МІНІСТЕРСТВО ОСВІТИ І НАУКА УКРАЇНИ
ВОЛИНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ЛЕСІ УКРАЇНКИ
Кафедра комп’ютерних наук та кібербезпеки

На правах рукопису

Міщук Владислав Михайлович

ПРОЄКТУВАННЯ ТА РОЗРОБКА СЕРВІСУ ДЛЯ ВЕДЕННЯ ПЕРСОНАЛЬНИХ ДОХОДІВ ТА ВИТРАТ

Спеціальність: 122 Комп’ютерні науки
Освітньо-професійна програма: Комп’ютерні науки та інформаційні технології
Робота на здобуття першого рівня «бакалавр»

Науковий керівник:
ГРИШАНОВИЧ ТЕТЯНА ОЛЕКСАНДРІВНА,
доцент кафедри комп'ютерних наук та кібербезпеки, кандидат фізико-математичних наук

РЕКОМЕНДОВАНО ДО ЗАХИСТУ
Протокол №_____ ,
засідання кафедри комп’ютерних наук
та кібербезпеки
від _____ 2025 р.
Завідувач кафедри
(_____) Гришанович Т. О.

ЛУЦЬК-2025

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

API – Application Programming Interface (інтерфейс прикладного програмування)

DOM – Document Object Model (об’єктна модель документа)

PWA – Progressive Web Application (прогресивний вебдодаток)

SDK – Software Development Kit (набір засобів розробника)

SPA – Single Page Application (односторінковий застосунок)

SQL – Structured Query Language (мова структурованих запитів)

БД – база даних

ІТ – інформаційні технології

ВСТУП

У сучасному світі фінансова грамотність і здатність раціонально розпоряджатися власними коштами набувають дедалі більшої важливості. Ведення обліку особистих доходів і витрат дозволяє людям краще розуміти свої фінансові звички, планувати бюджет, виявляти непотрібні витрати та ефективніше досягати фінансових цілей. Особливо актуальним є такий підхід у періоди економічної нестабільності, коли контроль за грошовими потоками стає необхідною умовою фінансової безпеки. Таким чином, персональний фінансовий облік є важливим інструментом самоконтролю та прийняття обґрунтованих рішень. [1,2]

Із розвитком цифрових технологій користувачі дедалі частіше звертаються до вебдодатків як зручного інструменту для автоматизованого обліку фінансів. [3] На відміну від традиційного запису на папері чи у таблицях, такі додатки забезпечують доступ з будь-якого пристрою, автоматичну обробку даних та збереження історії змін. Розробка сучасних вебрішень для ведення бюджету дає змогу користувачам швидко вносити фінансові операції, переглядати статистику та отримувати аналітику у зрозумілому вигляді. Актуальність створення подібного програмного забезпечення пояснюється не лише потребою в обліку, а й високими очікуваннями користувачів до зручності, швидкодії та функціональності сервісів. [4]

Метою роботи є дослідження та розробка сучасного вебдодатку для ведення особистих доходів і витрат, який:

- використовує хмарні сервіси Google Firebase як основу для зберігання, обробки даних і автентифікації користувачів;
- має інтерактивний і зручний інтерфейс користувача на основі бібліотеки готових компонентів Material UI;
- надає можливості для аналізу фінансових операцій та перегляду звітності у зрозумілому вигляді.

Для реалізації програмного засобу заплановано обрано сучасний і популярний стек технологій: для побудови інтерфейсу – бібліотеку React та набір компонентів Material UI, а для реалізації серверної частини – хмарні сервіси Google Firebase. Використання Firebase забезпечує швидку розробку з функціональністю автентифікації, зберігання даних та обробки запитів без необхідності налаштовувати власний сервер. Зі свого боку, React і Material UI дозволяють створити зручний, адаптивний та інтуїтивно зрозумілий інтерфейс, що є критично важливим для повсякденного використання додатку. Якісний інтерфейс знизить бар'єр входу для користувачів та сприятиме регулярному використанню системи для фінансового обліку.[5]

Для досягнення поставленої мети в роботі вирішуються наступні **завдання**:

- провести аналіз існуючих рішень для управління особистими фінансами, визначити їхні переваги та недоліки;
- спроєктувати архітектуру вебдодатку з використанням бібліотеки React.js та хмарної платформи Firebase;
- реалізувати функціональність системи, яка охоплює: облік доходів і витрат, категоризацію, автентифікацію користувачів, формування звітів та аналітики;
- забезпечити стабільну, безпечну та зручну роботу програми шляхом тестування і виправлення можливих помилок;
- підготувати документацію, що супроводжує програмне забезпечення, включаючи опис структури, функціоналу та інструкції для користувача.

Об'єкт дослідження є вебдодатки для ведення обліку персональних витрат і доходів користувачів.

Предметом дослідження є процес проєктування та розробки програмного інструменту для обліку фінансових операцій за допомогою хмарних сервісів Firebase.

РОЗДІЛ 1
ТЕОРЕТИЧНІ ВІДОМОСТІ ДЛЯ РОЗРОБКИ ПРОГРАМНОГО ЗАСОБУ

1.1. Облік особистих фінансів: значення та підходи

В теперішній час управління особистими фінансами є важливою навичкою, що прямо впливає на якість життя, здатність досягати фінансових цілей і уникати боргової залежності. Зростання цін, економічна нестабільність, поширення споживчого кредитування та легкість у здійсненні онлайн-покупок вимагають від людини усвідомленого підходу до розпорядження власними коштами.[6]

Облік доходів і витрат дозволяє людині бачити реальну картину свого фінансового стану. Регулярна фіксація надходжень та витрат допомагає виявити непотрібні витрати, оптимізувати споживання ресурсів, сформувати накопичення на важливі цілі. При цьому особиста фінансова аналітика сприяє формуванню фінансової дисципліни, підвищує обізнаність щодо структури витрат і допомагає приймати обґрунтовані фінансові рішення.[7]

Існують різні підходи до ведення фінансового обліку. Найпростіший – це ручне ведення записів у зошиті або таблицях. Проте з розвитком цифрових технологій набули поширення спеціалізовані програмні інструменти – мобільні та вебдодатки, які автоматизують процес фіксації операцій, візуалізують дані, дозволяють отримувати зручну аналітику.

Особливої популярності в останні роки набули застосунки, що дозволяють вести облік у реальному часі з різних пристроїв, синхронізуючи дані через хмарні сервіси. Це спрощує контроль фінансів для сучасного користувача, що активно використовує інтернет і технології в повсякденному житті.[8]

Таким чином, ведення особистого фінансового обліку є не просто корисною звичкою, а необхідністю, яка сприяє досягненню фінансової стабільності та впевненості в завтрашньому дні.

1.2. Актуальність та переваги вебдодатків для фінансового планування

Із розвитком інформаційних технологій зростає попит на цифрові інструменти, що дозволяють користувачам ефективно організовувати та контролювати особисті фінанси. Вебдодатки посідають особливе місце серед таких інструментів, оскільки поєднують у собі доступність, багатофункціональність та незалежність від конкретного пристрою.

На відміну від десктопних програм або мобільних застосунків, вебдодатки доступні з будь-якого пристрою, що має підключення до інтернету та веббраузер. Це особливо зручно для користувачів, які ведуть облік з декількох пристроїв. Наприклад, на роботі з комп’ютера, вдома зі смартфона або планшета. Зберігання даних у хмарі гарантує постійну актуальність інформації та мінімізує ризики втрати даних через поломку чи втрату пристрою.

Крім того, сучасні вебдодатки можуть бути побудовані за принципами SPA, що забезпечує швидку реакцію інтерфейсу, динамічне оновлення даних без перезавантаження сторінки та загалом наближає користувацький досвід до мобільних застосунків.

Особливої уваги заслуговує можливість легкої інтеграції вебдодатків з хмарними сервісами для автентифікації, зберігання та обробки даних. Це спрощує архітектуру програмного продукту та скорочує час розробки, водночас забезпечуючи масштабованість і безпеку.

Розробка саме веборієнтованих рішень також дозволяє уникати потреби у встановленні програм. Завдяки цьому вебдодатки для фінансового обліку стають доступними ширшій аудиторії – як технічно обізнаним користувачам, так і тим, хто просто прагне зручно контролювати свої фінанси.[9]

Таким чином, вебдодатки є ефективним засобом вирішення задачі ведення обліку доходів і витрат, поєднуючи в собі кросплатформенність,

інтерактивність, безпеку та зручність для кінцевого користувача.

1.3. Огляд технологічного стеку

Для реалізації вебдодатку було обрано сучасний і перевірений стек технологій, який забезпечує ефективну розробку, підтримку, масштабованість і зручність використання. У цьому розділі розглянуто компоненти стеку, які лягли в основу архітектури розробленого рішення.

1.3.1. React.js

React – це бібліотека JavaScript з відкритим вихідним кодом, призначена для створення інтерфейсів користувача, яка підтримується компанією Meta (раніше Facebook) та великою спільнотою розробників. Вона є однією з найпопулярніших технологій для фронтенд-розробки у світі завдяки своїй гнучкості, продуктивності та активному екосередовищу.[10]

Основною ідеєю React є компонентний підхід: інтерфейс користувача складається з незалежних повторно використовуваних компонентів. Це дозволяє легко масштабувати додаток, розбиваючи його на окремі логічні частини, які можна розробляти, тестувати та підтримувати окремо.[11]

React використовує віртуальний DOM – ефективну внутрішню модель, яка дозволяє виявляти мінімальні зміни в інтерфейсі та оновлювати лише ті елементи, які зазнали змін. Це значно покращує продуктивність порівняно з класичним підходом маніпуляції DOM.[12]

У рамках розробки даного проєкту React було обрано як основну технологію фронтенду з кількох причин:

- масштабованість і підтримка складних інтерфейсів. Компонентна архітектура дозволяє легко додавати нову функціональність і розширювати існуючу;

- активна спільнота та екосистема. Велика кількість готових бібліотек (наприклад, Material UI), навчальних матеріалів і документації;
- інтеграція з іншими інструментами. React добре поєднується з хмарними сервісами (як-от Firebase), системами маршрутизації (React Router), бібліотеками керування станом (Redux, Context API) та іншими технологіями.

У додатку React використовується для побудови SPA, що забезпечує швидкий відгук системи та безперервну роботу без перезавантаження сторінки. Компоненти додатку реалізують такі функції, як обробка форми додавання записів, відображення списку витрат, побудова аналітики, авторизація користувача тощо.

1.3.2. Material UI

Material UI (скорочено MUI) – це популярна бібліотека компонентів інтерфейсу користувача для React, створена на основі концепцій Material Design, розроблених компанією Google. Вона надає набір готових стилізованих компонентів, які дозволяють створювати сучасні, естетичні та зручні для користувача інтерфейси без необхідності розробляти дизайн з нуля.[13]

Ключові переваги використання Material UI:

- уніфікований візуальний стиль – компоненти мають єдину дизайн-систему, що гарантує цілісність зовнішнього вигляду додатку;
- гнучкість у кастомізації – розробник може змінювати теми, стилі та поведінку компонентів відповідно до вимог проєкту;
- велика кількість компонентів – форми, таблиці, модальні вікна, сповіщення, меню, панелі навігації та багато іншого;
- адаптивність – підтримка адаптивного дизайну забезпечує зручність користування додатком як на великих екранах, так і на мобільних пристроях.

Завдяки використанню Material UI можна значно пришвидшити процес розробки, досягти високого рівня якості інтерфейсу та забезпечити приємний

користувацький досвід.[14]

1.3.3. Toolpad Core

Toolpad Core – це набір UI-компонентів для швидкої побудови інтерфейсів вебдодатків, створений командою Material UI. Цей інструмент орієнтований на розробників, які прагнуть швидко верстати сторінки свого сайту, що взаємодіє із базами даних, API та іншими джерелами даних. Toolpad Core побудовано на основі React, що робить його сумісним із проєктами, де вже використовується цей фреймворк.

У межах розробки до кваліфікаційної роботи Toolpad Core було використано як допоміжний інструмент для створення інтерфейсу, а також для реалізації шаблонних елементів сторінки аналітики. Його використання дозволило скоротити час на реалізацію складних UI-компонентів і забезпечити більш структуровану побудову SPA.

Таким чином, Toolpad Core виступив ефективним доповненням до Material UI, особливо в контексті розробки зручного інтерфейсу вебдодатку для персонального фінансового обліку.[15]

1.3.4. Firebase

Firebase – це платформа мобільної та веброзробки, створена компанією Google, яка надає низку сервісів для розробників, що спрощують створення повноцінних додатків без потреби у власній серверній інфраструктурі. Firebase дозволяє зосередитися на функціональності додатку, делегуючи питання бекенду – зберігання даних, автентифікацію, хостинг, безпеку – хмарному середовищу.[16]

У межах розробки вебдодатку для ведення доходів і витрат Firebase було обрано як основу серверної частини проєкту через такі причини:

- хмарна архітектура: дозволяє уникнути розгортання власного сервера, забезпечуючи масштабованість і доступність додатку у будь-який момент;
- інтеграція з екосистемою Google: зручна робота з обліковими записами Google, можливість авторизації через Gmail;
- обробка даних в реальному часі: завдяки Firestore, дані оновлюються в інтерфейсі користувача без потреби у ручному перезавантаженні сторінки;
- безпека та контроль доступу: через правила безпеки можна точно визначати, які користувачі мають доступ до яких частин бази даних.

Firebase охоплює декілька ключових сервісів, які використовуються у цьому проєкті, зокрема Firebase Authentication для автентифікації користувачів та Cloud Firestore для зберігання і обробки фінансових записів. Кожен із цих сервісів розглянуто в окремих підпунктах.

Firebase також надає зручний вебінтерфейс для моніторингу активності користувачів, перевірки структури бази даних та налагодження правил доступу, що особливо корисно під час розробки, тестування та обслуговування додатку.

Таким чином, Firebase виступає не просто бекенд-рішенням, а повноцінною платформою для швидкої, безпечної та масштабованої розробки вебдодатків.

1.3.5. Firebase Authentication

Firebase Authentication – це сервіс, що надає зручний та безпечний спосіб автентифікації користувачів у додатках побудованих на платформі Firebase. Його основна мета – спростити процес входу в систему для розробників та забезпечити надійний захист даних користувачів.[17]

Firebase Authentication підтримує кілька варіантів автентифікації, серед яких:

- вхід за допомогою електронної пошти та пароля;
- вхід через облікові записи Google, Facebook, GitHub, Apple та інші провайдери;
- анонімна автентифікація (наприклад, для ознайомлення з додатком без створення акаунту).

У межах цього проєкту реалізовано автентифікацію користувачів за допомогою електронної пошти та пароля і за допомогою аккаунту Google, оскільки ці способи є простими для реалізації та зрозумілим для користувачів. Завдяки вбудованому механізму Firebase, облікові записи користувачів зберігаються у безпечному середовищі, а перевірка їхньої автентичності відбувається на стороні серверів Google.

Ключові переваги використання Firebase Authentication:

- безпека: збереження паролів і обробка облікових даних здійснюються на рівні Google, що зменшує ризики витоку даних;
- швидка інтеграція: сервіс легко інтегрується з React-додатками завдяки офіційним SDK;
- підтримка сесій: механізми автоматичного збереження стану входу/виходу користувача;
- гнучкість: можливість поєднання з іншими сервісами Firebase, зокрема Firestore для зберігання персоналізованих даних користувача.

Firebase Authentication забезпечує простий, надійний і безпечний механізм ідентифікації, що є важливою складовою будь-якого додатку, пов’язаного зі зберіганням персональних даних.

1.3.6. Firebase Firestore

Firestore – це гнучка та масштабована хмарна база даних NoSQL, яка є частиною платформи Google Firebase. Вона особливо добре підходить для вебдодатків, що працюють із динамічними даними, зокрема у фінансових

системах.

Firestore побудована на моделі колекцій і документів – це аналог таблиць і записів у класичних SQL-базах. Така структура дозволяє організувати дані у вигляді деревоподібної ієрархії, що дає гнучкість у зберіганні складних і вкладених даних без потреби у жорстко фіксованій схемі.[18]

Серед ключових переваг Firestore:

- реальний час. Будь-які зміни в базі одразу відображаються в інтерфейсі користувача без потреби в оновленні сторінки. Це надзвичайно зручно у фінансових додатках, де важливо бачити поточний стан рахунків і операцій;
- висока швидкодія. Кожен запит у Firestore автоматично індексується, що забезпечує миттєвий доступ до даних, навіть за значного обсягу інформації;
- безпека. Firestore підтримує інтеграцію з Firebase Authentication, що дозволяє налаштувати контроль доступу до даних на основі автентифікації користувачів. Це гарантує конфіденційність персональної фінансової інформації;
- масштабованість. Firestore здатна адаптуватися до зростання навантаження без втрати продуктивності, що важливо як для індивідуальних, так і для командних фінансових сервісів.

У межах розробленого вебдодатку Firestore використовується для зберігання транзакцій, категорій, обліку рахунків і зв’язку цих даних з конкретним користувачем. Таким чином, Firestore забезпечує надійну основу для обліку доходів і витрат у режимі реального часу.

1.4. Огляд аналогічних рішень

В мережі не так легко знайти приклади саме трекерів доходів і витрат з доступною вебверсією і відносною популярністю. Більшість існуючих рішень є

продуктами IT-компаній, де доступ отримується через платні підписки. Розглянемо одного з лідерів в галузі додатків для персонального менеджменту фінансів та інді-проєкт з PWA версією та відкритим кодом.

1.4.1. Credit Karma

Credit Karma – це не просто додаток для ведення доходів і витрат. Це платформа орієнтована на поліпшення кредитної історії. Тобто цільовою аудиторією застосунку є люди, котрі часто користуються позиками.[19]

Credit Karma є продуктом компанії Intuit, що має досвід в розробці технологічних рішень для персонального обліку фінансів. Їхнім попереднім продуктом був менеджер фінансів Mint. Він був дуже популярним та мав на піку 3.6 мільйона активних користувачів. Але наприкінці 2024-го року, після 17-ти років роботи, проєкт закрили і поставили в пріоритет розвиток Credit Karma, так як вона була більш прибуткова.

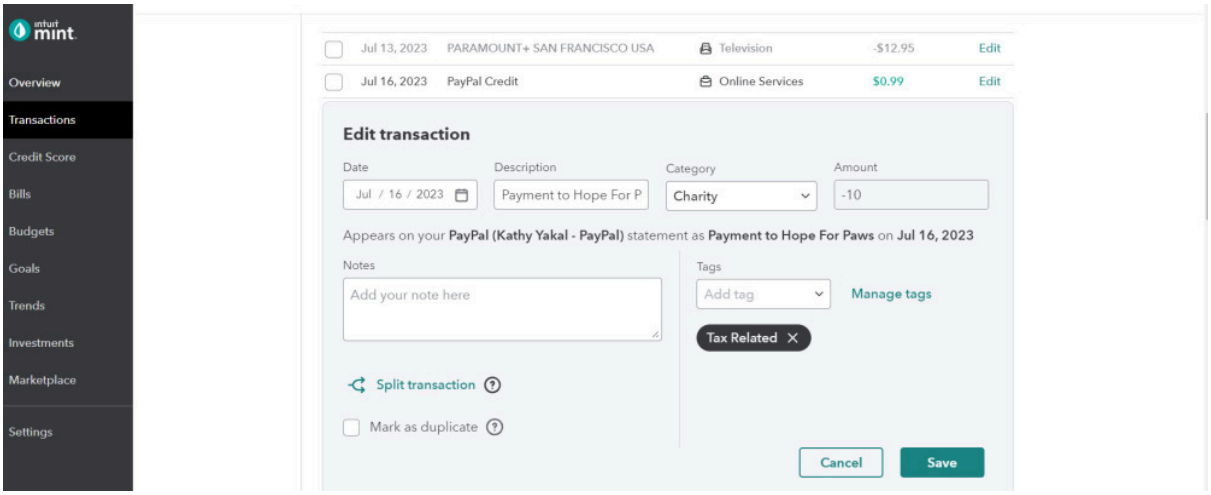

Рисунок 1.1 – Інтерфейс десктопної версії Mint

Такі фінансові менеджери часто містять у собі механізми імпорту та експорту даних користувача. Після закриття Mint і переходу на Credit Karma частина користувачів, яких не цікавив менеджмент позик, перейшли на інші

системи з подібним фунціоналом. Навіть зараз у багатьох менеджерах фінансів доступна можливість імпорту даних з файлів які були сформовані користувачем у Mint.

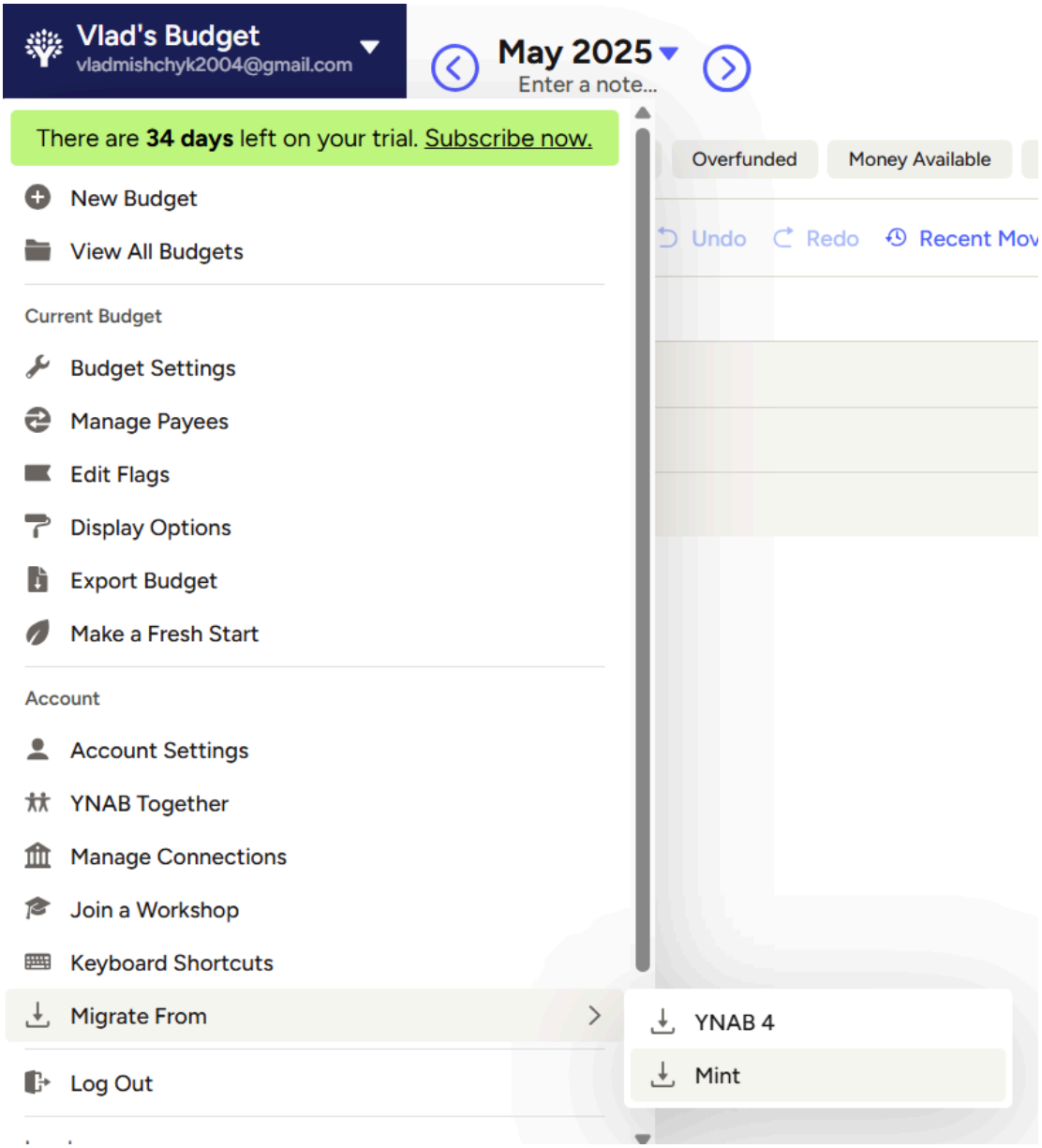

Рисунок 1.2 – Менеджер фінансів YNAB дозволяє імпортувати файли, сформовані у Mint

Credit Karma працює на ринках США, Великої Британії та Канади. В даних країнах такий застосунок дійсно актуальний, адже усі вони знаходяться в топі країн з найвищим середнім кредитним боргом на душу населення.[20]

У програмі є багато інтеграцій з різними установами, які надають позики. Доступні списки пропозицій, можна відфільтрувати найвигідніші пропозиції. Система працює так, що у користувача є рейтинг, який відомий різним установам-позиконадавачам. Тож з кращою історією користувач отримує кращі пропозиції.

Даний продукт демонструє наскільки далеко може вийти система персонального ведення бюджету.

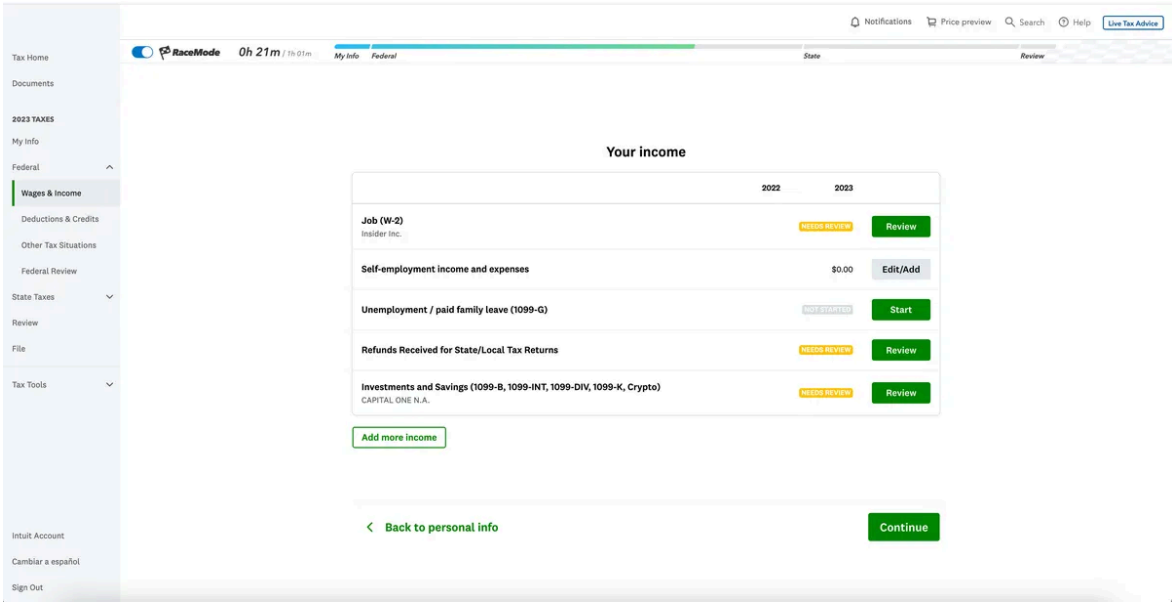

Рисунок 1.3 – Приклад вікна інтерфейсу Credit Karma

1.4.2. Cashew

Cashew – кросплатформерний додаток, створений для того, щоб допомогти користувачам ефективно керувати своїми фінансами. Ця програма, створена за допомогою Flutter. Як базу даних було використано SQL Drift, також

авторизація працює завдяки сервісам Firebase. Додаток забезпечує зручну та інтуїтивно зрозумілу роботу на різних пристроях.[21]

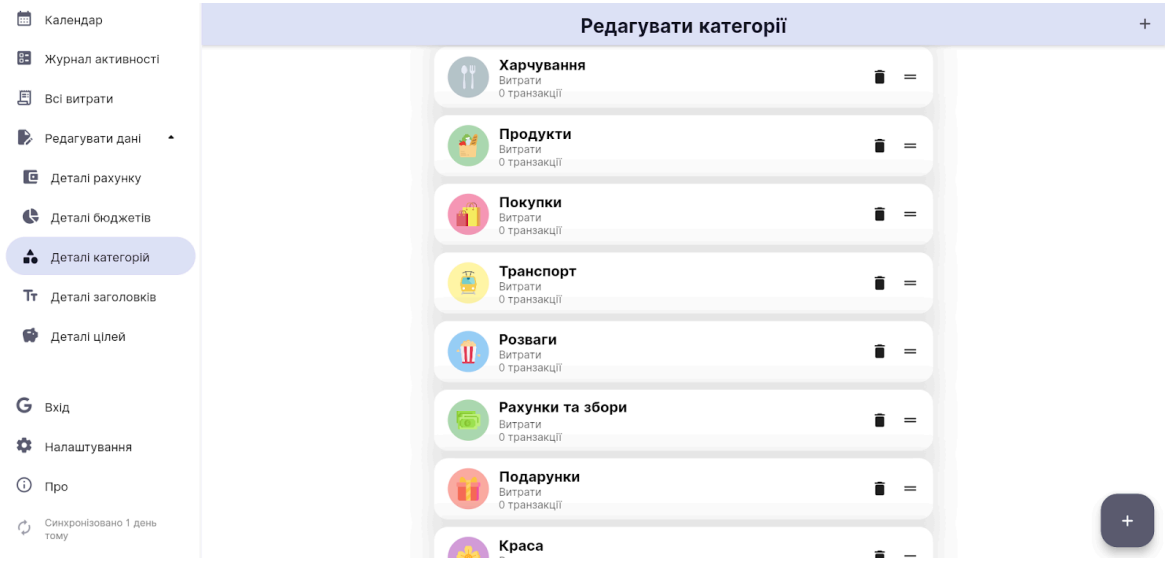

Рисунок 1.4 – Інтерфейс вебверсії Cashew

На момент написання даної роботи додаток має понад сто тисяч завантажень лише для Android користувачів.[22]

1.4.3. Висновок аналізу подібних рішень

На сучасному ринку існує велика кількість додатків для фінансового обліку, однак чимало з них мають спільні недоліки, які можуть стати бар'єром для користувачів. Один із найпоширеніших – відсутність повноцінного безкоштовного доступу. Більшість сервісів надають лише обмежену кількість функцій у безоплатній версії або пропонують короткостроковий пробний період. Після його завершення користувач змушений оформлювати платну підписку, навіть якщо йому потрібна лише базова функціональність. У результаті користувачі нерідко переплачують за функції, якими фактично не користуються.

Іншою проблемою є складна початкова конфігурація додатків. Щоб розпочати роботу, часто необхідно пройти багатоетапний процес налаштувань,

що вимагає часу та технічної обізнаності. Для недосвідчених користувачів це може стати демотивуючим фактором і звести нанівець бажання користуватися системою.

З огляду на ці недоліки, при створенні власного рішення було поставлено мету – зробити інструмент для ведення фінансів максимально простим і зручним. Важливим принципом стала миттєва готовність до використання: одразу після реєстрації користувач може додавати доходи й витрати без жодних додаткових кроків з налаштування. Такий підхід дозволяє уникнути перевантаження інтерфейсу, забезпечує легкий старт і робить додаток привабливим для широкої аудиторії – незалежно від рівня підготовки чи попереднього досвіду.

РОЗДІЛ 2
РОЗРОБКА ДОДАТКУ ДЛЯ ВЕДЕННЯ ПЕРСОНАЛЬНИХ ДОХОДІВ І ВИТРАТ

2.1. Постановка задачі, призначення та вимоги до вебдодатку

Метою розробки є створення сучасного вебдодатку для ведення особистих доходів і витрат, який забезпечує користувачам зручний, безпечний і функціональний інструмент для щоденного фінансового обліку. Програмний засіб має бути простим у використанні, мати зрозумілий інтерфейс, не вимагати складного налаштування при першому запуску та забезпечувати базову аналітику для прийняття обґрунтованих фінансових рішень.

Перед розробкою стояло завдання реалізувати застосунок, який би:

- мав зрозумілий інтерфейс без складних початкових налаштувань;
- дозволяв швидко почати користування відразу після реєстрації;
- підтримував категоризацію транзакцій для зручного сортування;
- забезпечував обробку та зберігання даних у реальному часі з використанням Firebase Firestore;
- реалізовував захищену автентифікацію через Firebase Authentication;
- був адаптований для роботи у браузері без необхідності встановлення;
- мав адаптивний інтерфейс для користування з різних пристроїв;
- надавав можливість перегляду загальної фінансової статистики за період.

Призначенням програмного засобу є допомога користувачам у щоденному веденні особистих фінансів, підвищенні фінансової грамотності та плануванні витрат за допомогою простого й візуально зрозумілого інструменту.

Функціональні вимоги до вебдодатку:

- реєстрація та вхід користувача через електронну пошту, або Google аккаунт;
- додавання, редагування, видалення фінансових операцій;
- вибір категорії для кожної операції;
- аналітика фінансів в розрізі рахунків;
- перегляд загального балансу та суми доходів/витрат за місяць;
- аналіз витрат в розрізі загальних напрямків;
- динамічне оновлення інформації при зміні даних у базі;
- генерація простої візуалізації для аналітики.

Вимоги до інтерфейсу користувача:

- адаптивний, зручний інтерфейс на основі Material UI;
- підтримка темної та світлої теми;
- зрозуміле групування функцій;
- доступність усіх функцій у кілька кліків, без необхідності переходити на інші сторінки;
- інтуїтивно зрозуміле введення даних.

Таким чином, результатом розробки має стати сучасний вебдодаток, орієнтований на користувача, який поєднує зручність використання з достатньою функціональністю для щоденного контролю фінансів.

2.2. Вибір моделі та методології розробки

Для реалізації програмного засобу було обрано спіральну модель розробки програмного забезпечення, яка поєднує елементи каскадного підходу з ітеративністю та фокусом на управлінні ризиками. Ця модель особливо добре підходить для проєктів, у яких очікується поступове зростання функціональності, а також потреба в оцінці й усуненні ризиків на ранніх етапах.

Розробка вебдодатку для ведення доходів і витрат відбувалась поетапно, через послідовні ітерації. Кожен цикл охоплював планування, аналіз ризиків,

реалізацію функціоналу, тестування та оцінку результатів. Після кожної ітерації приймалося рішення про наступні кроки: розширення функціоналу, вдосконалення інтерфейсу або усунення виявлених проблем.

Такий підхід дозволив [23]:

- гнучко реагувати на виявлені недоліки або потреби користувача;
- поступово розширювати можливості додатку, не порушуючи вже реалізовану стабільну частину;
- зменшити ймовірність серйозних помилок завдяки багаторазовому тестуванню на кожному етапі;

Для мінімізації можливих недоліків спіральної моделі в процесі розробки були впроваджені наступні організаційні рішення:

- формулювання чітких цілей кожної ітерації, що дозволяло уникнути розмитості задач і зосередитися на ключовій функціональності;
- регулярне тестування та рефакторинг після завершення кожного циклу, що сприяло стабільності й надійності вебдодатку;
- аналіз ризиків перед кожною ітерацією, з подальшою розробкою заходів щодо їх мінімізації.

Окрім вибору моделі життєвого циклу, важливим аспектом стало управління робочим процесом. Для цього було обрано методологію Kanban, яка забезпечила візуалізацію завдань, зручне планування та контроль виконання. Усі етапи роботи відображались у вигляді інтерактивної дошки з колонками («заплановано», «в роботі», «готово»), що дозволяло контролювати навантаження й ефективно керувати часом.[24]

Об'єднання переваг спіральної моделі з гнучким керуванням задачами в Kanban дозволило ефективно організувати процес створення вебдодатку з дотриманням принципів якості, стабільності та зручності для кінцевого користувача.

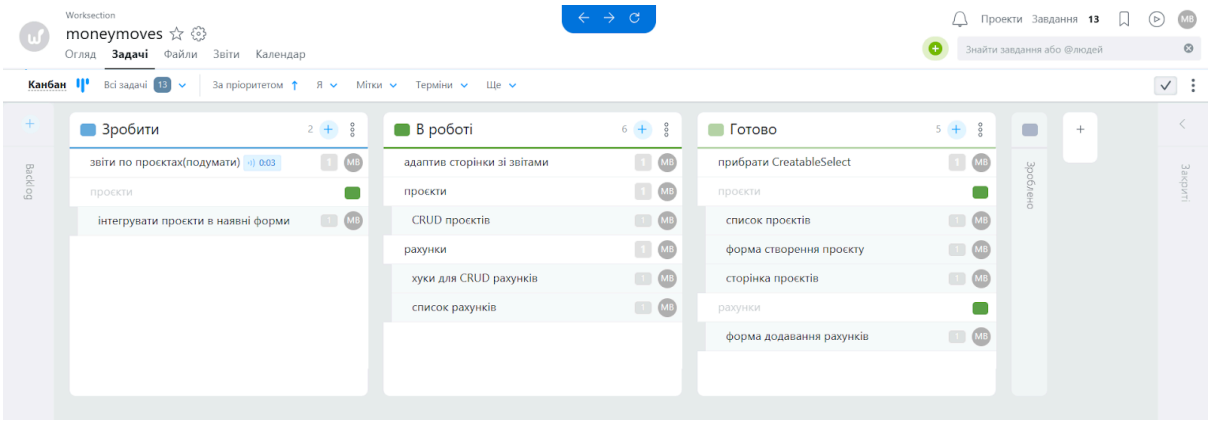

Рисунок 2.1 – Канбан розробки нових сторінок для вебдодатку

2.3. Загальний опис проєкту

Проєкт являє собою односторінковий вебдодаток (SPA), призначений для обліку особистих фінансових операцій – доходів та витрат. Основна мета системи полягає у наданні користувачам простого, інтуїтивно зрозумілого та функціонального інструменту для ведення бюджету, аналізу фінансової активності та формування звітності. У додатку реалізовано механізми автентифікації, категоризації транзакцій, перегляду історії операцій, побудови графічних звітів, а також захисту й зберігання даних за допомогою хмарних сервісів Firebase. Ключовими технологіями розробки виступають React, Material UI та Firestore.

Уся бізнес-логіка реалізована на клієнтській стороні за допомогою бібліотеки React, а для керування станом, зберігання та обробки даних використано хмарні сервіси Google Firebase.

Вебдодаток складається з кількох функціональних модулів, зокрема:

- реєстрація та авторизація користувачів;
- внесення та редагування грошових операцій;
- управління категоріями;
- робота з рахунками та проєктами витрат;

- формування статистичних звітів та аналітики у графічній та табличній формах.

Інтерфейс користувача реалізовано у вигляді шести основних екранів, кожен з яких відповідає окремому маршруту додатку:

- сторінка авторизації з полями для введення електронної пошти та паролю, а також кнопками «Зареєструватись» та «Зареєструватись через Google»;
- список фінансових операцій із можливістю додавання, редагування й видалення записів;
- сторінка категорій доходів і витрат;
- сторінка рахунків;
- сторінка проєктів витрат;
- сторінка статистики, яка відображає аналітичні дані за періодами у вигляді діаграм.

Коли користувач відкриває сайт, система спочатку перевіряє, чи він вже авторизований. Якщо автентифікація пройдена успішно, користувачу надається доступ до основного функціоналу вебдодатку. У разі відсутності авторизації автоматично відображається форма входу в систему(рис 2.2).

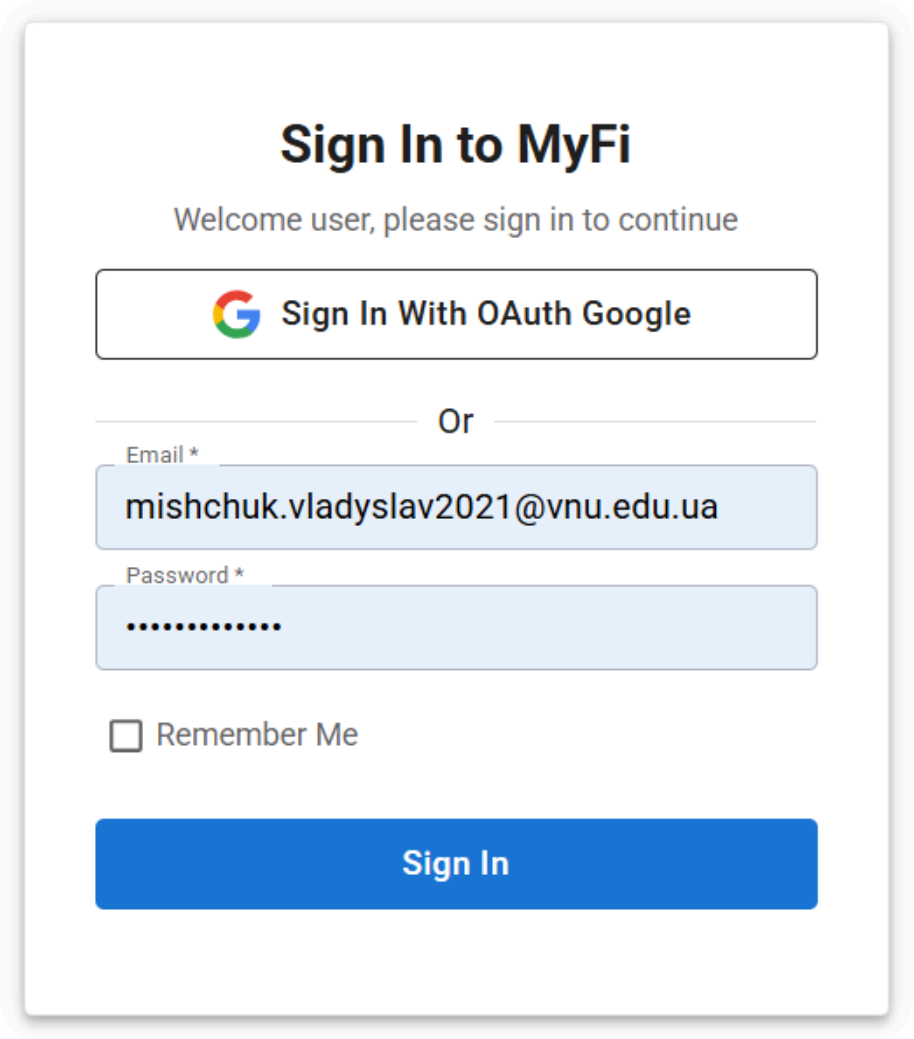

Рисунок 2.2 – Вікно реєстрації з двома варіантами авторизації

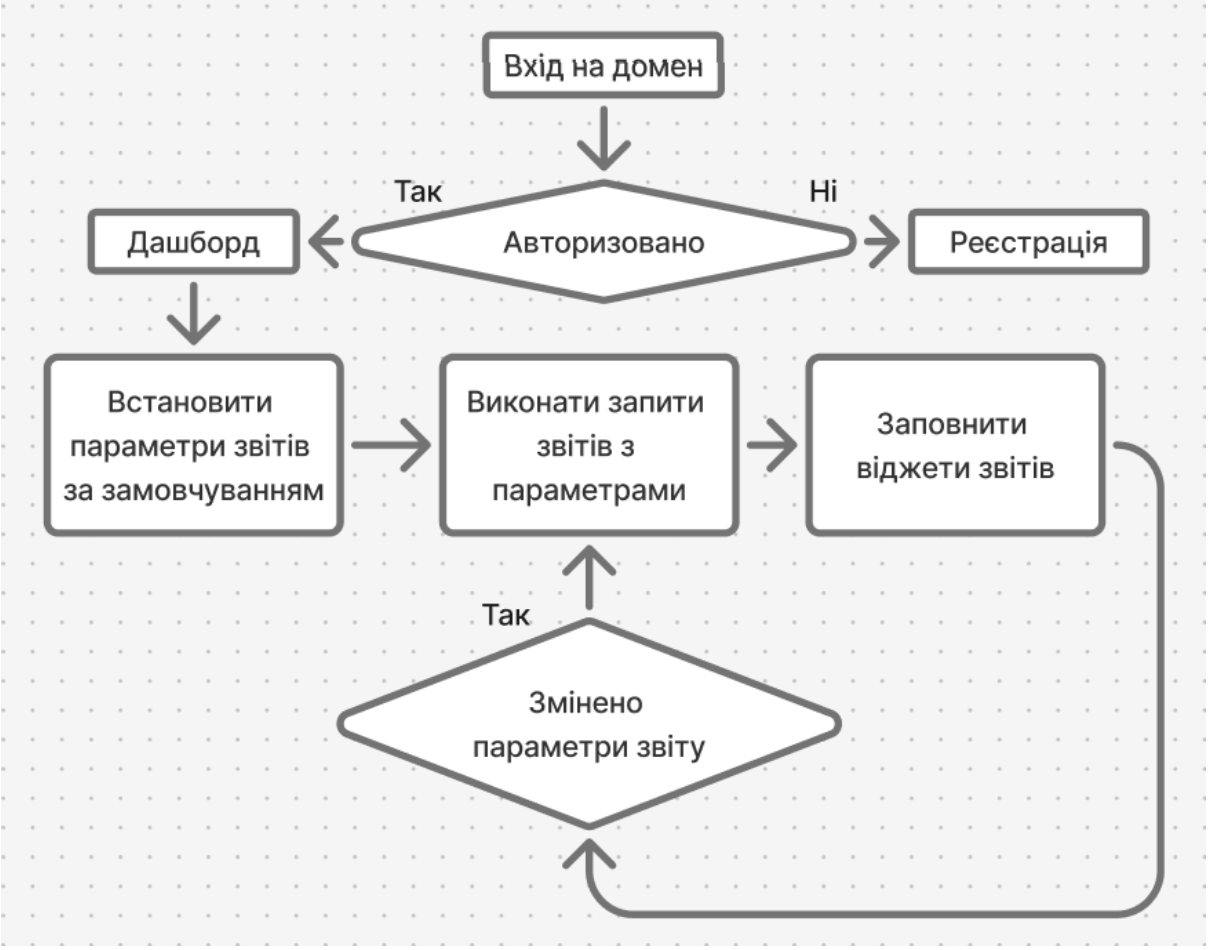

Рисунок 2.3 – Алгоритм, з яким зустрічається користуваа при відвідуванні домену

На рисунку 2.3 зображено алгоритм роботи із користувачем. У разі вдалої авторизації користувач перенаправляється на сторінку з аналітикою. Дані за замовчуванням відображають стан поточного та попереднього місяців. Юзер має змогу відредагувати періоди та інші параметри звітів у відповідних для цього полях (рис. 2.4).

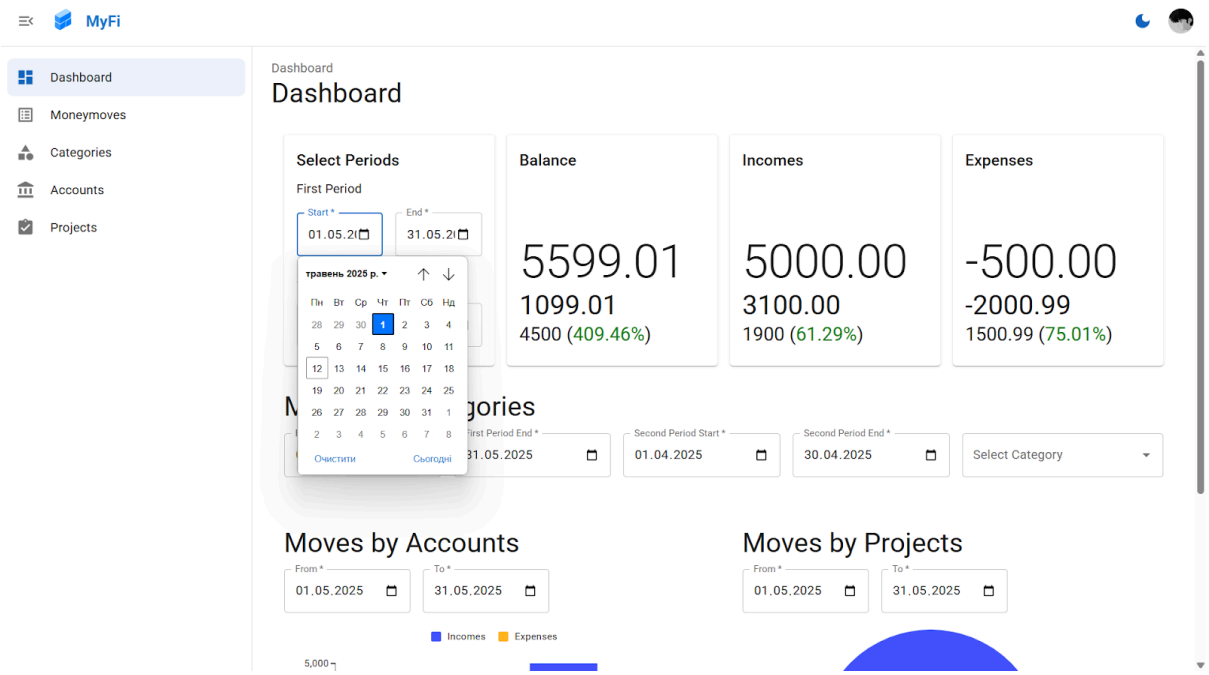

Рисунок 2.4 – Дашборд який бачить користувач після авторизації

Звідси можна перейти на інші сторінки сайту, що доступні в бічній панелі навігації, або змінити тему чи обліковий запис (рис. 2.5).

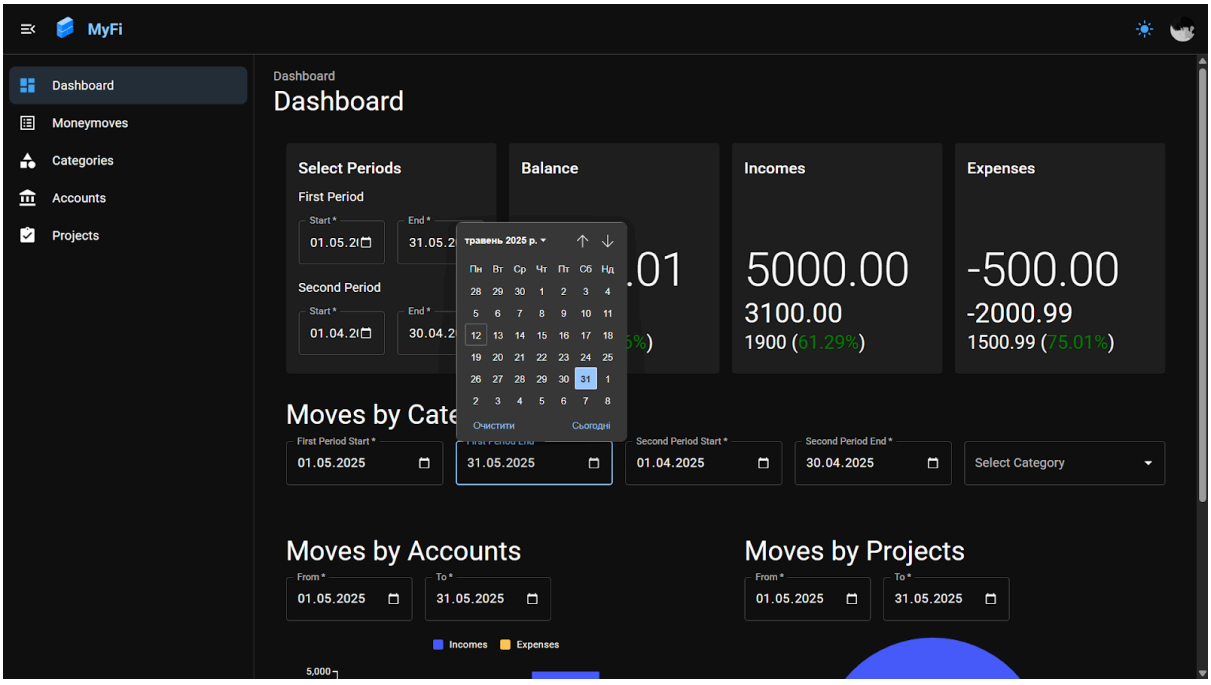

Рисунок 2.5 – Демонстрація темної теми сайту

На рисунку 2.6 показано один з доступних віджетів дашборду – звіт з рухами коштів за певний період в розрізі рахунків. Він складається з форми для вказання періоду та діаграми. Звіт оновлюється автоматично зі зміною якоїсь з дат періоду.

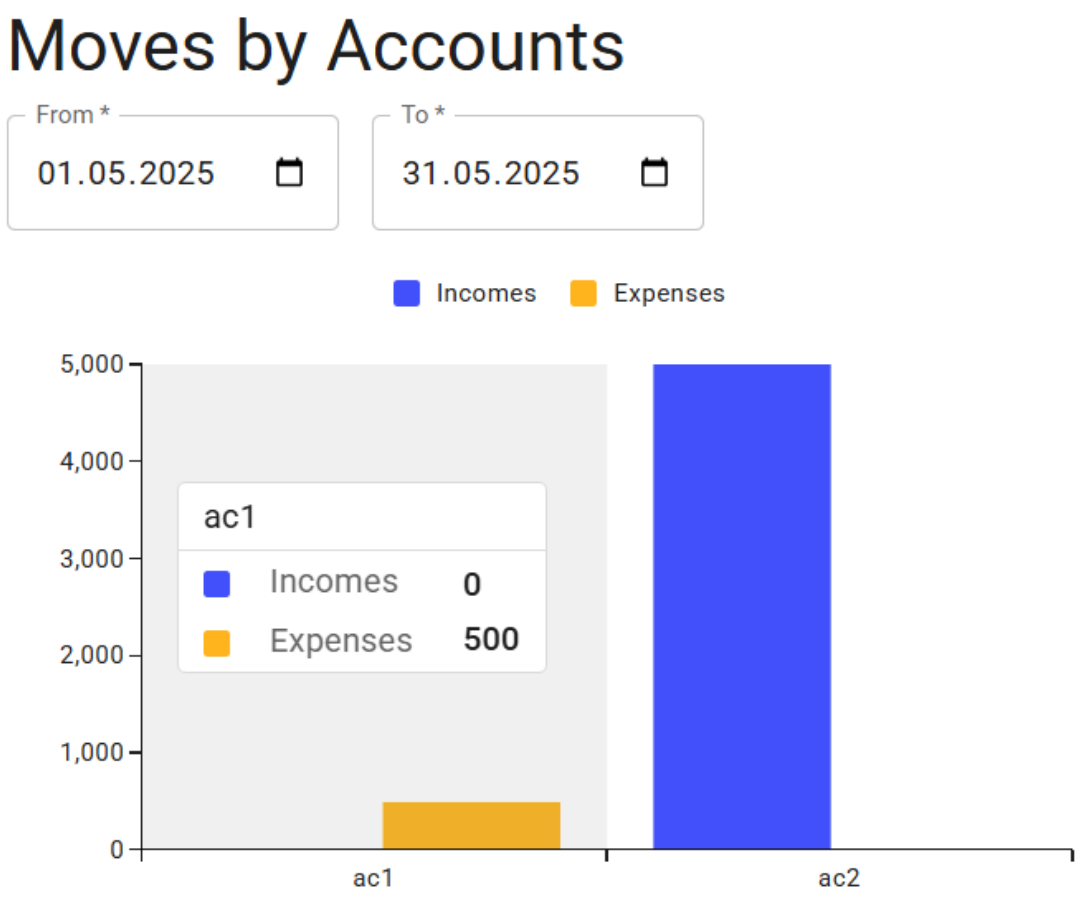

Рисунок 2.6 – Діаграма порівняння об’ємів рухів коштів за рахунками

Далі розглянемо сторінку додавання записів про рух коштів. Тут є форма для додавання нових записів, та список внесених раніше даних (рис. 2.7). Поля таблиці є інтерактивними. Змінивши дані в комірці таблиці – запис оновиться в базі даних. На рисунку 2.8 зображено алгоритм сторінки з витратами та доходами.

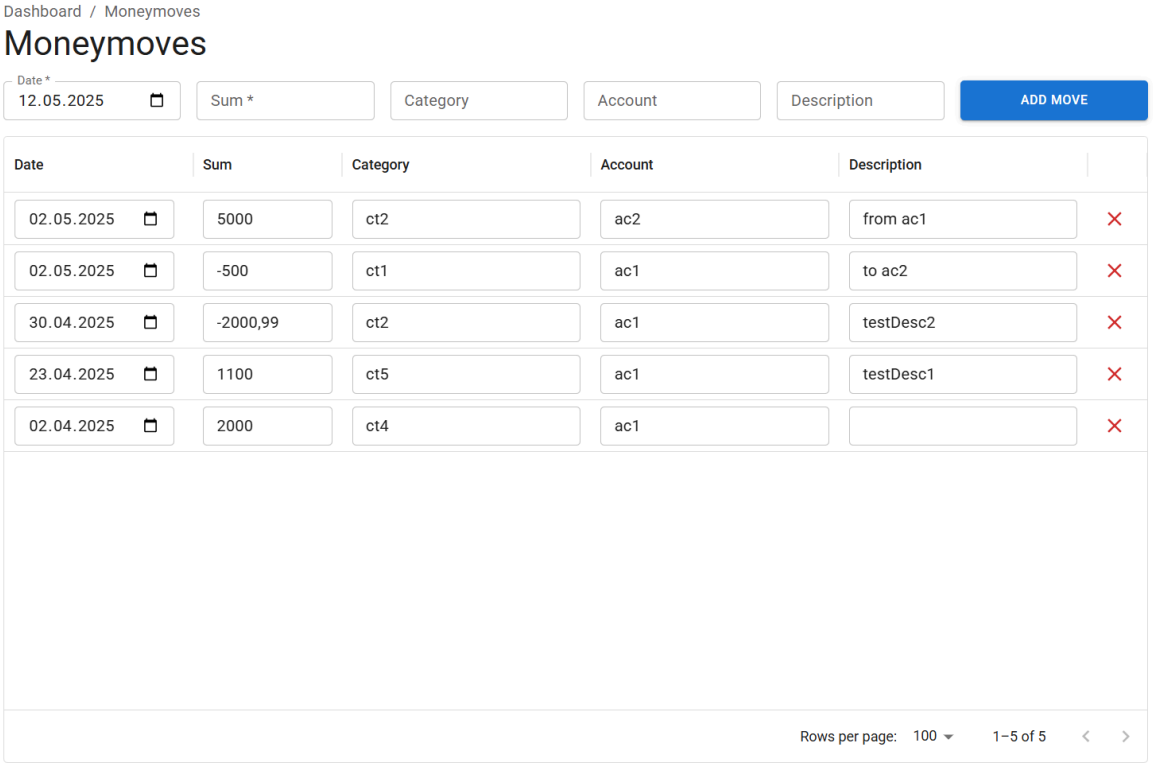

Рисунок 2.7 – Вигляд сторінки для роботи з записами про рух коштів

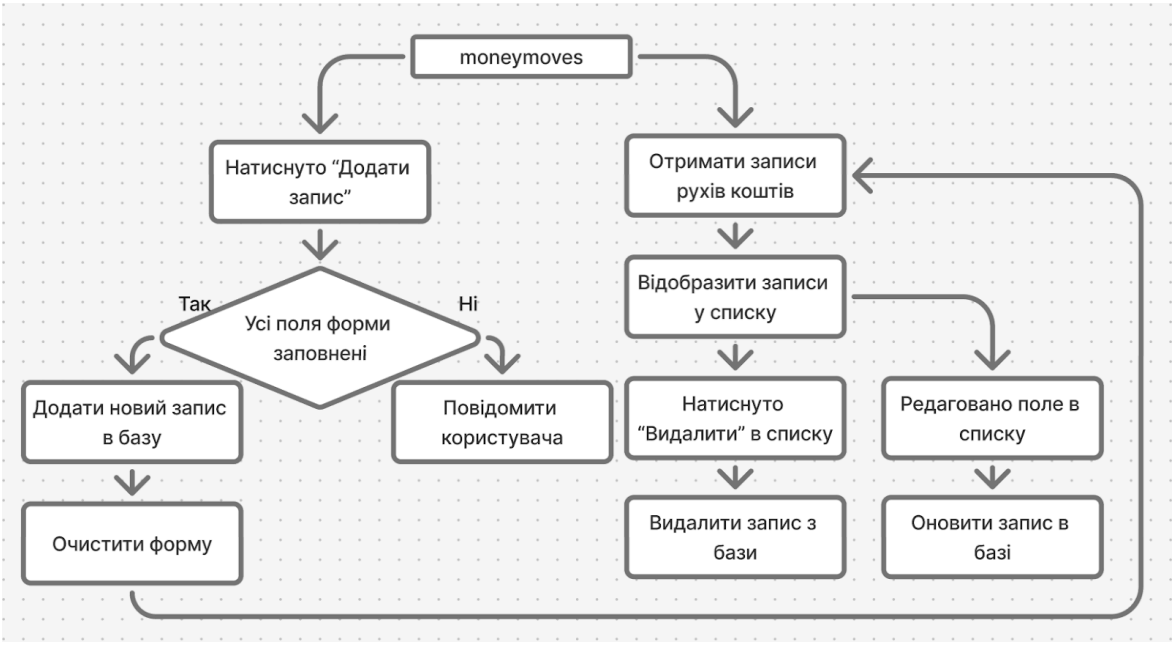

Рисунок 2.8 – Алгоритм, що працює на сторінках зі списками

Решта вкладок сайту(категорії, рахунки, проєкти) мають аналогічну структуру. На сторінках категорій і проєктів є поля для заповнення періоду за який порахувати об’єм руху коштів. Також усі вікна сайту є адаптивними під різні розміри пристроїв. На рисунку 2.9 показано як виглядає вікно категорій на мобільному пристрої.

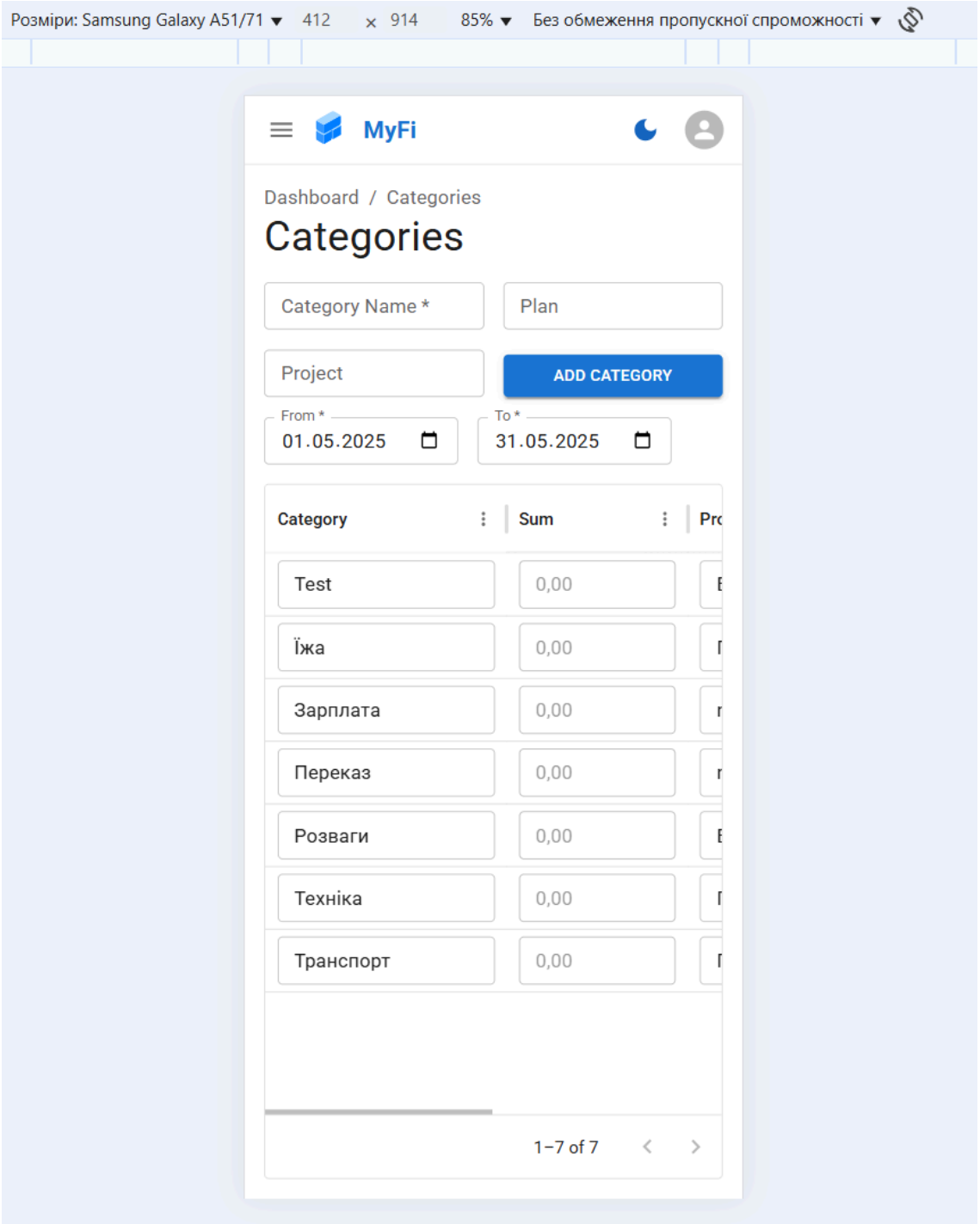

Рисунок 2.9 – Вигляд сторінки для роботи з категоріями витрат і доходів

Переміщуючись сторінками сайту можна помітити, що за різними посиланнями перемальовується лише частина з контентом. Навігація і верхнє меню є статичним (окрім того, що воно адаптивне) і однаковим на усіх сторінках(рис. 2.10). В сьогоденні це є досить популярним рішенням дизайну для SPA.

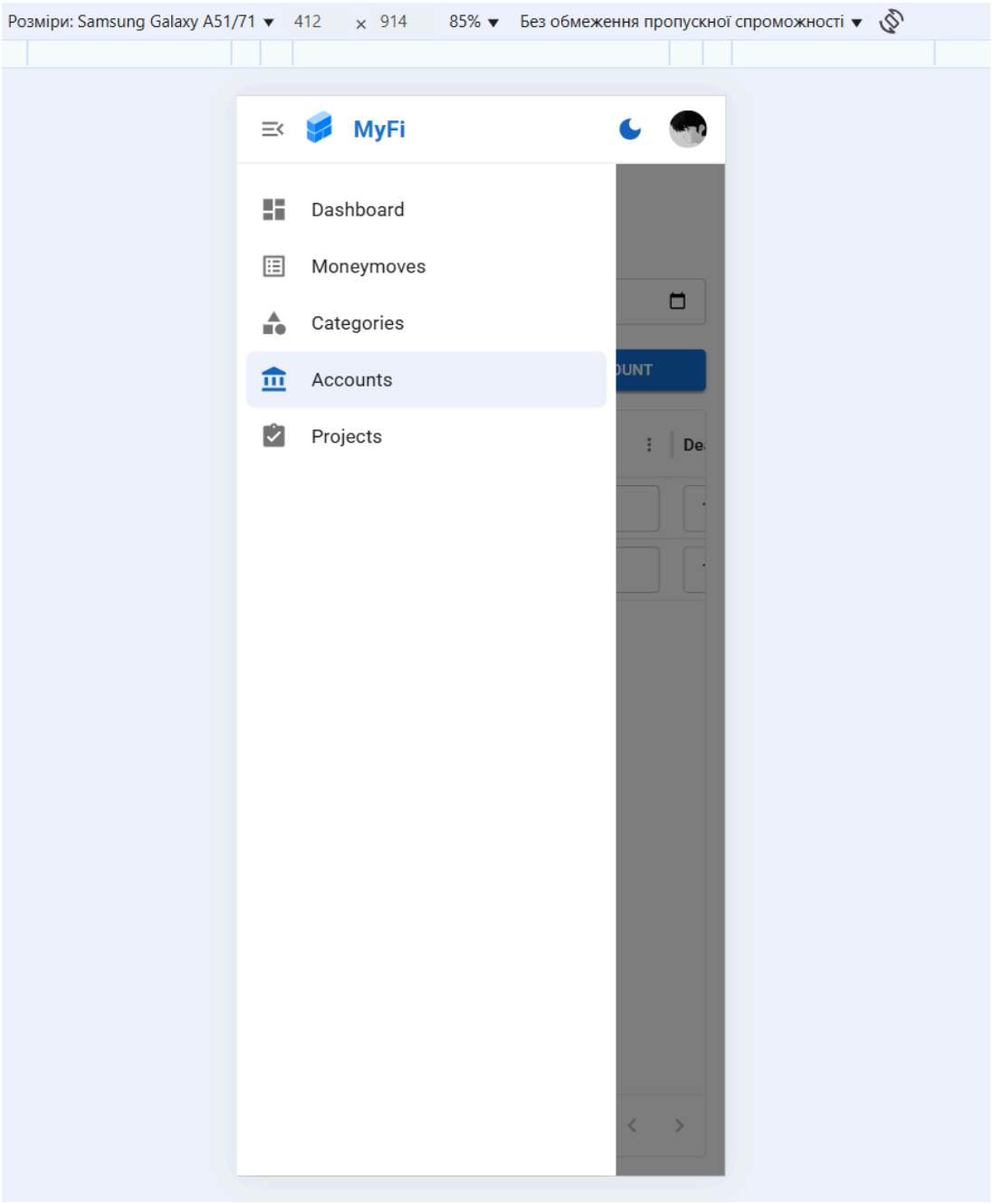

Рисунок 2.10 – Реалізація навігації по сайту на мобільних пристроях

2.4. Вибір та обгрунтування інструментальних засобів розробки

Для створення вебдодатку для обліку доходів і витрат були обрані сучасні технології, які забезпечують ефективну розробку, стабільну роботу та високу продуктивність системи. Розробка передбачає наявність зручного інтерфейсу користувача, а також сховища даних для збереження фінансових операцій. Крім того, необхідним є використання серверного функціоналу для реалізації механізму автентифікації користувачів.

React – це JavaScript-бібліотека для створення інтерфейсів користувача, яка дозволяє ефективно реалізовувати односторінкові вебдодатки, де оновлення контенту відбувається без повного перезавантаження сторінки. Це забезпечує швидку та плавну взаємодію з користувачем, а також високу продуктивність і зручне масштабування.

React має величезну спільноту розробників і активну екосистему, що включає численні додаткові модулі, бібліотеки та інструменти, які значно прискорюють та спрощують розробку. Саме тому React на сьогоднішній день є найпопулярнішою бібліотекою для фронтенд-розробки у JavaScript-середовищі.

Ключовою особливістю React є компонентний підхід до побудови інтерфейсу. Кожен елемент сторінки реалізується у вигляді окремого компонента, який може приймати параметри, реагувати на зміни стану та повторно використовуватися в різних частинах додатку. Це сприяє модульності, читабельності коду та спрощує його підтримку.

Завдяки реактивним хукам (наприклад, useState, useEffect тощо) React дозволяє легко керувати внутрішнім станом компонентів та реалізовувати складні сценарії поведінки при зміні даних. Також, завдяки JSX-синтаксису, розмітка сторінки пишеться у зрозумілій і наближеній до HTML формі, що підвищує швидкість розробки і знижує поріг входження.[25]

Ще однією перевагою є вбудована оптимізація повторного рендерингу: React оновлює лише ті елементи DOM, у яких дійсно відбулися зміни, що

позитивно впливає на продуктивність навіть у складних інтерфейсах з великою кількістю динамічних елементів.

Усі ці характеристики роблять React.js надійним та ефективним інструментом для створення клієнтської частини сучасного вебдодатку, саме тому його було обрано для реалізації цього проєкту.

Для реалізації серверної частини вебдодатку було обрано хмарну платформу Firebase, яка надає потужний набір інструментів для розробки сучасних кросплатформних застосунків. Вона дозволяє зосередитись на функціональності продукту без необхідності самостійно налаштовувати інфраструктуру, оскільки більшість серверних рішень уже надаються «з коробки» і обслуговуються надійними серверами Google.

Однією з ключових переваг Firebase є інтеграція її внутрішніх сервісів, необхідних для створення повноцінного застосунку. У межах цього проєкту використовуються два основні інструменти: Firestore та Authentication.

Firestore – це гнучка масштабована база даних, побудована за принципом NoSQL. Вона не використовує класичну табличну модель, як у реляційних СУБД, натомість оперує колекціями та документами, що робить структуру зберігання даних більш гнучкою. Додатковою перевагою Firestore є можливість налаштування прав доступу до окремих колекцій і документів, що підвищує безпеку даних. Також платформа дозволяє використовувати тригери для реалізації певної логіки на боці сервера.

Firebase Authentication – це сервіс, який дозволяє швидко реалізувати систему автентифікації користувачів з підтримкою електронної пошти, пароля, а також сторонніх провайдерів, таких як Google, Facebook тощо. Цей інструмент значно спрощує реалізацію безпечного входу в систему та управління сесіями користувачів.

Вибір Firebase для цього проєкту обумовлений його простотою інтеграції, високою надійністю, можливістю масштабування, а також наявністю зручного інтерфейсу для адміністрування та гнучкими засобами контролю доступу. У

результаті розробник отримує повноцінну серверну платформу з готовими рішеннями для зберігання даних та управління користувачами, що дозволяє зосередитись на створенні функціональності клієнтської частини.[26]

Material UI (MUI) – набір компонентів інтерфейсу користувача, побудований на базі дизайну Google Material Design. Завдяки цій бібліотеці міжливо швидко реалізувати інтуїтивно зрозумілий та адаптивний інтерфейс, який добре сприймається користувачами на різних пристроях. Це найпопулярніша і найбільш універсальна бібліотека готових компонентів інтерфейсу для Javascript.

MUI Toolpad Core – набір інструментів для побудови вебдодатків на основі MUI, який дозволяє створювати адмініструвальні інтерфейси, діаграми, таблиці та інші візуальні компоненти з мінімальними витратами часу. Він ідеально підходить для реалізації аналітики та фінансових звітів, які є важливою частиною функціональності розробленого додатку.

Вибрані технології добре інтегруються між собою, є підтримуваними, задокументованими та активними в професійному середовищі. Такий стек дозволив швидко та ефективно реалізувати функціональність вебдодатку з урахуванням сучасних вимог до безпеки, зручності та продуктивності.

2.5. Особливості програмної реалізації

Перед початком роботи потрібно ініціалізувати проєкт. Ми будемо використовувати Toolpad Core тому створення проєкту відбувається за допомогою команди показаної на рисунку 2.11.

```
PS D:\VNU\diploma> npx create-toolpad-app
Need to install the following packages:
create-toolpad-app@0.15.0
Ok to proceed? (y) y

✓ Enter path of directory to bootstrap new app .
✓ Which framework would you like to use? Vite
✓ Would you like to enable authentication? yes
✓ Select authentication providers to enable: Google
```

Рисунок 2.11 – Команда ініціалізації проєкту та початкових налаштувань

Після запуску команди потрібно буде пройти початкові налаштування. Назвати проєкт, вибрати фреймворк для роботи з React.js, підключити Firebase до проєкту та обрати провайдерів автентифікації.

На наступному рисунку 2.12 показано файл “package.json” в якому видно перелік усіх пакетів потрібних в нашому проєкті.

{} package.json > ...

```
{
  "name": "diploma",
  "version": "0.1.0",
  "type": "module",
  ▷Debug
  "scripts": {
    "dev": "vite",
    "preview": "vite preview"
  },
  "dependencies": {
    "@emotion/react": "^11.14.0",
    "@emotion/styled": "^11.14.0",
    "@mui/icons-material": "^6",
    "@mui/material": "^6.4.9",
    "@mui/x-charts": "^8.0.0-beta.3",
    "@mui/x-data-grid": "^7.28.3",
    "@toolpad/core": "^0.12.1",
    "firebase": "^11",
    "moment": "^2.30.1",
    "react": "^19.0.0",
    "react-dom": "^19.0.0",
    "react-router": "^7"
  },
  "devDependencies": {
    "@types/react": "^19.0.0",
    "@types/react-dom": "^19.0.0",
    "@vitejs/plugin-react": "^4.3.2",
    "typescript": "^5",
    "vite": "^5.4.8"
  }
}
```

Рисунок 2.12 – “package.json” зі списком пакетів проєкту

Далі ми маємо створити додаток на сайті Firebase, щоб там налаштувати авторизацію та створити нашу БД. Це можна зробити і локально, але ми скористаємось графічним інтерфейсом - для зручності (рис. 2.13).

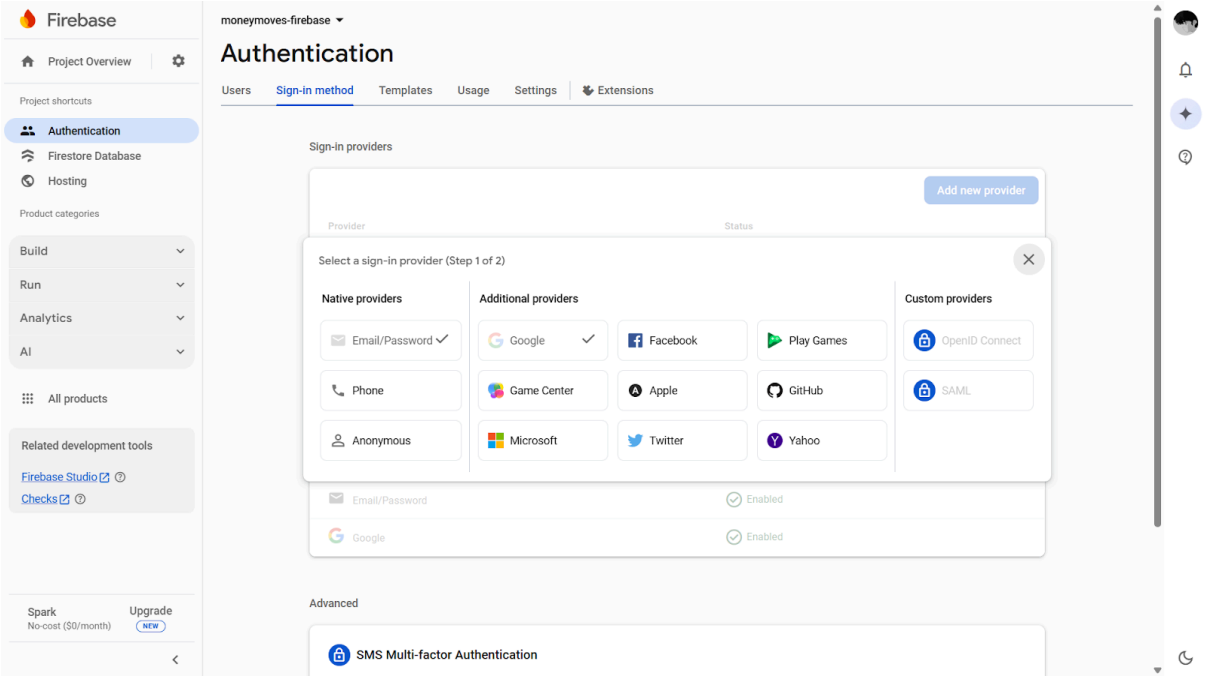

Рисунок 2.13 – Перелік доступних провайдерів аутентифікації для Firebase Authentication

Наступним кроком потрібно внести змінні середовища, а саме налаштування та API-ключ для синхронізації з Firebase. Ці дані можна отримати при створенні додатку Firebase, або пізніше в налаштуваннях проєкту.

Вебдодаток побудований таким чином, що його функціональність охоплює наступні ключові сценарії взаємодії користувача з системою:

- авторизація;
- робота з рухами коштів;
- робота з категоріями;
- робота з рахунками;
- робота з проєктами витрат;

- формування аналітики.

Для впровадження цієї структури за допомогою React.js було застосовано відповідну архітектуру проєкту (рис. 2.14).

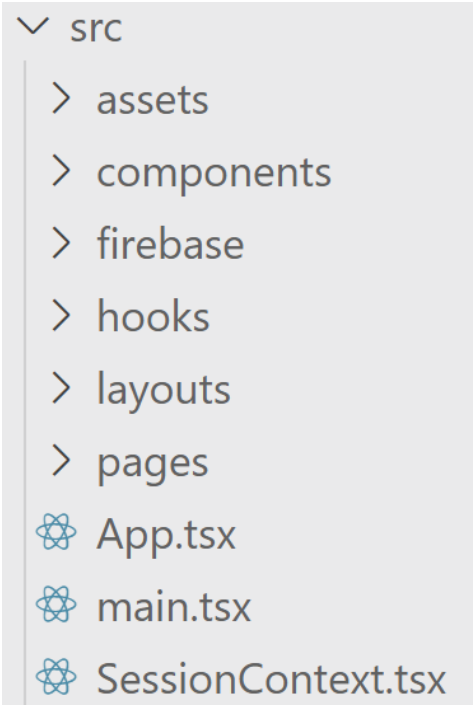

Рисунок 2.14 – Структура кореневої директорії проєкту

Окрім загальної структури усі модулі було категоризовано відповідно області застосування:

- рухи коштів;
- категорії;
- рахунки;
- проєкти;
- аналітика.

Приклад такого групування зображено на рисунку 2.15.

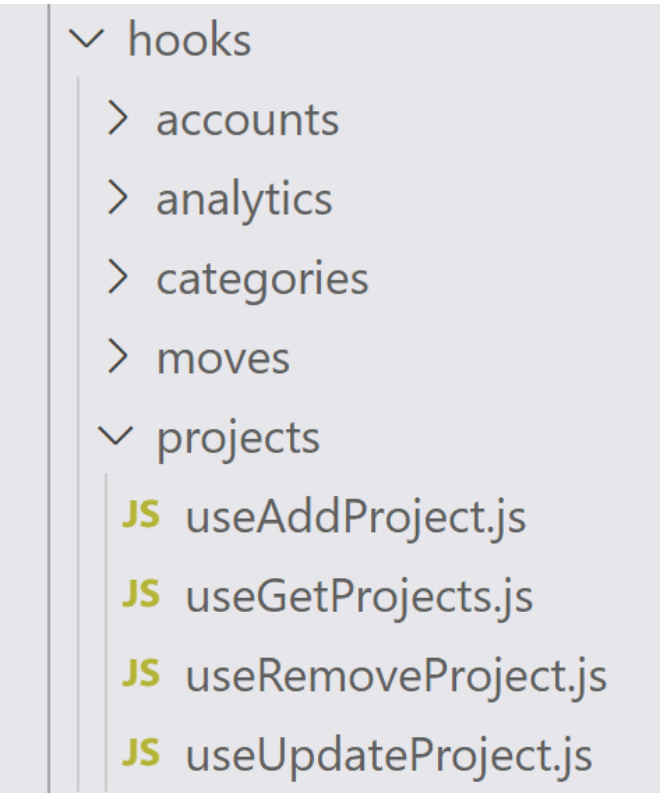

Рисунок 2.15 – Структура папки “hooks”

Всього сайт має 6 енд-поінтів:

- “/sign-in” – сторінка реєстрації;
- “/moneymoves” – сторінка зі списком рухів;
- “/categories” – сторінка зі списком категорій доходів та витрат;
- “/projects” – сторінка проєктами;
- “/accounts” – сторінка рахунків;
- “/” – сторінка дашборду.

На рисунку 2.16 зображено назви доступних сторінок та React-компоненти, що їм відповідають.

```
const router = createBrowserRouter([
  {
    Component: App,
    children: [
      {
        path: "/",
        Component: Layout,
        children: [
          { path: "",
            Component: DashboardPage,
          },
          { path: "moneymoves",
            Component: Moneymoves,
          },
          { path: "categories",
            Component: Categories,
          },
          { path: "accounts",
            Component: Accounts,
          },
          { path: "projects",
            Component: Projects,
          },
        ],
      },
      { path: "/sign-in",
        Component: SignInPage,
      },
    ],
  },
]);
```

Рисунок 2.16 – Частина файлу “main.tsx” в якій описані маршрути вебдодатку

Усі сторінки захищені від несанкціонованого доступу редиректом на сторінку авторизації у разі якщо користувач не авторизований. Також доступ до даних обмежено правилами Firestore налаштованими у Firebase(рис 2.17).

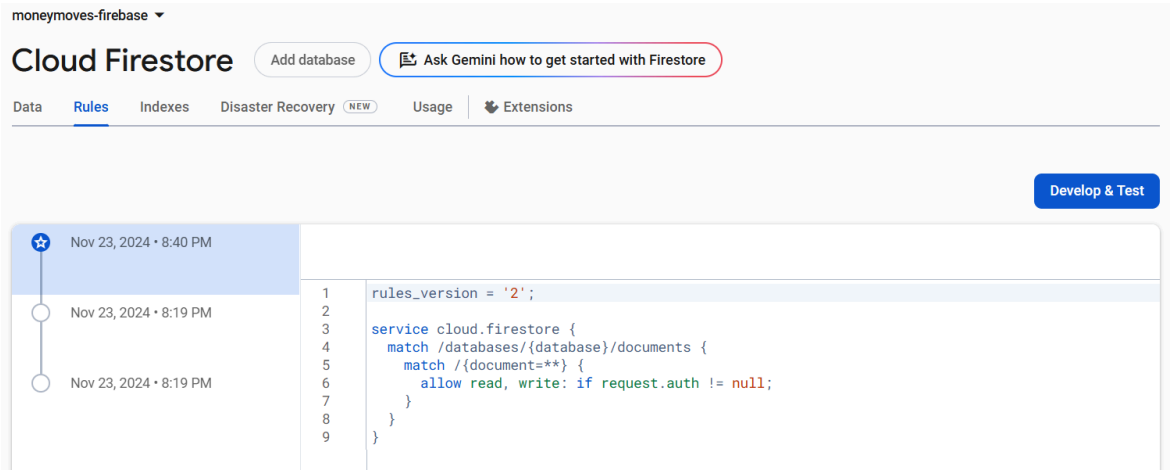

Рисунок 2.17 – Правила доступу у Firestore

Опишемо сторінку створення рухів коштів. Аналогічно їй працюють сторінки: категорій, рахунків, проєктів. Зверху користувач має форму для заповнення полів нового запису, а знизу список внесених даних.

Для списку було використано компонент з бібліотеки MUI – DataGrid. Це потужний компонент для створення інтерактивних таблиць. Він дозволяє зручно відображати великі обсяги табличних даних з підтримкою широкого набору функцій, необхідних для роботи з інформацією (рис. 2.18).

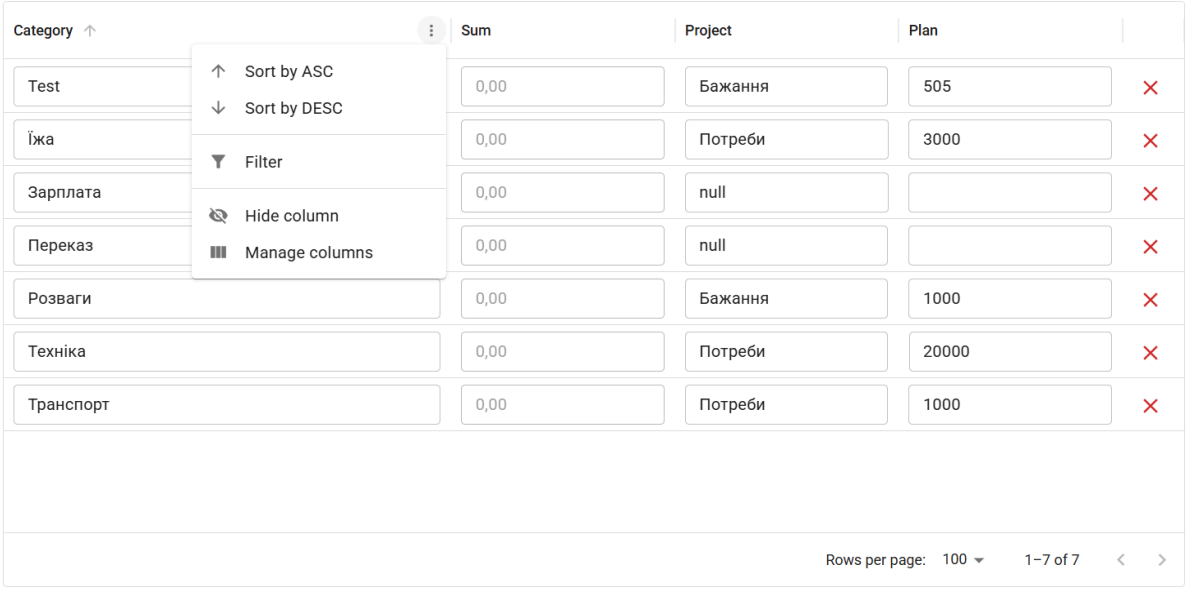

Рисунок 2.18 – Фільтри та сортування доступні у DataGrid

2.6. Організація тестування та налагодження додатку

Розроблений вебдодаток призначений для обліку особистих фінансів, надаючи користувачеві зручний інструмент для ведення записів про доходи й витрати, створення власних категорій, рахунків та проєктів, а також для перегляду зведеної статистики та аналітики.

Клієнтська частина реалізована за допомогою бібліотеки React.js, що забезпечує швидкий відгук інтерфейсу з компонентною архітектурою. Сторінки додатку не перезавантажуються при навігації, що забезпечує плавний досвід користувача.

Застосунок було протестовано на коректність виконання основного функціоналу, а саме:

- реєстрація нового користувача та авторизація через email/пароль або обліковий запис Google;
- додавання, редагування та видалення фінансових записів (доходи та витрати);
- створення власних категорій операцій та використання їх при додаванні нових записів;
- робота з рахунками та проєктами витрат;
- формування звітів та перегляд статистики за обраний період;
- захищені маршрути: забезпечення доступу до функціональності тільки після авторизації.

Тестування програмного засобу здійснюється для виявлення помилок в роботі. Тестування проводилось на ПК з процесором Intel Core i5-7300U та з ОЗП 16 ГБ.

Додаток стабільно працює в браузерах Chrome 136.0.7103.93, Edge 136.0.3240.64.

2.7. Рекомендації щодо використання та впровадження програмного засобу

Розроблений вебдодаток призначений для обліку особистих фінансів, зокрема для фіксації доходів і витрат, створення власних категорій, рахунків і проєктів витрат, а також формування статистичних звітів на основі введених даних. Застосунок може бути рекомендований до використання окремими користувачами для організації фінансового обліку та аналізу витрат.

Для коректної роботи додатку необхідно дотримуватись таких умов:

- наявність стабільного доступу до мережі Інтернет;
- наявність сучасного веббраузера з підтримкою JavaScript;
- створення облікового запису або авторизація за допомогою облікового запису Google;
- уважне введення даних у відповідні поля (сума, дата, категорія тощо), що забезпечить коректність відображення аналітики.

Для локального розгортання або доопрацювання застосунку з боку розробника необхідно на машині мати встановлений Node.js (версія 18 або вище) – для запуску середовища розробки та аккаунт Google – для створення додатку в Firebase.

ВИСНОВКИ

Під час виконання роботи було проаналізовано існуючі рішення для ведення особистого або бізнес-бюджету, а також розглянуто сучасні технології, що можуть бути використані для створення таких систем. Особливу увагу приділено бібліотеці React.js як інструменту для побудови інтерактивного інтерфейсу користувача, а також Material UI – набору готових компонентів, що дозволяє швидко створювати сучасний і адаптивний дизайн. Для реалізації зберігання даних та автентифікації обрано Google Firebase, зокрема сервіси Authentication і Firestore.

У результаті розробки створено повнофункціональний вебдодаток для обліку доходів і витрат. Користувачі можуть реєструватися та входити в систему, додавати записи про грошові операції, створювати та редагувати категорії, вести облік за рахунками та проєктами, а також переглядати аналітичні звіти за різними параметрами. Усі дані зберігаються в хмарній NoSQL-базі Firestore, що забезпечує високу швидкість доступу та масштабованість.

Функціонал додатку включає:

- додавання, редагування та видалення записів про доходи й витрати;
- створення та управління категоріями;
- ведення обліку по рахунках та проєктах витрат;
- побудову статистики за періодами, категоріями, рахунками;
- захищену авторизацію користувачів.

Розроблений додаток є ефективним інструментом для управління фінансами як для індивідуального використання, так і в межах невеликих команд або бізнесів. Його архітектура та використані технології дозволяють легко масштабувати функціонал і адаптувати продукт під нові вимоги користувачів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. True Tamplin. The benefits of expense tracking and how you can do it effectively. *Forbes*. URL: https://www.forbes.com/sites/truetamplin/2025/01/15/the-benefits-of-expense-tracking-and-how-you-can-do-it-effectively/ (дата звернення: 16.05.2025).
2. Зарплати в Україні та контроль витрат – що треба знати про фінансову грамотність - reNews. *reNews*. URL: https://renews.com.ua/ekonomika/zarplati-v-ykrayini-ta-kontrol-vitrat-sho-treba-znati-pro-finansovy-gramotnist/ (дата звернення: 16.05.2025).
3. The Business Research Company. How will the budgeting assistance market grow? key trends and opportunities for 2025 and beyond - latest global market insights. Latest Global Market Insights. URL: https://blog.tbrc.info/2025/03/budgeting-assistance-market-share/ (дата звернення: 16.05.2025).
4. *BizMag.* URL: https://bizmag.com.ua/finansova-gramotnist/ (дата звернення: 16.05.2025).
5. TheRoom. Важливість доступності та інклюзивного дизайну. *UXPUB*. URL: https://ux.pub/theroom_db/vazhlivist-dostupnosti-ta-inkliuzivnogho-dizainu-3dg3 (дата звернення: 16.05.2025).
6. Carnegie M., Carroll L. Why americans' 'YOLO' spending spree baffles economists. *BBC*. URL: https://www.bbc.com/worklife/article/20231130-why-americans-yolo-spending-attitude-baffles-economists (дата звернення: 16.05.2025).
7. Планування = стабільність: як створити місячний особистий бюджет. покрокова інструкція. *Work.ua*. URL: https://www.work.ua/articles/career/3113/ (дата звернення: 16.05.2025).

8. Nissen K. One-third of Americans use three or more financial apps. *S&P Global*. URL: https://www.spglobal.com/market-intelligence/en/news-insights/research/one-third-of-americans-use-three-or-more-financial-apps (дата звернення: 16.05.2025).
9. FoxmindEd. Чим відрізняється сайт від веб додатку: технічні аспекти. *FoxmindEd*. URL: https://foxminded.ua/chym-vidrizniaietsia-sait-vid-veb-dodatku/ (дата звернення: 16.05.2025).
10. React. React. *React*. URL: https://react.dev/ (дата звернення: 16.05.2025).
11. React. Component. *React*. URL: https://react.dev/reference/react/Component (дата звернення: 16.05.2025).
12. React. React DOM APIs. *React*. URL: https://react.dev/reference/react-dom (дата звернення: 16.05.2025).
13. Material UI. React components that implement Material Design. *MUI*. URL: https://mui.com/material-ui/ (дата звернення: 16.05.2025).
14. Material UI. Overview. *MUI*. URL: https://mui.com/material-ui/getting-started/ (дата звернення: 16.05.2025).
15. Material UI. Full stack components for React. *MUI*. URL: https://mui.com/toolpad/ (дата звернення: 16.05.2025).
16. Firebase. Make your app the best it can be with Firebase and generative AI. *Firebase*. URL: https://firebase.google.com/ (дата звернення: 16.05.2025).
17. Firebase. Privacy and security in Firebase. *Firebase*. URL: https://firebase.google.com/support/privacy (дата звернення: 16.05.2025).
18. Firebase. Cloud Firestore. *Firebase*. URL: https://firebase.google.com/docs/firestore (дата звернення: 16.05.2025).
19. Credit Karma. Free credit scores and tools to make financial progress. *Intuit Credit Karma*. URL: https://www.creditkarma.com/ (дата звернення: 16.05.2025).

20. CardRates.com. Average Credit Card Debt by Country in 2025. *CardRates.com*. URL: https://www.cardrates.com/advice/average-credit-card-debt-by-country/ (дата звернення: 16.05.2025).
21. Cashew. Finally, get financially fit. *Cashew*. URL: https://cashewapp.web.app/ (дата звернення: 16.05.2025).
22. Google Play. Cashew. *Google Play.* URL: https://play.google.com/store/apps/details?id=com.budget.tracker_app/ (дата звернення: 19.05.2025).
23. QALight. Спіральна модель (spiral model). *QALight*. URL: https://qalight.ua/baza-znaniy/cpiralna-model-spiral-model/ (дата звернення: 16.05.2025).
24. Федченков Є. Що таке канбан і чим він корисний? - Блог системи управління проектами Worksection. *Worksection*. URL: https://worksection.com/ua/blog/kanban.html (дата звернення: 16.05.2025).
25. React. Built-in React Hooks. *React*. URL: https://react.dev/reference/react/hooks (дата звернення: 16.05.2025).
26. Firebase. Firebase Authentication. *Firebase*. URL: https://firebase.google.com/docs/auth (дата звернення: 16.05.2025).

ДОДАТКИ

Додаток А

ТЕХНІЧНЕ ЗАВДАННЯ НА РОЗРОБКУ ВЕБДОДАТКУ ДЛЯ ВЕДЕННЯ ДОХОДІВ ТА ВИТРАТ

ВСТУП

ПІДСТАВИ ДЛЯ РОЗРОБКИ

Розробка проводиться на основі індивідуального завдання, поставленого керівником бакалаврської роботи для спеціальності 122 “Комп’ютерні науки”

ПРИЗНАЧЕННЯ РОЗРОБКИ

Метою створення програмного продукту є реалізація сучасного вебдодатку для персонального фінансового обліку, що дозволяє користувачеві зручно фіксувати доходи та витрати, аналізувати фінансову інформацію за допомогою динамічних звітів, а також вести облік за проєктами та рахунками. Додаток демонструє практичне застосування сучасних вебтехнологій, таких як React, Firebase та бібліотека інтерфейсів MUI Toolpad Core.

ВИМОГИ ДО ПРОГРАМНОГО ПРОДУКТУ

Вимоги до функціональних характеристик: вебдодаток має бути інтуїтивно зрозумілим для користувача та виконувати такі функції:

- реєстрація та авторизація користувачів за допомогою Firebase Authentication;
- додавання доходів і витрат із зазначенням суми, категорії, дати, рахунку, проєкту та опису;
- управління категоріями: створення, редагування, видалення;
- управління рахунками та проєктами: створення, редагування, видалення;

- відображення історії фінансових операцій у вигляді таблиці з можливістю фільтрації;
- зручне перемикання між функціональними розділами за допомогою маршрутизації;
- підтримка адаптивної верстки для коректного відображення на пристроях з різною роздільною здатністю екрана.
- генерація аналітичних звітів в розрізі обраних параметрів.

Вимоги до надійності: Застосунок має стабільно функціонувати в середовищі сучасних веббраузерів (Google Chrome, Mozilla Firefox, Microsoft Edge, Safari) і забезпечувати коректну обробку даних користувача в онлайн-режимі. Усі операції зберігаються в хмарній NoSQL базі даних Firestore з урахуванням правил безпеки доступу.

Умови експлуатації:

- наявність стабільного інтернет-з’єднання;
- пристрій із встановленим сучасним браузером із підтримкою стандартів ECMAScript 6+, HTML5 і CSS3;
- реєстрація користувача для доступу до персоналізованих даних.

ВИМОГИ ДО ПРОГРАМНОЇ ДОКУМЕНТАЦІЇ

До складу документів входять

- технічне завдання;
- звіт з кваліфікаційної роботи;
- презентація;
- відеодемонстрація користувацького інтерфейсу.

СТАДІЇ І ЕТАПИ РОЗРОБКИ

1. Аналіз предметної області – вивчення наявних рішень для обліку доходів і витрат, виявлення вимог до майбутнього продукту.

2. Проєктування інтерфейсу – створення структури інтерфейсу користувача з урахуванням адаптивності.
3. Вибір інструментів і технологій – визначення стеку розробки (React, Firebase, MUI, Toolpad Core).
4. Реалізація основного функціоналу – створення компонентів, налаштування маршрутизації, інтеграція з базою даних.
5. Тестування – перевірка роботи функцій, коректність обробки даних, виявлення та виправлення помилок.
6. Оформлення проєкту та підготовка демонстраційних матеріалів.

ПОРЯДОК КОНТРОЛЮ І ПРИЙМАННЯ

Приймання результатів роботи здійснюється в процесі здачі кваліфікаційної роботи.

ДОДАТОК Б
ІНСТРУКЦІЯ КОРИСТУВАЧУ ВЕБДОДАТКУ ДЛЯ ВЕДЕННЯ ДОХОДІВ І ВИТРАТ

1. Загальні відомості

Вебдодаток для ведення персонального фінансового обліку – це сучасний інструмент ведення грошових потоків, що дозволяє користувачу керувати особистими коштами, створювати аналітичні звіти, вести облік за різними проєктами та рахунками, а також здійснювати доступ до даних з будь-якого пристрою.

2. Функціональне призначення

Застосунок розроблено з метою автоматизації процесу обліку доходів і витрат, систематизації фінансових записів, контролю бюджету та формування звітності.

3. Умови застосування програми

Додаток підтримується в сучасних браузерах: Google Chrome, Mozilla Firefox, Microsoft Edge, Safari (актуальні версії). Працює на десктопних і мобільних пристроях завдяки адаптивному дизайну. Для повноцінного використання потрібне стабільне інтернет-з’єднання.

4. Опис роботи програми

Після створення облікового запису за допомогою електронної пошти та пароля, користувач потрапляє у персоналізоване середовище для ведення власного фінансового обліку. Основна функціональність програми охоплює кілька ключових розділів.

По-перше, користувач може додавати фінансові операції – як доходи, так і витрати – із зазначенням суми, категорії, дати, опису, а також вибором рахунку і проєкту. Усі додані записи можна згодом редагувати або видаляти.

По-друге, реалізовано зручну систему управління категоріями, рахунками та проєктами. Користувач має можливість створювати, редагувати та видаляти власні елементи для точнішої організації фінансової інформації.

Вебдодаток також дозволяє переглядати всю історію транзакцій у вигляді зручної таблиці з підтримкою фільтрації та сортування даних.

Окрему увагу приділено модулю аналітики. Користувач може формувати інтерактивні звіти з графіками та діаграмами, використовуючи різні параметри: обраний період часу, категорії, рахунки, проєкти, а також тип операцій (дохід або витрата).

Усі дані автоматично зберігаються у хмарній базі даних Firestore, що забезпечує надійне зберігання інформації та її синхронізацію між усіма пристроями користувача в режимі реального часу.

АНОТАЦІЯ

Міщук В. М. – Проєктування та розробка сервісу для ведення персональних доходів і витрат – Рукопис.

Кваліфікаційна робота за спеціальністю 122 Комп'ютерні науки. – Волинський національний університет імені Лесі Українки, Луцьк. – 2025 р.

Робота присвячена розробці вебзастосунку для ведення особистих витрат, що забезпечує зручне додавання, зберігання та аналіз фінансових операцій. У процесі реалізації досліджено сучасні підходи до створення вебінтерфейсів, обґрунтовано вибір технологій та розроблено повнофункціональний додаток для користувачів.

В основі клієнтської частини використано бібліотеку React, що забезпечує компонентну структуру та динамічну взаємодію з користувачем. Для створення інтерфейсу застосовано Material UI. Зберігання даних та автентифікацію реалізовано за допомогою сервісів Firebase Authentication та Cloud Firestore.

При виконанні кваліфікаційної роботи розроблено масштабований вебдодаток для ведення фінансів із можливістю створення категорій, рахунків і проєктів, формування звітів та аналітики. Реалізовано підтримку аутентифікації, захищених маршрутів, багаторівневого структурування фінансових записів, а також логіку планування бюджету з використанням концепції «конвертів».

У роботі подано архітектурні рішення, результати тестування функціоналу, а також рекомендації щодо впровадження додатку.

Ключові слова: вебдодаток, управління фінансами, облік витрат, облік доходів, Firebase, Firestore, React, Material UI, Toolpad Core, JavaScript.