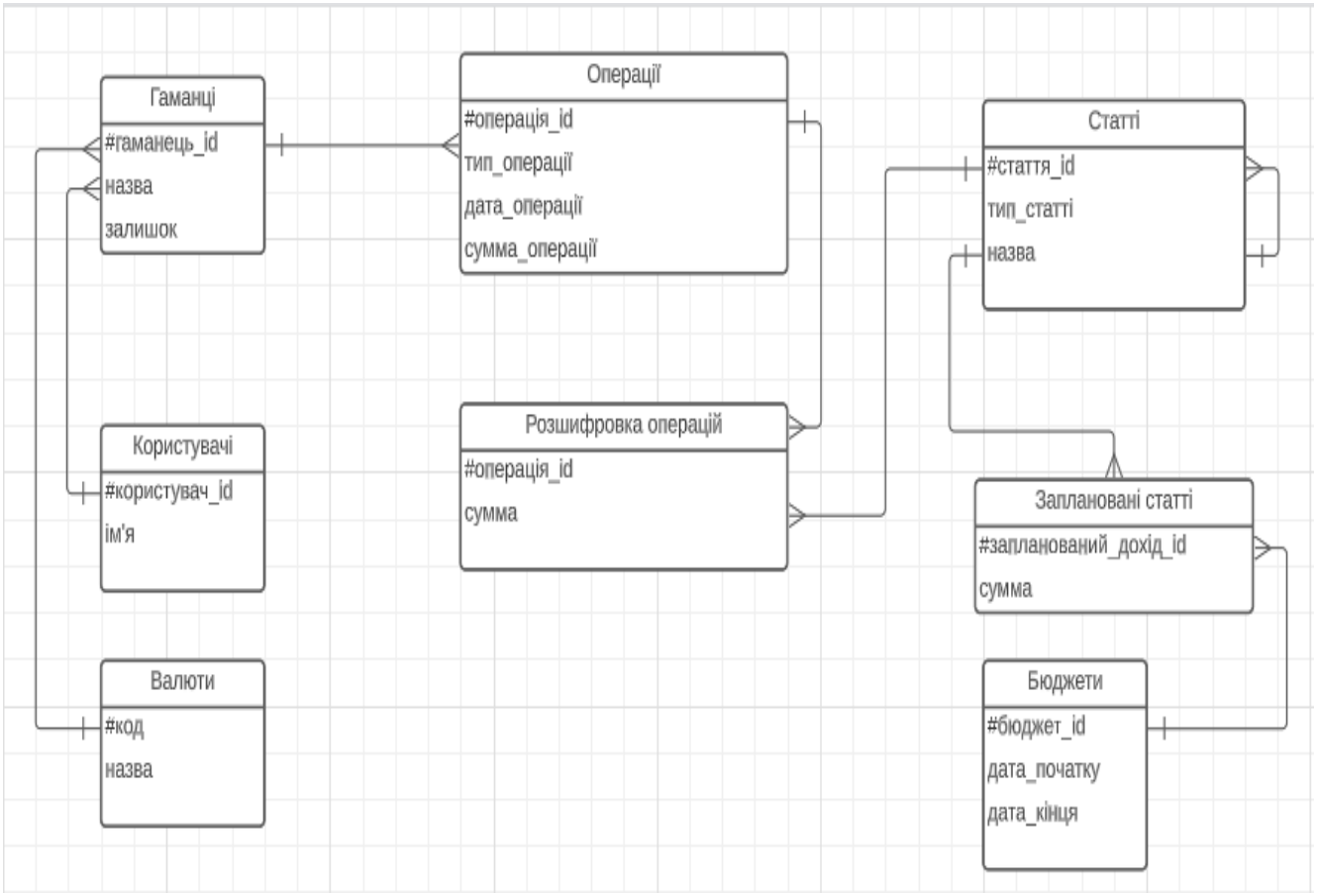


Рисунок 3.1 – ER-діаграма за нотацією Баркера

3.2 Вибір та побудова логічної моделі бази даних на основі ER-діаграми

Реляційна база даних – це база даних, яка використовується для зберігання та організації доступу до взаємопов’язаних елементів інформації [12]. Реляційні бази даних ґрунтуються на реляційній моделі – інтуїтивно зрозумілому, наочному поданні інформації у вигляді таблиць. Кожен рядок у таблиці такої бази даних являє собою запис унікальним ідентифікатором – ключем. Стовпці таблиць мають атрибути даних, що дає змогу встановлювати зв’язки між елементами даних. Саме цьому, реляційна модель бази даних є найбільш ефективною для вирішення задачі даного курсового проекту (див. рис. 3.2).

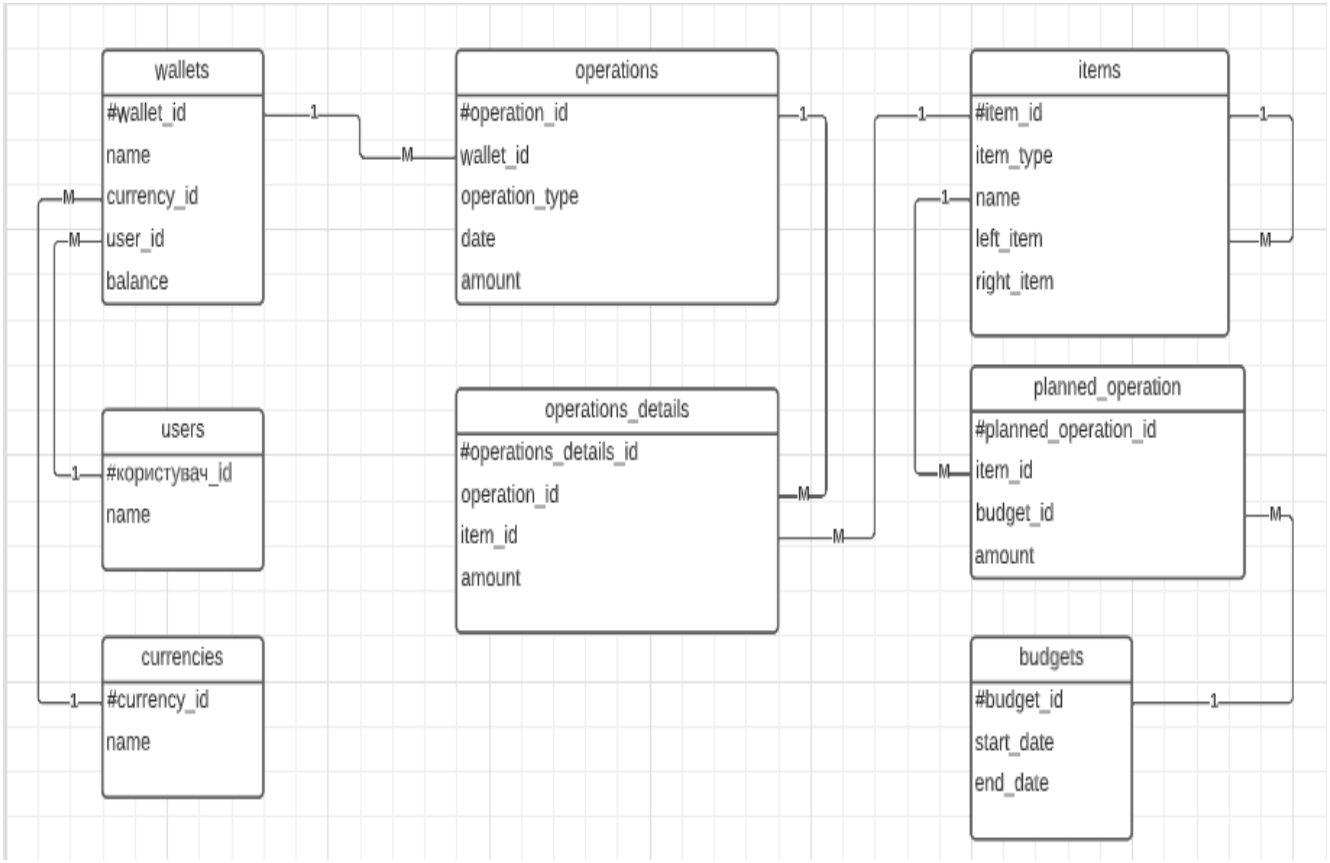


Рисунок 3.2 – Схема реляційної бази даних

3.3 Побудова логічної моделі бази даних шляхом нормалізації

Логічна модель бази даних є ключовим етапом у розробці інформаційної системи для управління фінансами домогосподарства. Оптимальна організація даних дозволяє забезпечити ефективну роботу з інформацією, швидкий доступ та зручність у взаємодії з базою даних.

Нормалізація є методом структуризації даних у базі таким чином, щоб уникнути аномалій та забезпечити ефективність запитів та збереження інтегрітету даних. Процес нормалізації включає розбиття таблиць на менші та детальніші, але більш декомпозовані, щоб уникнути зберігання зайвої інформації та забезпечити гнучкість структури.

Перший крок у нормалізації — це переведення всіх значень у таблиці у простий скалярний тип (атомарні значення). Наприклад, якщо у нас є колонка

"Адреса," ми розбиваємо її на окремі колонки, такі як "Місто," "Вулиця," "Поштовий індекс," щоб уникнути повторюваності даних.

Для досягнення другого нормального формального стану, необхідно впевнитися, що всі колонки в таблиці пов'язані із заголовним ключем, а не лише частиною його. Це може включати в себе розбиття таблиць на декілька, якщо потрібно.

Третій нормальний формальний стан вимагає видалення транзитивних залежностей. Це означає, що жодна колонка не повинна залежати від іншої не ключової колонки.

У нашому фінансовому додатку, нормалізація буде застосована до таблиць, які містять інформацію про транзакції, категорії витрат, користувачів та інші аспекти фінансового обліку. Розглядаючи зв'язки між цими елементами, ми визначимо ключі, розіб'ємо дані на більш малих сутностей, та визначимо правила для збереження цілісності даних. В нашому проекті при нормалізації виконується приведення таблиць до третьої нормальної форми. Відношення знаходиться у третій нормальній формі, коли воно знаходиться у першій та другій нормальній формі, тобто всі атрибути атомарні, кожний не ключовий атрибут повністю функціонально залежить від первісного ключа відношення та між неключовими атрибутами немає транзитивних залежностей.

Нормалізуємо таблицю де будуть зберігатись дані про операції. Початкова таблиця буде зберігати наступні дані:

- дата операції;
- тип операції (прибуток чи видаток);
- стаття доходу або расходу;
- сумма по статті доходу або расходу;
- назва гаманця;
- власник гаманця;
- валюта гаманця.

Таблиця операцій в початковому вигляді не відповідає 1-й нормальній формі, оскільки в ній є повторювані атрибути. Зокрема, атрибут Сума по статті доходу або расходу повторюється для кожної статті в операції.

Для того, щоб привести таблицю до 1-й нормальної форми, необхідно розбити її на дві таблиці:

1) таблиця операцій:
    1) ідентифікатор операції (первинний ключ);
    2) дата операції;
    3) тип операції;
    4) назва гаманця;
    5) валюта гаманця;
2) таблиця статей операцій:
    1) ідентифікатор операції (зовнішній ключ з таблиці операцій);
    2) назва статті;
    3) сума.

У таблиці операцій залишилися тільки атрибути, які не повторюються. Атрибут Сума по статті доходу або расходу був перенесений в таблицю статей операцій, оскільки він залежить від двох ключів: Ідентифікатор операції і Назва статті.

Для того, щоб таблиця перебувала в другій нормальній формі, вона повинна відповідати наступним критеріям:

- таблиця повинна бути в першій нормальній формі;
- будь-який неключовий атрибут повинен функціонально залежати від первинного ключа в цілому, а не від його частини.

У таблиці операцій атрибут Назва гаманця функціонально залежить від первинного ключа Ідентифікатор операції. Однак, оскільки Ідентифікатор операції складається з двох атрибутів, то Назва гаманця функціонально залежить від частини первинного ключа.

Для того, щоб привести таблицю до другої нормальної форми, необхідно розбити її на дві таблиці:

1) таблиця операцій:
    1) ідентифікатор операції (первинний ключ);
    2) дата операції;
    3) тип операції;
2) таблиця гаманців:
    1) ідентифікатор гаманця (первинний ключ);
    2) назва гаманця;
    3) валюта гаманця.

У таблиці операцій залишилися тільки атрибути, які залежать тільки від основного ключа. Атрибут Назва гаманця був перенесений в таблицю гаманців, оскільки він залежить від атрибута Ідентифікатор операції.

Таким чином, таблиця операцій в другій нормальній формі має наступний вигляд:

1) таблиця операцій:
    1) ідентифікатор операції (первинний ключ);
    2) дата операції;
    3) тип операції;
2) таблиця гаманців:
    1) ідентифікатор гаманця (первинний ключ);
    2) назва гаманця;
    3) валюта гаманця;
3) таблиця статей операцій:
    1) ідентифікатор операції (зовнішній ключ з таблиці операцій);
    2) назва статті;
    3) сума.

Для того, щоб таблиця перебувала в третій нормальній формі, вона повинна відповідати наступним критеріям:

- таблиця повинна бути в другій нормальній формі;
- будь-який неключовий атрибут не повинен функціонально залежати від іншого неключового атрибута, крім первинного ключа.

У таблиці операцій атрибут Назва статті функціонально залежить від атрибута Ідентифікатор операції. Однак, оскільки Назва статті не є частиною первинного ключа, то вона функціонально залежить від іншого неключового атрибута.

Для того, щоб привести таблицю до третьої нормальної форми, необхідно розбити її на чотири таблиці:

1) таблиця операцій:
    1) ідентифікатор операції (первинний ключ);
    2) дата операції;
    3) тип операції;
    4) ідентифікатор гаманця (зовнішній ключ з таблиці гаманців)
2) таблиця статей:
    1) ідентифікатор статті (первинний ключ);
    2) назва статті;
3) таблиця операцій і статей:
    1) ідентифікатор операції (зовнішній ключ з таблиці операцій);
    2) ідентифікатор статті (зовнішній ключ з таблиці статей);
4) таблиця гаманців:
    1) ідентифікатор гаманця (первинний ключ);
    2) назва гаманця;
    3) валюта гаманця.

У таблиці операцій залишилися тільки атрибути, які залежать тільки від основного ключа. Атрибут Назва статті був перенесений в таблицю статей, оскільки він залежить від атрибута Ідентифікатор операції.

В усіх відношеннях всі атрибути є атомарними, тому вони знаходяться в 1 НФ. При зведенні до другої нормальної форми було доведено, що всі відношення не містять неповних функціональних залежностей, тобто в складі потенційного ключа відсутня менша підмножина атрибутів від якої можна також вивести дану функціональну залежність. При зведенні до третьої нормальної форми була проведена перевірка на відсутність транзитивних функціональних залежностей.

Отже, отримана схема БД знаходиться в 3НФ.

3.4 Вибір засобів реалізації

Візуальне представлення програми розроблено на основі фреймворку Angular. Angular – це JavaScript-фреймворк, який дозволяє створювати веб-застосунки з використанням мови TypeScript. Angular забезпечує широкий спектр можливостей для створення зручного та інтуїтивно зрозумілого інтерфейсу користувача.

Для зберігання даних інформаційної системи було використано систему управління базами даних MySQL. MySQL – це відкрита система управління базами даних, яка є однією з найпопулярніших у світі. MySQL відрізняється високою продуктивністю, надійністю та масштабованістю.

Доступ до баз даних здійснюється за допомогою технології Node.js Express та бібліотеки MySQL2. Node.js Express – це фреймворк для розробки веб-застосунків на основі Node.js. Він забезпечує широкий спектр можливостей для створення ефективних і масштабованих веб-застосунків. MySQL2 – це бібліотека, яка надає доступ до системи управління базами даних MySQL. Вона є частиною Node.js і дозволяє виконувати SQL-запити до MySQL з JavaScript-коду.

3.5 Виклик і завантаження інформаційної системи

Для розгортання інформаційної системи необхідно встановити наступне програмне забезпечення [12]:

- Сервер бази даних MySql;
- Node.js.

Необхідно з веб-сайту MySql завантажити інсталятор для поточної операційної системи та запустити інсталятор, дотримуючись інструкцій.

Для установки Node.js необхідно відвідайте веб-сайт Node.js та завантажити інсталятор для вашої операційної системи, запустити інсталятор і дотримуватись інструкцій на екрані.

Для створення таблиць у базі даних MySql необхідно виконати наступні дії:

Запустіть програму для адміністрування баз даних MySql, наприклад MySQL Workbench.

Відкрийте файл schema.sql.

Виконайте запити з файлу schema.sql.

Налаштування параметрів підключення до бази даних

Параметри підключення до бази даних необхідно налаштувати у файлі \app\config\db.config.js.

Запуск бекендної частини виконується наступним чином [13]. Для запуску бекендної частини необхідно виконати наступні дії:

- перейдіть в папку з проектом;
- виконайте команду:

npm install

Ця команда встанове всі необхідні залежності для проекту.

- Виконайте команду:

npm start

Ця команда запустить сервер бекендної частини.

Запуск веб-застосунку виконується наступним чином. Для запуску веб-застосунку необхідно виконати наступні дії:

- перейдіть в папку з проектом;
- виконайте команду:

npm install

Ця команда встанове всі необхідні залежності для проекту.

- Виконайте команду:

npm start

Ця команда запустить веб-застосунок.

Файл schema.sql містить запити для створення таблиць у базі даних. Запити можна налаштувати відповідно до ваших потреб.

Параметри підключення до бази даних можна налаштувати в файлі \app\config\db.config.js. Параметри включають:

- ім'я сервера бази даних;
- порт сервера бази даних;
- ім'я користувача бази даних;
- пароль користувача бази даних;
- веб-застосунок можна запускати в будь-якому веб-браузері.

3.6 Призначення і логічна структура інформаційної системи

Інформаційна система призначена для управління фінансами домогосподарства. Вона дозволяє зберігати інформацію про доходи, витрати, гаманці, бюджети, валюти та користувачів[11].

Логічна структура інформаційної системи визначається взаємозв'язком між її складовими частинами. Вона складається з наступних елементів:

Об'єкти – це основні елементи предметної області, які зберігаються в БД.

Відношення – це зв'язки між об'єктами.

Процедури – це алгоритми обробки даних.

Об'єкти інформаційної системи можуть бути простими чи складними. Прості об'єкти мають один або кілька атрибутів, які характеризують їх. Складні об'єкти можуть містити в собі інші об'єкти.

Відношення між об'єктами можуть бути різних типів. Найбільш поширеними типами відносин є:

- однозначне співвідношення – один об'єкт може бути пов'язаний з одним або багатьма іншими об'єктами;
- багатозначне співвідношення – один об'єкт може бути пов'язаний з багатьма іншими об'єктами, а кожен з цих інших об'єктів може бути пов'язаний з багатьма об'єктами першого типу;
- зворотне співвідношення – два об'єкти можуть бути пов'язані один з одним в обох напрямках.

Об'єкти інформаційної системи для управління фінансами домогосподарства наступні:

- дохід – це сума грошей, отримана домогосподарством за певний період часу;
- витрати – це сума грошей, витрачених домогосподарством за певний період часу;
- гаманець – це контейнер для зберігання грошей;
- бюджет – це план розподілу грошей;
- валюта – це грошова одиниця країни або регіону;
- користувач – це особа, яка використовує інформаційну систему.

Відношення між об'єктами наступні:

- доход і витрати – доходи можуть бути використані для покриття витрат;
- гаманець і бюджет – бюджет може бути призначений для конкретного гаманця;
- валюта і бюджет – бюджет може бути виражений в певній валюті;
- користувач і бюджет – бюджет може бути створений або відредагований конкретним користувачем.

Процедури інформаційної системи наступні:

- внесення даних – це процедура внесення нової інформації в БД;
- оновлення даних – це процедура зміни існуючої інформації в БД;
- вилучення даних – це процедура видалення інформації з БД;
- пошук даних – це процедура пошуку інформації в БД.

При проектуванні логічної структури інформаційної системи для управління фінансами домогосподарства необхідно враховувати такі фактори:

- властивості предметної області. Логічна структура повинна відображати властивості предметної області, в якій працює інформаційна система;
- вимоги до функціональності. Логічна структура повинна забезпечувати реалізацію всіх необхідних функцій інформаційної системи;
- вимоги до продуктивності. Логічна структура повинна забезпечувати високу продуктивність інформаційної системи;

– вимоги до безпеки. Логічна структура повинна забезпечувати захист даних інформаційної системи.

Логічна структура інформаційної системи для управління фінансами домогосподарства повинна бути такою, щоб вона дозволяла користувачам легко і ефективно керувати своїми фінансами. Вона повинна відображати властивості предметної області, забезпечувати реалізацію всіх необхідних функцій, забезпечувати високу продуктивність і захист даних.

4 ОПИС ПРИЙНЯТИХ ПРОГРАМНИХ РІШЕНЬ

4.1 Описання фізичної моделі бази даних

При реалізації фізичної моделі бази даних було створено вісім таблиць [11]. Нижче наведено команди на їх створення.

```
CREATE SCHEMA IF NOT EXISTS `householddb` DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci ;
USE `householddb` ;

CREATE TABLE IF NOT EXISTS `householddb`.`budgets` (
  `budget_id` INT NOT NULL AUTO_INCREMENT,
  `start_date` DATE NOT NULL,
  `end_date` DATE NOT NULL,
  PRIMARY KEY (`budget_id`));

CREATE TABLE IF NOT EXISTS `householddb`.`currencies` (
  `currency_id` INT NOT NULL,
  `name` VARCHAR(255) NOT NULL,
  PRIMARY KEY (`currency_id`));
CREATE TABLE IF NOT EXISTS `householddb`.`items` (
  `item_id` INT NOT NULL AUTO_INCREMENT,
  `item_type` INT NOT NULL,
  `name` VARCHAR(255) NOT NULL,
  `left_item` INT NULL DEFAULT NULL,
  `right_item` INT NULL DEFAULT NULL,
  PRIMARY KEY (`item_id`));

CREATE TABLE IF NOT EXISTS `householddb`.`users` (
  `user_id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(255) NOT NULL,
  PRIMARY KEY (`user_id`));

CREATE TABLE IF NOT EXISTS `householddb`.`wallets` (
  `wallet_id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(255) NOT NULL,
  `currency_id` INT NOT NULL,
  `user_id` INT NOT NULL,
  `balance` DECIMAL(12,2) NULL DEFAULT '0.00',
  PRIMARY KEY (`wallet_id`),
  INDEX `currency_idx` (`currency_id` ASC) VISIBLE,
  INDEX `user_idx` (`user_id` ASC) VISIBLE,
  CONSTRAINT `currency`
    FOREIGN KEY (`currency_id`)
    REFERENCES `householddb`.`currencies` (`currency_id`),
  CONSTRAINT `user`
    FOREIGN KEY (`user_id`)
    REFERENCES `householddb`.`users` (`user_id`));

CREATE TABLE IF NOT EXISTS `householddb`.`operations` (
  `operation_id` INT NOT NULL AUTO_INCREMENT,
  `operation_type` INT NOT NULL,
  `wallet_id` INT NOT NULL,
  `date` DATETIME NULL DEFAULT NULL,
```

```
  `amount` DECIMAL(12,2) NULL DEFAULT NULL,
  PRIMARY KEY (`operation_id`),
  INDEX `wallet_idx` (`wallet_id` ASC) VISIBLE,
  INDEX `operations_wallet_idx` (`wallet_id` ASC) VISIBLE,
  CONSTRAINT `operations_wallet`
    FOREIGN KEY (`wallet_id`)
    REFERENCES `householddb`.`wallets` (`wallet_id`));

CREATE TABLE IF NOT EXISTS `householddb`.`operations_details` (
  `operations_details_id` INT NOT NULL AUTO_INCREMENT,
  `operation_id` INT NOT NULL,
  `item_id` INT NOT NULL,
  `amount` DECIMAL(12,2) NOT NULL,
  PRIMARY KEY (`operations_details_id`),
  INDEX `operation_idx` (`operation_id` ASC) VISIBLE,
  INDEX `item_idx` (`item_id` ASC) VISIBLE,
  INDEX `item_id_idx` (`item_id` ASC) VISIBLE,
  CONSTRAINT `item_id`
    FOREIGN KEY (`item_id`)
    REFERENCES `householddb`.`items` (`item_id`),
  CONSTRAINT `operation`
    FOREIGN KEY (`operation_id`)
    REFERENCES `householddb`.`operations` (`operation_id`))
    ON DELETE CASCADE;

CREATE TABLE IF NOT EXISTS `householddb`.`planned_operation` (
  `planned_operation_id` INT NOT NULL AUTO_INCREMENT,
  `item_id` INT NOT NULL,
  `budget_id` INT NOT NULL,
  `amount` DECIMAL(12,2) UNSIGNED ZEROFILL NULL DEFAULT NULL,
  PRIMARY KEY (`planned_operation_id`),
  INDEX `item_idx` (`item_id` ASC) VISIBLE,
  INDEX `budget_idx` (`budget_id` ASC) VISIBLE,
  CONSTRAINT `budget`
    FOREIGN KEY (`budget_id`)
    REFERENCES `householddb`.`budgets` (`budget_id`)
    ON DELETE CASCADE,
  CONSTRAINT `item`
    FOREIGN KEY (`item_id`)
    REFERENCES `householddb`.`items` (`item_id`));
```

4.2 Опис програмної реалізації

Програмна реалізація задачі автоматизації формування бюджету на підставі статистики доходів та витрат в попередніх періодах здійснюється за допомогою веб-додатку, що працює на сервері.

Веб-додаток складається з двох основних частин [14]:

− клієнтська частина, яка виконується на стороні користувача і забезпечує інтерфейс для введення даних та перегляду результатів;

– серверна частина, яка виконується на сервері і забезпечує обробку даних та формування бюджету.

Клієнтська частина веб-додатку написана на JavaScript і HTML [15,156]. Вона забезпечує інтерфейс для введення даних про доходи та витрати.

Користувач може вводити дані про доходи та витрати за допомогою форм.(Рисунок 4.1, 4.2) Дані про доходи включають інформацію про суму, дату та тип доходу. Дані про витрати включають інформацію про суму, дату, тип витрати та мету витрат.

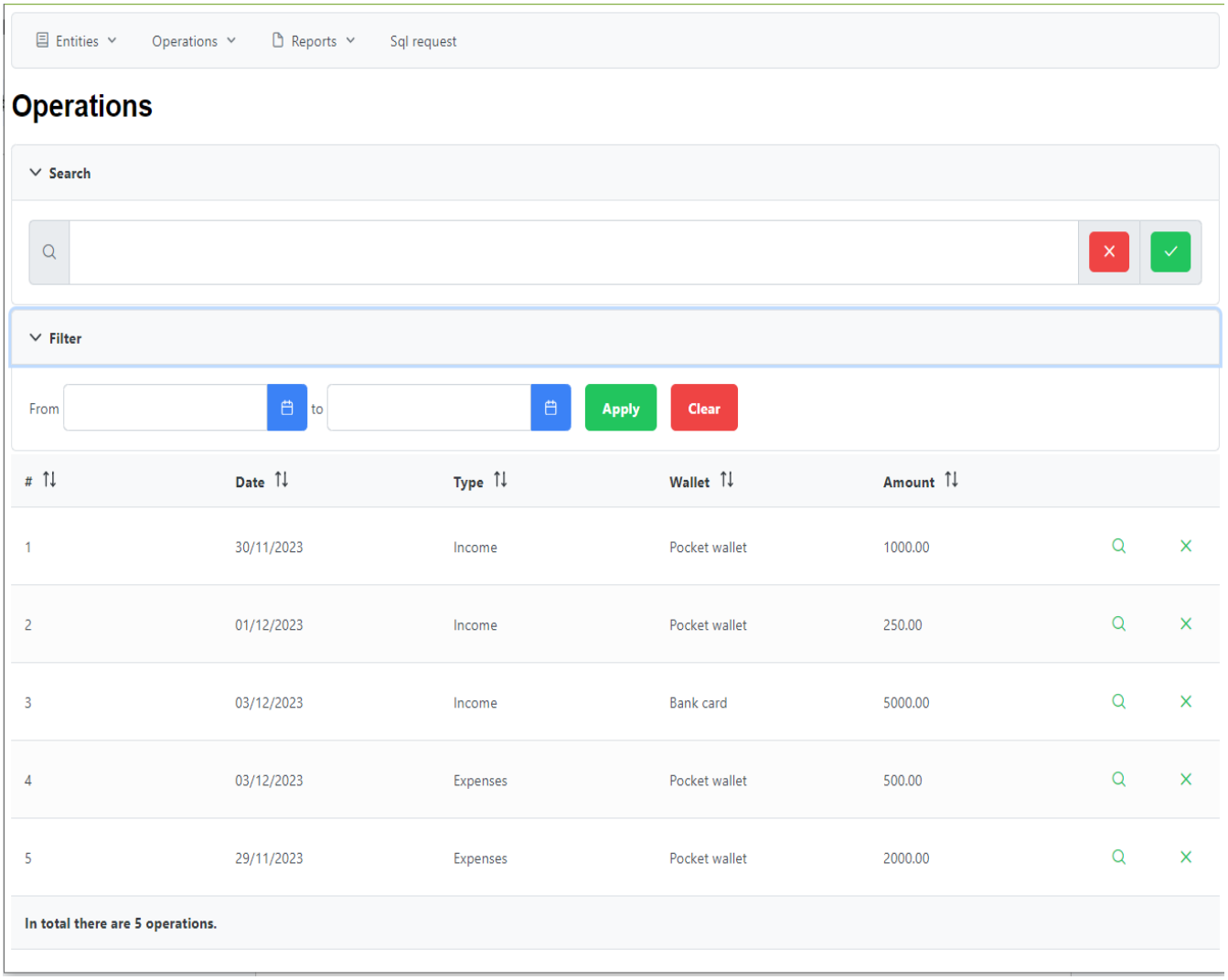


Рисунок 4.1 – Форма списку операцій

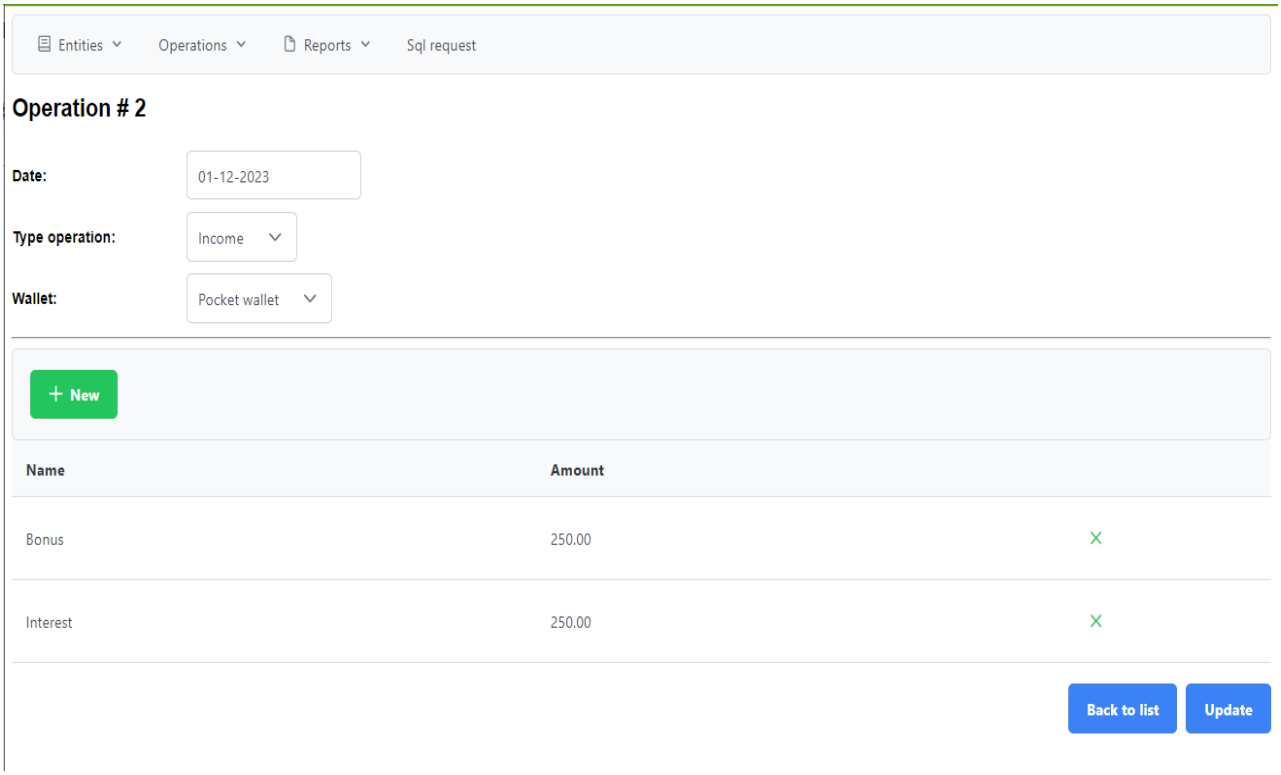


Рисунок 4.2 – Форма операції

Крім того, клієнтська частина веб-додатку забезпечує інтерфейс для перегляду результатів формування бюджету (Рисунок 4.3, рисунок 4.4). Користувач може переглядати бюджет у вигляді таблиці або графіка.

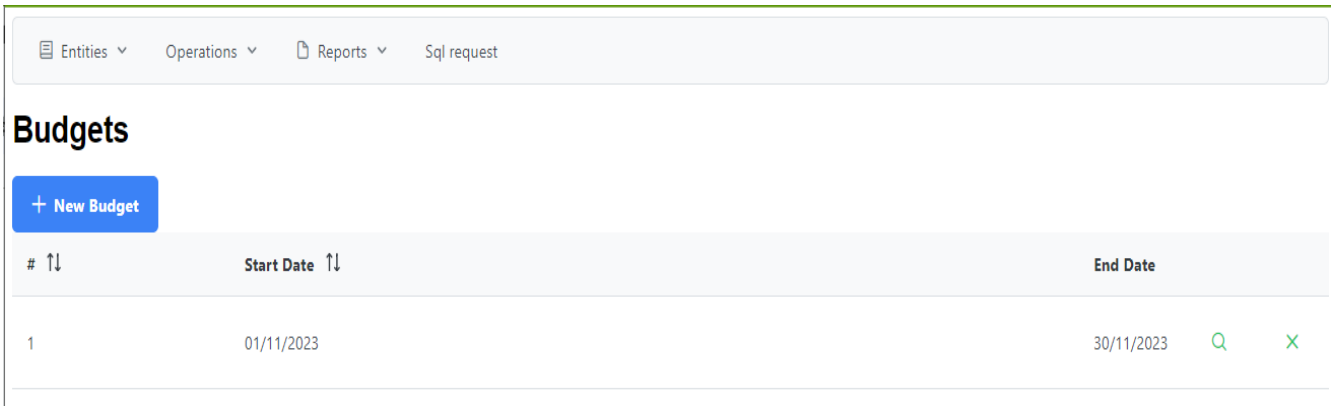


Рисунок 4.3 – Форма списку бюджетів

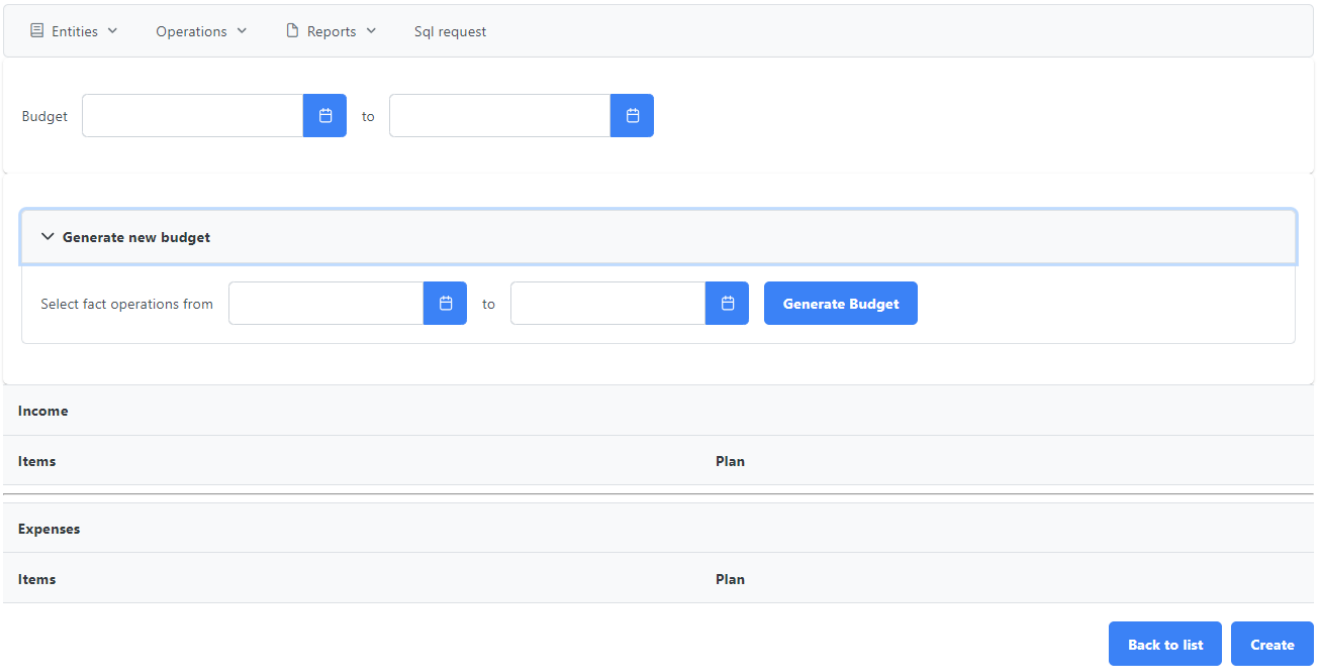


Рисунок 4.4 – Форма створення бюджету

Серверна частина веб-додатку написана на Node.js. Вона забезпечує обробку даних та формування бюджету.

Серверна частина веб-додатку отримує дані від клієнтської частини через HTTP-запити. Після обробки даних серверна частина формує бюджет і повертає його клієнтській частині.

Обробка даних здійснюється за допомогою наступних кроків:

– збір даних: дані про доходи та витрати збираються з різних джерел, наприклад, з бухгалтерських документів, електронних платежів, банківських виписок тощо;

– аналіз даних: на цьому етапі здійснюється аналіз даних, щоб визначити тенденції та закономірності в доходах та витратах. Це може включати такі операції, як розрахунок середнього значення, стандартного відхилення, коефіцієнта варіації тощо;

– формування бюджету: на цьому етапі здійснюється формування бюджету на підставі результатів аналізу даних. Це може включати такі операції, як розподіл бюджету по категоріям витрат, визначення запасу коштів тощо.

Крім основних функцій, веб-додаток може забезпечувати додаткові можливості, такі як:

− автоматичне завантаження даних: дані про доходи та витрати можуть бути автоматично завантажені з різних джерел, наприклад, з бухгалтерських програм, електронних платежів, банківських виписок тощо;

− налаштування бюджету: користувач може налаштувати бюджет відповідно до своїх потреб;

− повідомлення про перевищення бюджету: користувач може отримувати повідомлення про перевищення бюджету.

Реалізація додаткових можливостей здійснюється шляхом розширення функціональності веб-додатку. Наприклад, для реалізації автоматичного завантаження даних необхідно додати в клієнтську частину веб-додатку додаткові форми для введення інформації про джерела даних. Для реалізації налаштування бюджету необхідно додати в клієнтську частину веб-додатку додаткові параметри для налаштування бюджету. Для реалізації повідомлення про перевищення бюджету необхідно додати в серверну частину веб-додатку додаткові функції для обробки даних про бюджет.

Структура проекту серверної частини організована таким чином, щоб розділити відповідальність між різними частинами застосунку, що сприяє легшому управлінню кодом і підвищенню його підтримуваності:

− папка config містить файл db.config.js, в якому зберігається конфігурація для підключення до бази даних;

− папка controllers містить файли, в яких зберігаються контролери для обробки запитів, що пов’язані з різними моделями системи;

− папка models містить файли, в яких зберігаються моделі з різними даними системи;

− папка routes містить файли, в яких зберігаються маршрути щодо об’єктів системи.

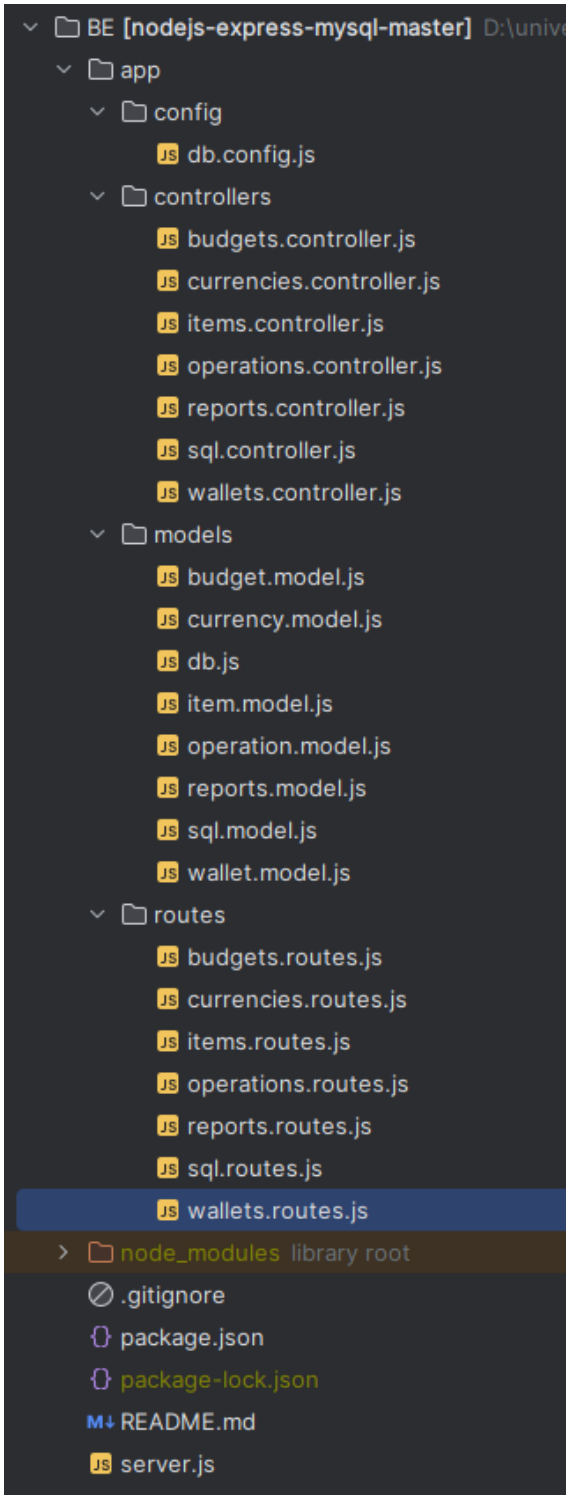


Рисунок 4.5 – Структура серверної частини проекту

wallets.routes.js

```
module.exports = app => {
    const wallets : {...}  = require("../controllers/wallets.controller.js");

    var router : Router  = require("express").Router();

    // Create a new Wallet
    router.post( path: "/", wallets.create);

    // Retrieve all Wallets
    router.get( path: "/", wallets.findAll);

    // Retrieve a single Wallet with id
    router.get( path: "/:id", wallets.findOne);

    // Update Wallet with id
    router.put( path: "/:id", wallets.update);

    // Delete Wallet with id
    router.delete( path: "/:id", wallets.delete);


    app.use('/api/wallets', router);
};

```

Рисунок 4.6 – Маршрути для обробки запитів, пов'язаних з гаманцями

wallets.controller.js

```
const Wallet : function(any): void | {...}  = require("../models/wallet.model.js");

// Create and Save a new item
exports.create = (req, res) : void  => {
    // Validate request
    if (!req.body) {
        res.status(400).send({
            message: "Content can not be empty!"
        });
    }

    // Create an item
    const item : Wallet  = new Wallet( item: {
        item_id: req.body.item_id,
        name: req.body.name,
    });

    // Save Item in the database
    Wallet.create(item,  result: (err, data) : void  => {
        if (err)
            res.status(500).send({
                message:
                    err.message || "Some error occurred while creating the Wallet."
            });
        else res.send(data);
    });
};

exports.findAll = (req, res) : void  => {
    Wallet.getAll(req.query,  result: (err, data) : void  => {
        if (err)
            res.status(500).send({
                message:
                    err.message || "Some error occurred while retrieving wallets."
            });
        else res.send(data);
    });
};
```

Рисунок 4.7 – Контролер для обробки запитів, пов'язаних з гаманцями

wallet.model.js

```
const sql : Pool | {...}  = require("./db.js");

const allowedSortFields : string[]  = ['wallet_id', 'name'];
// constructor
5+ usages  ivava777
const Wallet = function(item) : void  {
  this.wallet_id = item.wallet_id;
  this.name = item.name;
};

Wallet.create = (newItem, result) : void  => {
  sql.query( sql: "INSERT INTO wallets SET ?", newItem,  callback: (err : QueryError | null , res : ... ) : void  => {
    if (err) {
      console.log("error: ", err);
      result(err, null);
      return;
    }

    console.log("created item: ", { id: res.insertId, ...newItem });
    result(null, { id: res.insertId, ...newItem });
  });
};

Wallet.findById = (item_id, result) : void  => {
  sql.query( sql: `SELECT * FROM wallets WHERE wallet_id = ${item_id}`,  callback: (err : QueryError | null , res : ... ) : void  => {
    if (err) {
      console.log("error: ", err);
      result(err, null);
      return;
    }

    if (res.length) {
      console.log("found wallet: ", res[0]);
      result(null, res[0]);
      return;
    }

    // not found Item with the id
    result({ kind: "not_found" }, null);
  });
};
```

Рисунок 4.8 – Модель для роботи з даними по гаманцям

4.3 Описання задачі автоматизації

Задача автоматизації полягає в формуванні бюджету на підставі статистики доходів та витрат в попередніх періодах.

Вхідними даними для задачі є: дані про доходи, дані про витрати в обраному періоді

Вихідними даними для задачі є бюджет, який включає інформацію про суму, період та категорії витрат та доходів.

Автоматизація задачі реалізована в кілька етапів:

- збір даних. На першому етапі необхідно зібрати дані про доходи та витрати за попередні періоди. Ці дані можуть бути зібрані з різних джерел, наприклад, з бухгалтерських документів, електронних платежів, банківських виписок тощо;
- аналіз даних. Необхідно провести аналіз даних, щоб визначити тенденції та закономірності в доходах та витратах;
- формування бюджету. На цьому етапі необхідно сформувати бюджет на підставі результатів аналізу даних. Це включає такі операції, як розподіл бюджету по категоріям витрат, визначення запланованих доходів.

5 ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Для тестування програмної системи управління фінансами домогосподарства використовувалося мануальне тестування. Основними підвидами мануального тестування є як функціональне, так і нефункціональне, так як у front-end розробці потрібно враховувати обидва види тестування [17].

5.1 Мета тестування

Мета тестування веб-додатку "Програмна система управління фінансами домогосподарства" полягає у виявленні та усуненні дефектів, перевірці відповідності функціональних та нефункціональних вимог, а також забезпеченні стабільної та безпечної роботи системи.

5.2 Методи тестування

Тестування системи включає наступні методи:

- Функціональне тестування: перевірка відповідності функціональності системи заявленим вимогам. При проведенні функціонального тестування було виявлено що система повністю відповідає поставленій задачі та може використовуватися сторонніми користувачами.
- Інтеграційне тестування: перевірка взаємодії між різними модулями системи. На даному етапі було перевірено інтеграцію веб-додатку з базою даних. Також була протестована інтеграція з зовнішньою системою клієнт-банкінгу. Інтеграційне тестування не виявило дефектів.
- Системне тестування: перевірка системи в цілому, включаючи інтеграцію всіх компонентів. Після завершення розробки усіх компонентів було проведено системне тестування, яку показало, що всі компоненти інтегровано коректно.
- Регресійне тестування: перевірка стабільності системи після внесення змін або виправлення дефектів. Після додавання кожної нової функції було проведено

регресійне тестування, яке показало, що додавання нових функцій не призвело до зниження якості програмного продукту.

− Тестування продуктивності: оцінка швидкості роботи системи під різними навантаженнями. При тестуванні було перевірено паралельне використання програмної системи 500 юзерами. Система відпрацювала без просідань.

− Тестування безпеки: перевірка захищеності системи від можливих загроз. При тестуванні безпеки була перевірена стійкість системи до зовнішнього втручання та виключена можливість SQL-ін'єкцій.

5.3 План тестування

План тестування включає наступні етапи:

− підготовка тестових сценаріїв: створення сценаріїв для кожного методу тестування, включаючи позитивні та негативні тести. Тестові сценарії було об’єднано у тест-сьюти, які проганялися при перевірці кожної нової версії системи;

− вибір тестових даних: підготовка необхідних даних для проведення тестів. Тестові дані можуть використовуватися на кожній ітерації тестування що дозволить знизити час тестування;

− виконання тестів: проведення тестування за підготовленими сценаріями. Даний етап дозволяє виявити помилки на кожному етапі тестування системи;

− аналіз результатів тестування: виявлення дефектів та їх класифікація за ступенем критичності. Заведення дефектів у трекінговій системі дозволяє навіть після виправлення помилки розуміти проблемні місця системи, які необхідно краще покривати регресійним тестуванням.

− виправлення дефектів: усунення виявлених дефектів розробниками. Даний етап підвищує якість системи.

− повторне тестування: перевірка виправлених дефектів та виконання регресійного тестування.

5.4 Результати тестування

На основі результатів тестування було прийнято рішення що система готова до впровадження. Система може бути рекомендована до використання користувачами для управління фінансами домогосподарства.

6 ВПРОВАДЖЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

6.1 Визначення плану впровадження

У сучасному світі, де фінансова стабільність є важливою складовою добробуту кожної родини особливо у періоди підвищення цін та підвищення рівня безробіття, ефективне управління фінансами домогосподарства стає необхідним інструментом підтримки родин. Веб-додаток "Програмна система управління фінансами домогосподарства" призначений для допомоги сім'ям у відстеженні доходів та витрат, плануванні бюджету та прийнятті фінансових рішень.

Мета впровадження програмної системи управління фінансами домогосподарства полягає в забезпеченні ефективного та зручного інструменту для управління фінансами домогосподарств. Система надає можливість детального аналізу фінансових потоків, планування витрат та формування бюджету на основі реальних даних.

При впровадженні програмної системи управління фінансами домогосподарства слід виділити наступні етапи:

1) Підготовчий етап:
    a) Вивчення потреб користувачів.
    b) Розробка технічного завдання та проектування системи.
    c) Вибір технологічних інструментів для розробки (Webstorm, Angular, Express.js, MySQL Server).
2) Розробка системи:
    a) Створення архітектури веб-додатку.
    b) Розробка бази даних для зберігання фінансових даних.
    c) Реалізація функціоналу для введення, зберігання та аналізу даних про доходи та витрати.
3) Тестування:
    a) Проведення функціонального, інтеграційного та системного тестування.
    b) Виправлення виявлених дефектів та проведення регресійного тестування.

4) Впровадження та навчання користувачів:
    a) Встановлення системи на серверах та налаштування доступу для користувачів.
    b) Проведення навчальних сесій для користувачів з метою ознайомлення з функціоналом системи.
    c) Надання технічної підтримки користувачам під час первинного етапу експлуатації.

Впровадження програмної системи управління фінансами домогосподарства забезпечить:

– оптимізацію процесу управління фінансами домогосподарства;
– зручний інтерфейс для введення та аналізу фінансових даних;
– можливість планування бюджету на основі реальних витрат та доходів;
– підвищення фінансової грамотності користувачів та поліпшення їх фінансового стану.

Впровадження програмної системи управління фінансами домогосподарства є важливим кроком на шляху до підвищення фінансової стабільності родин. Система надає зручний інструмент для аналізу та планування фінансів, що сприятиме прийняттю обґрунтованих фінансових рішень та покращенню фінансового добробуту домогосподарств.

ВИСНОВКИ

У роботі було проведено аналіз предметної області управління фінансами домогосподарства. Було визначено основні об'єкти та відносини, які характеризують цю предметну область.

Була описана задача автоматизації формування бюджету на підставі статистики доходів та витрат в попередніх періодах. Були визначені вхідні дані, вихідні дані, етапи автоматизації та переваги від автоматизації.

На підставі проведеного аналізу можна зробити наступні висновки: інформаційна система для управління фінансами домогосподарства повинна бути такою, щоб вона дозволяла користувачам легко і ефективно керувати своїми фінансами. Вона повинна відображати властивості предметної області, забезпечувати реалізацію всіх необхідних функцій, забезпечувати високу продуктивність і захист даних.

Автоматизація задачі формування бюджету на підставі статистики доходів та витрат має ряд переваг, таких як збільшення продуктивності, підвищення точності, зменшення витрат часу та покращення якості прийняття рішень. Врахування статистики доходів та витрат в попередніх періодах дозволяє проводити детальний аналіз та планування розвитку домогосподарства та своєчасне виявлення проблем планування.

Отже, впровадження інформаційної системи ведення обліку витрат домогосподарства у формі веб-додатка сприяє підвищенню фінансової грамотності та покращенню фінансового стану домогосподарств, забезпечуючи контроль і планування фінансових ресурсів на основі точної та актуальної інформації. Використання у повсякденному житті інформаційної системи управління фінансами домогосподарства дозволить більш ефективно вирішувати соціально-економічні проблеми розвитку фінансів домогосподарств, забезпечити гідний рівень і якість життя населення.

Вимоги до кваліфікаційної роботи виконані повністю. Інформаційна система відповідає актуальним вимогам і може бути використана в реальній системі управління фінансами домогосподарства.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Ковалев С. Ф., Волкова О. І. Інформаційні системи управління фінансами домогосподарств: навч. посіб. – К.: КНУБА, 2022.

2. Юрій С. І. Фінанси домогосподарств: теоретичні підходи до трактування сутності / С. І. Юрій, Т. О. Кізима // Фінанси України. – 2008. – № 8. – С. 3 – 10.

3. Фінанси : підручник / [С. І. Юрій, В. М. Федосов, Л. М. Алексеєнко та ін.] За ред. С. І. Юрія, В. М. Федосова. – Київ: Знання, 2008. – 611 с.

4. Скриньковський Р. М. Формування доходів домогосподарств та їх вплив на рівень життя населення / Р. М. Скриньковський, С. Р. Леськів // Агросвіт. – 2015. – № 1. – С. 25–29.

5. Дмитренко О. Розробка систем управління фінансами домогосподарств: вивчення випадків. / О. Дмитренко, В.Василенко, Р.Шаблисті.– Київ: Київський університет, 2020. – 150 с.

6. Петрова Н. Роль систем управління фінансами домогосподарств в сучасних економічних умовах. – Харків: Харківський національний економічний університет, 2019. – 180 с.

7. Коваленко А. Фінансова грамотність та управління бюджетом домогосподарств: програмні рішення. / А. Коваленко , А. Світлична. – Львів: Львівська політехніка, 2021. – 200 с.

8. Черняк В. Інновації в системах управління фінансами домогосподарств в Україні. – Одеса: Одеський національний університет, 2018. – 220 с.

9. Бондаренко Ю. Програмне забезпечення для управління фінансами домогосподарств: розробка та впровадження. – Дніпро: Дніпровський державний технічний університет, 2022. – 190 с.

10. Ambler, S. W. The Elements of UML 2.0 Style. Cambridge University Press, 2005. – 200 с. ISBN: 978-0521616782

11. Зайцев Г. В. Реляційні бази даних: навч. посіб. – К.: НТУУ «КПІ», 2021.

12. Paul Dubois, MySQL 8: The Definitive Guide, Apress, 2022.

13. Maximilian Schwarzmüller, Node.js: The Complete Reference, Packt Publishing, 2023

14. Eric Myer, Cascading Style Sheets: The Definitive Guide, O'Reilly Media, 2017

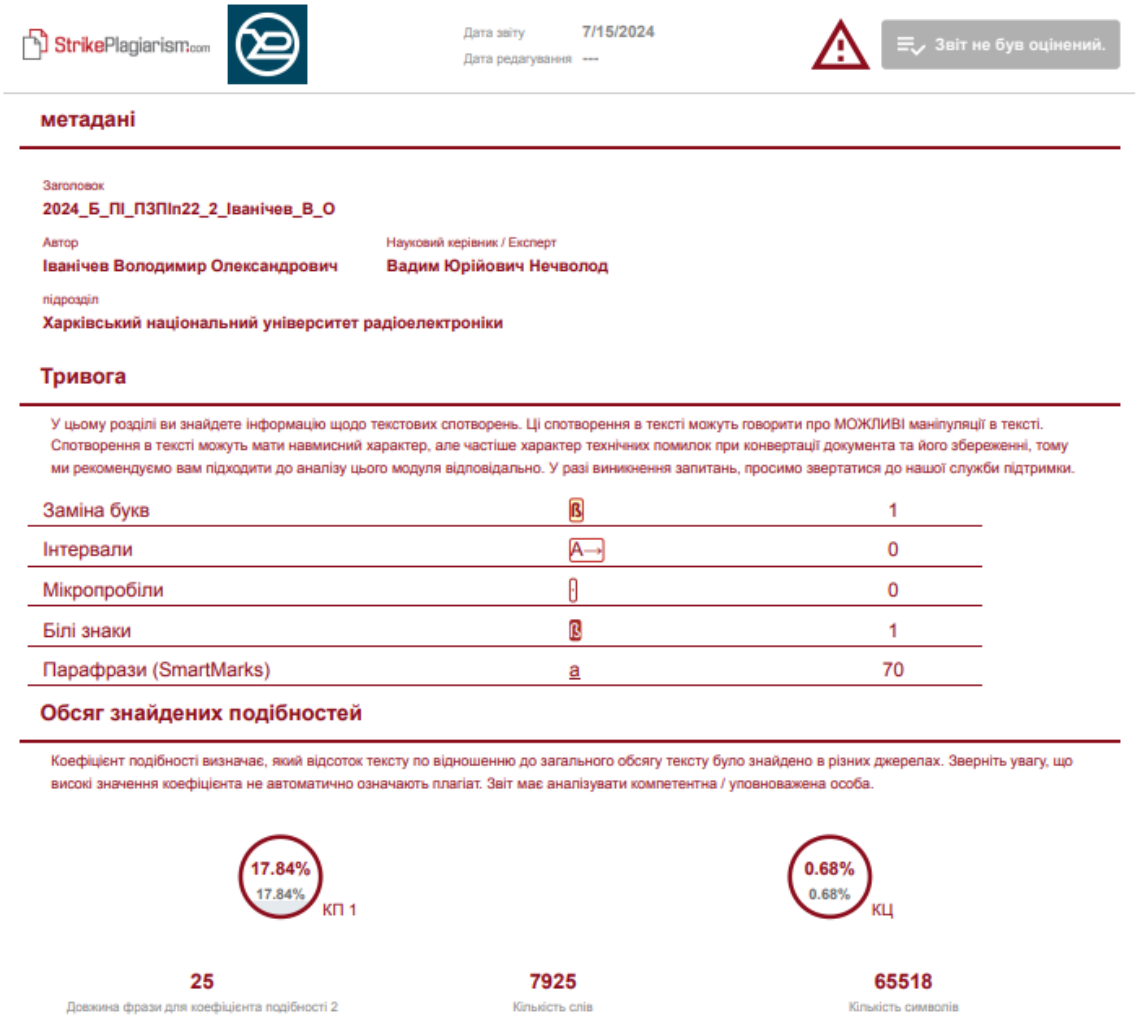
15. Maximilian Schwarzmüller, Angular: Complete Guide, Packt Publishing, 2023

16. Ben Nadel, TypeScript Cookbook, O'Reilly Media, 2021

17. Myers, G. J., Sandler, C., & Badgett, T. (2011). The Art of Software Testing. – Вид. 3-е. – Нью-Йорк: Вайлі, 2011. – 256 с.

ДОДАТОК А

Звіт результатів перевірки на унікальність тексту в базі ХНУРЕ



StrikePlagiarism.com

Дата звіту 7/15/2024
Дата редагування ---

Звіт не був оцінений.

метадані

Заголовок
2024_Б_ПІ_ПЗПІп22_2_Іванічев_В_О

Автор
Іванічев Володимир Олександрович

Науковий керівник / Експерт
Вадим Юрійович Нечволод

підрозділ
Харківський національний університет радіоелектроніки

Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв	ß	1
Інтервали	A→	0
Мікропробіли	¶	0
Білі знаки	ß	1
Парафрази (SmartMarks)	a	70

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.

17.84% 17.84% КП 1

0.68% 0.68% КЦ

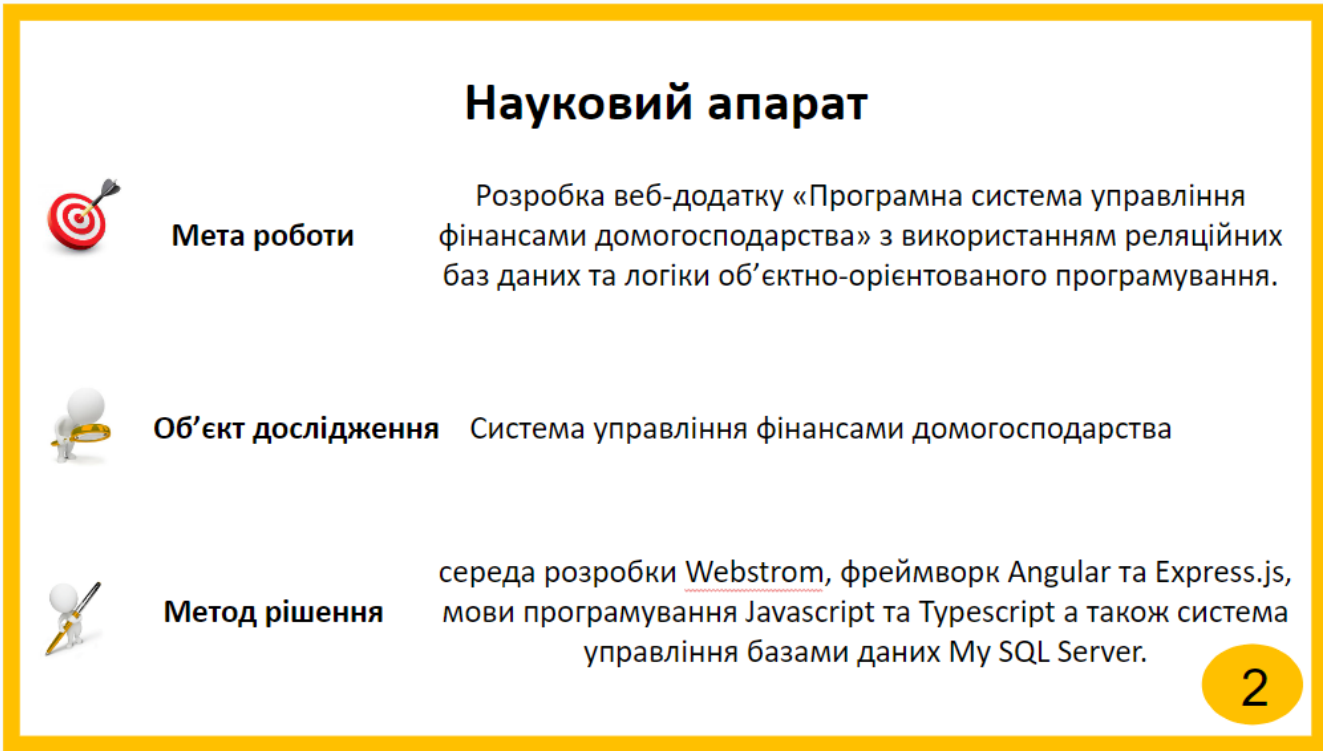
25
Довжина фрази для коефіцієнта подібності 2

7925
Кількість слів

65518
Кількість символів

ДОДАТОК Б

Слайди презентації

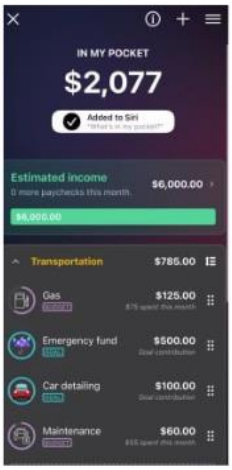


Аналіз проблеми

3

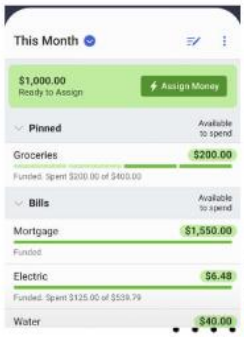
Аналіз існуючих технологічних рішень

Mint



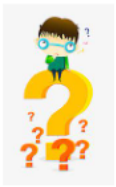
PocketGuard

You Need A Budget



4

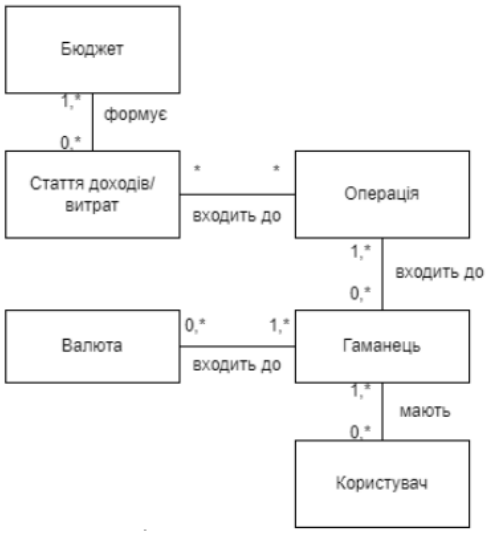
Завдання проєкту



1. Аналіз існуючих рішень;
2. система повинна відображати дані:
3. підтримка додавання нових та редагування існуючих даних про користувачів, гаманці, бюджети, прибутки та видатки;
4. підтримка арифметичної обробки даних у вигляді обчислювальних полів: стосовно залишків грошових коштів на гаманцях;
5. підтримка сортування, пошуку та фільтрації даних;
6. підтримка вилучення інформації про прибутки та видатки;
7. підтримка формування статистики;
8. підтримка формування довільного запиту до БД на мові SQL;
9. підтримка підготовки та друку звітів.

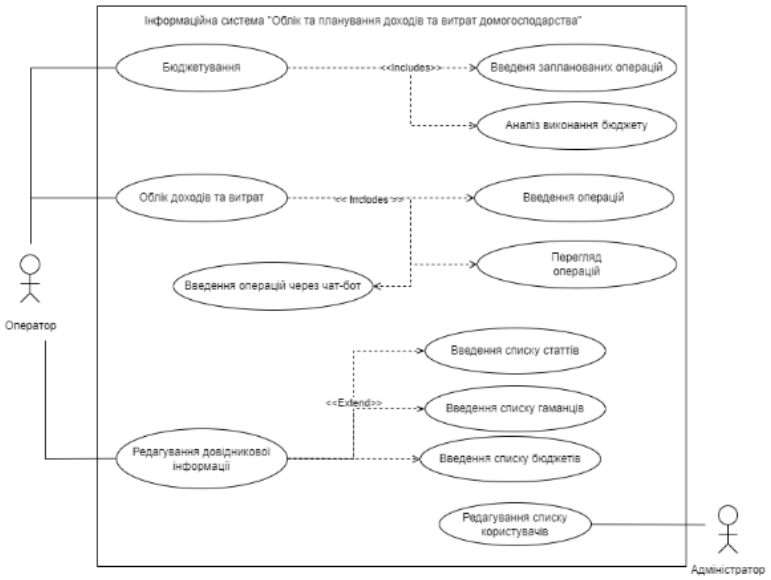
5

Загальна діаграма класів



6

USE-CASE діаграма для опису інформаційних потреб

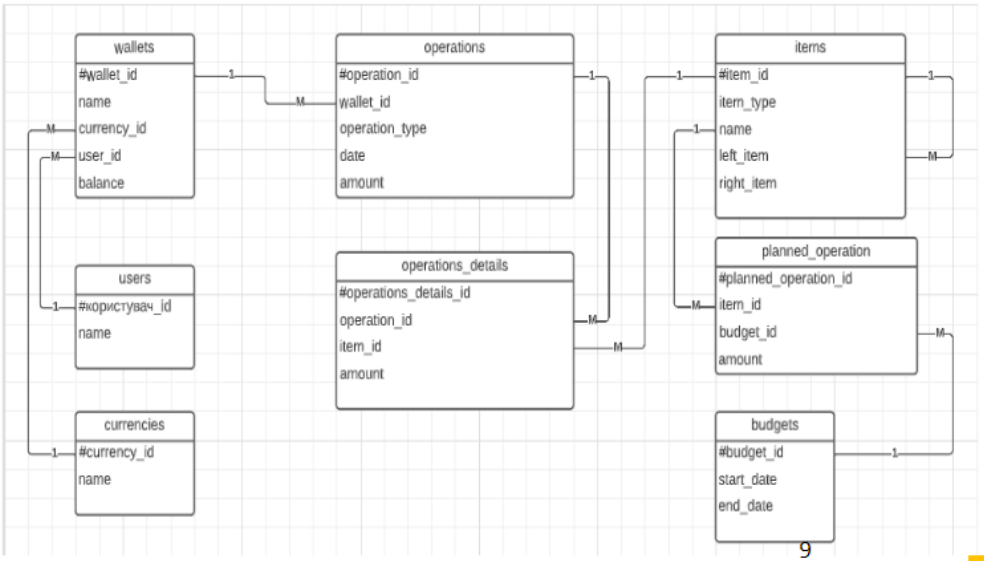


7

Діаграма станів роботи з бюджетом

8

Схема реляційної бази даних



9

Вибір програмних засобів

10

Приклад реалізації

Маршрути для обробки запитів, пов'язаних з гаманцями

wallets.routes.js

```
module.exports = app => {
    const wallets = require("../controllers/wallets.controller.js");

    var router = require("express").Router();

    // Create a new Wallet
    router.post("/", wallets.create);

    // Retrieve all Wallets
    router.get("/", wallets.findAll);

    // Retrieve a single Wallet with id
    router.get("/:id", wallets.findOne);

    // Update Wallet with id
    router.put("/:id", wallets.update);

    // Delete Wallet with id
    router.delete("/:id", wallets.delete);


    app.use('/api/wallets', router);
};
```

11

Приклад реалізації

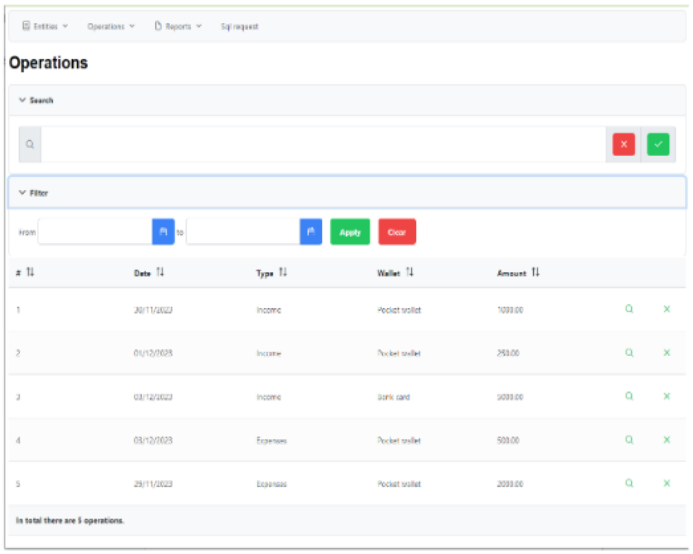
Контролер для обробки запитів, пов'язаних з гаманцями

wallets.controller.js

```
const Wallet = require("../models/wallet.model.js");

// Create and Save a new Item
exports.create = (req, res) => {
    // Validate request
    if (!req.body) {
        res.status(400).send({
            message: "Content can not be empty!"
        });
    }

    // Create an Item
    const item = new Wallet({
        item_id: req.body.item_id,
        name: req.body.name,
    });

    // Save Item in the database
    Wallet.create(item, (err, data) => {
        if (err)
            res.status(500).send({
                message:
                    err.message || "Some error occurred while creating the Wallet."
            });
        else res.send(data);
    });
};

exports.findAll = (req, res) => {
    Wallet.getAll(req.query, (err, data) => {
        if (err)
            res.status(500).send({
                message:
                    err.message || "Some error occurred while retrieving wallets."
            });
        else res.send(data);
    });
};
```

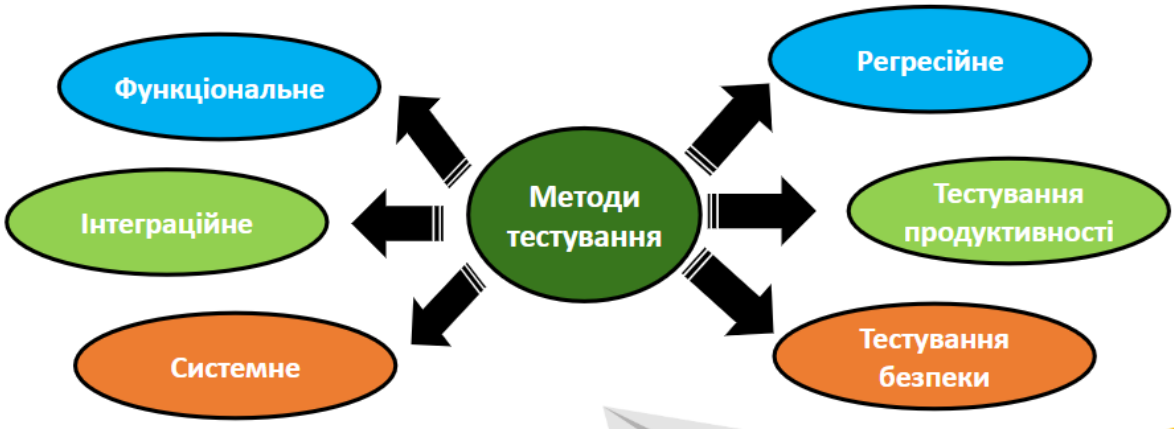
12

Інтерфейс користувача



13

Тестування



14

Переваги використання

- оптимізація процесу управління фінансами домогосподарства;
- зручний інтерфейс для введення та аналізу фінансових даних;
- можливість планування бюджету на основі реальних витрат та доходів;
- підвищення фінансової грамотності користувачів та поліпшення їх фінансового стану.

15

Висновки

- проведено аналіз предметної області управління фінансами домогосподарства;
- розроблена ІС дозволяє користувачам легко та ефективно керувати своїми фінансами. Вона відображає властивості предметної області, забезпечує реалізацію всіх необхідних функцій, забезпечує високу продуктивність і захист даних;
- впровадження ІС у формі веб-додатка сприяє підвищенню фінансової грамотності та покращенню фінансового стану домогосподарств;
- використання у повсякденному житті ІС дозволить більш ефективно вирішувати соціально-економічні проблеми розвитку фінансів домогосподарств, забезпечити гідний рівень і якість життя населення.

16

ДОДАТОК В

Тези доповіді

Матеріали XLVI-ої Міжнародної науково-практичної конференції
(07 липня 2024 року, м.Копенгаген, Данія), дистанційно)

СЕКЦІЯ 11.
ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

Саманцов О.О.
старший викладач кафедри програмної інженерії,
Харківський національний університет радіоелектроніки
м. Харків, Україна

Іванічев В.О.
студент кафедри програмної інженерії,
Харківський національний університет радіоелектроніки
м. Харків, Україна

УПРАВЛІННЯ ФІНАНСАМИ ДОМОГОСПОДАРСТВА З ВИКОРИСТАННЯМ ІНФОРМАЦІЙНОЇ СИСТЕМИ

На поточному етапі розвитку фінансової системи України одним із головних пріоритетів є удосконалення та розвиток фінансів домогосподарств. Фінанси домогосподарств постійно взаємодіють з іншими сегментами фінансової системи держави, що об'єктивно сприяє створенню необхідних умов для подальшого розвитку як фінансового, так і нефінансового секторів економіки, формуванню та розподілу валового внутрішнього продукту (ВВП), а також державних і місцевих бюджетів.

Особливо важлива роль домогосподарств в умовах військового стану та економічної нестабільності, коли їхні раціональні рішення та поведінка безпосередньо впливають на стабільність основних параметрів рівноваги соціально-економічної та фінансової системи. Вони визначають напрями розподілу доходів і їх використання, а також формування заощаджень та їх інвестиційне спрямування.

Сума доходу, необхідного для повного покриття витрат домогосподарства, залежить від потреб його членів, які можуть задовольнятися як матеріальними благами, так і набором послуг. Фінансові ресурси, що здобуваються окремими членами домогосподарства, можуть тимчасово належати індивіду, але згодом використовуються для задоволення потреб усіх членів родини.

Координація економічних інтересів усіх членів домогосподарства є важливим завданням для ефективного функціонування фінансів. Завдяки фінансовим ресурсам, домогосподарства можуть перерозподіляти загальний дохід між своїми членами, а також змінювати частку доходу, що припадає на кожного конкретного члена родини.

Рішення щодо частки доходу, яка припадає на кожного члена домогосподарства, ухвалюються всіма його членами і зазвичай приймаються колегіально, досягнувши консенсусу. Це забезпечує збалансований розвиток усіх сфер функціонування домогосподарства загалом.

Описані фінансові відносини всередині домогосподарства є проявом розподільчої функції фінансів домогосподарств. Завдяки цій функції кожен член домогосподарства забезпечується ресурсами, необхідними для його існування, при цьому об'єктом розподілу є фінансові ресурси, сформовані всіма членами родини.

Для вирішення поставлених задач запропоновано інформаційну систему управління фінансами домогосподарства. Основним принципом цієї системи є розділення доходів і витрат на окремі категорії, що дозволяє більш детально і систематизовано аналізувати фінансовий стан і ефективно планувати бюджет.

Рекомендована кількість категорій витрат становить до 15, включаючи основні аспекти життєдіяльності, такі як транспортні витрати, харчування, охорона здоров'я, одяг, комунальні послуги та зв'язок. Категорії доходів включають заробітну плату, стипендію, пенсію та інші джерела доходу, що забезпечує повне відстеження джерел фінансових надходжень.

Користувач вводить дані про кожну витрату вручну, вказуючи суму і дату операції. Після цього система автоматично сумує витрати по кожній категорії за визначений період часу, що дозволяє отримати повне уявлення про фінансові витрати.

Система дозволяє редагувати дані різними користувачами. Це забезпечує високу точність і достовірність інформації про фінансові операції, враховуючи кожну транзакцію і забезпечуючи актуальність даних. Автоматизація формування бюджету на підставі статистики доходів та витрат в попередніх періодах дозволяє проводити детальний аналіз та планування розвитку домогосподарства та своєчасне виявлення проблем планування.

Основною метою системи є не лише облік, але і аналіз фінансових даних. Користувачі можуть створювати звіти та графіки, які допомагають візуалізувати і аналізувати свої фінансові потоки. Це дозволяє приймати обґрунтовані рішення про витрати, планувати бюджет і ефективно управляти своїми фінансами.

Отже, впровадження інформаційної системи ведення обліку витрат домогосподарства у формі веб-додатка сприяє підвищенню фінансової грамотності та покращенню фінансового стану домогосподарств, забезпечуючи контроль і планування фінансових ресурсів на основі точної та актуальної інформації. Використання у повсякденному житті інформаційної системи управління фінансами домогосподарства дозволить більш ефективно вирішувати соціально-економічні проблеми розвитку фінансів домогосподарств, забезпечити гідний рівень і якість життя населення.