РЕФЕРАТ

Пояснювальна записка: ___ с., ___ рис., ___ табл., __ дод., ___ джерел.

Об'єкт розробки: мобільний додаток для ведення домашнього господарства, основою функцією якого є скорочення часу на ведення домашньої бухгалтерії, автоматизації процесу організації ведення обліку поточних коштів, отримання візуальних характеристик руху коштів і підбиття підсумків балансу сімейного бюджету.

Мета кваліфікаційної роботи: проєктування та розробка мобільного додатку для ведення фінансового обліку «BrainWallet» під ОС Android, що забезпечує автоматизацію процесу ведення бюджету господарства за допомогою обліку та аналізу фінансових ресурсів сім'ї за певний період часу.

У вступі розглядається аналіз та сучасний стан проблеми, конкретизується мета кваліфікаційної роботи та галузь її застосування, наведено обґрунтування актуальності теми та уточнюється постановка завдання.

У першому розділі проведено аналіз предметної галузі, визначено актуальність завдання та призначення розробки, розроблена постановка завдання, задані вимоги до програмної реалізації, технологій та програмних засобів.

У другому розділі виконано аналіз існуючих рішень, обрано вибір платформи для розробки, виконано проектування і розробка програми, наведено опис алгоритму і структури функціонування програми, визначені вхідні і вихідні дані, наведені характеристики складу параметрів технічних засобів, описаний виклик та завантаження програми, описана робота програми.

В економічному розділі визначено трудомісткість розробленої інформаційної системи, проведений підрахунок вартості роботи по створенню програми та розраховано час на його створення.

Практичне значення роботи полягає в тому, що розроблений додаток може бути використаний користувачами, зацікавленими в скороченні часу для ведення домашньої бухгалтерії і отриманні візуальних характеристик потоків фінансових ресурсів з метою найбільш правильного їх використання і ефективного розподілу.

Актуальність поставленого завдання обумовлюється високим попитом на такого роду розробки, в зв'язку з необхідністю підвищення ефективності ведення обліку фінансових ресурсів господарства в домашніх умовах за допомогою простих і зрозумілих будь-якому користувачеві ресурсів.

Список ключових слів: БЮДЖЕТ, ВИТРАТИ, ДОХОДИ, МОБІЛЬНИЙ ДОДАТОК, ПРОГРАМУВАННЯ, ІНФОРМАЦІЙНА СИСТЕМА.

ABSTRACT

Explanatory note: ___ pp., ___ fig., ___ table, __ appendix, ___ sources.

Object of development: a mobile application for housekeeping, the main function of which is to reduce the time for household accounting, automate the process of organizing the accounting of current funds, obtaining visual characteristics of cash flows and summarizing the balance of the family budget.

The purpose of the qualification work: design and development of a mobile application for financial accounting "BrainWallet" for Android, which provides automation of the process of managing the budget by accounting and analysis of financial resources of the family for a certain period of time.

The introduction considers the analysis and the current state of the problem, specifies the purpose of the qualification work and the field of its application, provides a justification for the relevance of the topic and clarifies the problem.

In the first section the analysis of the subject area is carried out, the urgency of the task and purpose of development is defined, the statement of the task is developed, requirements to software realization, technologies and software are set.

The second section analyzes the existing solutions, selects the platform for development, performs design and development of the program, describes the algorithm and structure of the program, determines the input and output data, provides characteristics of the parameters of hardware, describes the call and download program, describes the program .

The economic section determines the complexity of the developed information system, calculates the cost of work to create a program and calculates the time for its creation.

The practical significance of the work is that the developed application can be used by users interested in reducing the time for home accounting and obtaining visual characteristics of the flows of financial resources for the most correct use and efficient distribution.

The urgency of the task is due to the high demand for this type of development, due to the need to improve the efficiency of accounting for financial resources of the economy at home with simple and understandable to any user resources.

Keywords: BUDGET, COSTS, REVENUES, MOBILE APPLICATION, PROGRAMMING, INFORMATION SYSTEM.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ІС – інформаційна система;

БД – база даних;

ОС – операційна система;

ПЗ – програмне забезпечення;

ПК – персональний комп'ютер;

СКБД – система керування базою даних.

ADT – Android Development Tools, набір інструментів для розробки під Android.

ЗМІСТ

ВСТУП

Ведення персонального чи сімейного фінансового обліку – дуже важливий аспект в сучасному світі. Ведучи оцінку фінансів і підрахунок своїх витрат, можна дізнатися яку суму грошей,і на що конкретно ви витрачаєте. Деяким речам ви будете здивовані, а на деякі відкриються очі. Так влаштований наш розум, в нашій підсвідомості запам'ятовуються істотні витрати – виплати кредитів, закупівля продуктів або великих речей в магазинах, покупка бензину, подарунки. Таким чином, часто здається, що витрачаємо ми менше. Але, якщо включити повну усвідомленість і проаналізувати свої витрати, виявиться, що більша частина нашого бюджету складають саме дрібні непередбачені витрати. Наприклад, незначні витрати на шкідливу їжу, кафе, непотрібні покупки, всі вони зберігаються непоміченими. Вам відкриються очі на вартість своїх звичок. Кава в затишному кафе, посиденьки в барах, регулярне використання таксі – якщо це порахувати, то може виявитися, що обходиться це недешево. Аналіз допоможе усвідомити, що цінність цих радощів, значно нижче за ціну на них. Тільки в разі зменшення частини непотрібних витрат, можна навчитися накопичувати і створювати вклади.

Для ведення свого бюджету можна використовувати блокнот та ручку, але у час, коли людина більшу частину свого дня проводить в русі, є необхідність впровадити цей процес у її життя. І в цьому випадку незамінним інструментом стає телефон або планшет. Сучасний світ дуже складно уявити без техніки, яка завжди поруч з нами, в цьому випадку, вона може послужити не лише, як засіб зв'язку або дозвілля, а як особистий фінансовий помічник. Перевагами в такому методі обліку стануть: зручність, компактність, надійність. Адже ваш помічник завжди разом з вами: він може нагадувати про витрати, ним можна користуватися у транспорті, на перерві й навіть під час роботи, адже це швидко і легко.

Метою кваліфікаційної роботи є проєктування та розробка мобільного додатку для ведення фінансового обліку «BrainWallet» під ОС Android, що

забезпечує автоматизацію процесу ведення бюджету господарства за допомогою обліку та аналізу фінансових ресурсів сім'ї за певний період часу.

Додаток дозволяє скоротити час для ведення домашньої бухгалтерії, автоматизувати процес організації ведення обліку поточних коштів, отримувати візуальні характеристики руху ресурсів і підведення підсумків балансу сімейного бюджету.

Актуальність поставленого завдання обумовлюється високим попитом на такого роду розробки, в зв'язку з необхідністю підвищення ефективності ведення обліку фінансових ресурсів господарства в домашніх умовах за допомогою простих і зрозумілих будь-якому користувачеві ресурсів.

Розроблена система може бути використана користувачами, зацікавленими в скороченні часу для ведення домашньої бухгалтерії і отриманні візуальних характеристик потоків фінансових ресурсів з метою найбільш правильного їх використання і ефективного розподілу.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1. Загальні відомості з предметної галузі

Домашні господарства виступають одним з важливих суб'єктів економічної діяльності, від результатів якої залежить не тільки добробут окремої господарської одиниці, а й усього населення країни в цілому. Ставши найбільшим суб'єктом економіки поряд з комерційними підприємствами і державою, домашні господарства беруть участь у всіх макрорегулюючих процесах. Глибока економічна криза і як наслідок цього - тяжке матеріальне становище більшої частини населення призводять до активної діяльності домашнього господарства. В економічній теорії під домашнім господарством розуміється господарство, яке ведеться одним або кількома особами, які проживають разом і мають загальний бюджет.

Фінанси домашнього господарства (домогосподарства), як і фінанси суспільства в цілому, являють собою економічні грошові відносини щодо формування та використання фондів грошових коштів з метою забезпечення матеріальних і соціальних умов життя членів господарства та їх відтворення. Будучи ланкою у фінансовій системі на рівні окремої сім'ї, вони виступають первинним елементом соціально-економічної структури суспільства. На відміну від фінансів комерційних підприємств і організацій, що мають вирішальне значення в створенні, первинному розподілі і використанні вартості валового внутрішнього продукту і національного доходу, фінанси домогосподарства не стали пріоритетним ланкою фінансової системи і відіграють підлеглу, хоча і важливу роль в загальній сукупності фінансових відносин. Виділення фінансів домашнього господарства в самостійну ланку фінансової системи в умовах розвинених ринкових відносин, коли кругообіг капіталу охоплює переважну частину процесу виробництва, обумовлене рядом факторів. Функції фінансів домогосподарств. Сутність фінансів домогосподарств знаходить свій вияв у

функціях.

Нині вони виконують дві базові функції:

– забезпечення життєвих потреб сім'ї;

– розподільчу функцію.

Початкова і головна - функція забезпечення життєвих потреб сім'ї. Вона створює реальні умови існування членів даної сім'ї.

Розвиток ринкових відносин суттєво вплинуло на форму прояви цієї функції. У період натурального господарства продукція, створювана членами, задовольняла їх потреби, і обмін надлишками виникав рідко, в невеликій кількості і, як правило, по сусідству. В результаті товарно-грошових відносин, появи, а за тим і збільшення ринку сталося:

– розширення матеріальних, соціальних, культурних та інших потреб сім'ї;

– створення та зростання грошових коштів домашнього господарства;

– виникнення грошового фонду - сімейного бюджету, призначеного для забезпечення матеріальними благами.

Розподільна функція фінансів домогосподарств охоплює первинне розподіл національного доходу і формування первинних доходів сім'ї. Фінансові відносини домашнього господарства включають дві групи:

– відносини між даної господарської одиницею і іншими ланками фінансової системи (державними фінансами - бюджетами і позабюджетними фондами, і фінансами комерційних організацій і підприємств), створюючи первинні доходи у вигляді заробітної плати, пенсій, допомог і т.п .;

– відносини між членами домогосподарства, коли кошти розподіляються і відокремлюються, утворюючи відокремлені грошові фонди.

Фінансові ресурси домогосподарства - це сукупний фонд грошових коштів, що знаходиться в розпорядженні сім'ї. Створений в результаті виробничої діяльності членів домогосподарства, він виступає частиною національного доходу суспільства. Обсяг грошового фонду домогосподарства залежить від зусиль кожного в господарстві.

1.2. Призначення розробки та галузь застосування

Фінансові ресурси домогосподарства - це сукупний фонд грошових коштів, що знаходиться в розпорядженні сім'ї. Створений в результаті виробничої діяльності членів домогосподарства, він виступає частиною національного доходу суспільства. Обсяг грошового фонду домогосподарства залежить від зусиль кожного в господарстві.

Фінансові ресурси домогосподарства виступають у вигляді відособлених грошових фондів, що мають, як правило, цільове призначення. Створюються два основних фонду:

– фонд споживання, призначений для задоволення особистих потреб даного колективу - родини (придбання продуктів харчування, товарів промислового виробництва, оплата різних платних послуг та ін.);

– фонд заощаджень (відкладених потреб), який буде використаний в майбутньому для придбання дорогих товарів або як капітал для отримання прибутку.

Склад фінансових ресурсів домогосподарств включає:

– власні кошти, тобто зароблені кожним членом сім'ї - зарплата, дохід від підсобного господарства, прибуток від комерційної діяльності;

– кошти, мобілізовані на ринку, у формі отриманого кредиту у кредитних організацій, дивіденди, відсотки;

– кошти, що надійшли в порядку перерозподілу пенсії, допомоги, позики з бюджетів та позабюджетних соціальних фондів.

Фінансові ресурси формують бюджет домашнього господарства. За своїм матеріальним змістом бюджет домогосподарства - це форма освіти і використання фонду грошових коштів домогосподарства. Він об'єднує сукупні доходи членів домогосподарства та витрати, що забезпечують їх особисті потреби. Коштів сімейного бюджету постійно не вистачає в зв'язку з розширенням потреб.

Розрізняють постійний і тимчасовий дохід домогосподарства.

Дохід сімейного бюджету визначає обсяг споживання домогосподарств. Постійний дохід, розмір якого повторюється з року в рік, не викличе серйозних коливань в їх споживчих витратах ..

Розвиток комп'ютерних інформаційних систем і телекомунікаційних технологій привели до впровадження сучасних інноваційних технологій в усі сфери життєдіяльності людини, в тому числі і до використання даних ресурсів для полегшення управління фінансовими коштами родини в домашніх умовах за допомогою ПК.

Розроблений мобільний додаток призначений для створення інформаційної системи ведення домашнього аудиту, що забезпечує автоматизацію процесу ведення бюджету господарства за допомогою обліку та аналізу фінансових ресурсів сім'ї.

Додаток має дати можливість скоротити час для ведення домашньої бухгалтерії, автоматизувати процес організації ведення обліку поточних коштів, отримувати візуальні характеристики руху засобів та підведення підсумків балансу сімейного бюджету.

1.3. Підстава для розробки

Підставою для розробки кваліфікаційної роботи бакалавра на тему «Розробка мобільного додатку для ведення аудиту фінансових витрат сімейного бюджету під ОС Android» є наказ по Національному технічному університету «Дніпровська політехніка» від __.__. 2021р. № ____-_.

1.4. Постановка завдання

Завдання даної кваліфікаційної роботи є спроєктувати та розробити мобільний додаток для ведення аудиту фінансових витрат сімейного бюджету під ОС Android.

Дана програма повинна допомогти користувачам раціонально розподіляти і контролювати їх бюджет.

Розроблений додаток повинен забезпечувати вирішення наступних завдань:

– ведення бази даних домашнього аудиту (введення, додавання, видалення, редагування даних про витрати, доходи);

– виконання розрахунків показників сум за різні періоди часу;

– візуалізація стану бюджету;

– розрахунок балансу бюджету сім'ї;

– виконання аналізу даних витрат і надходжень;

– фільтрування і угруповання інформації за обраними показниками.

Для вирішення цих завдань при розробці програми необхідно здійснити наступні дії:

1. Розробка логічної моделі програми.

2. Створення бази даних ІС.

3. Розробка простого і зрозумілого інтерфейсу додатку.

4. Забезпечення доступу до даних БД.

5. Можливість редагування, додавання, видалення, збереження інформації з бази даних в додатку в режимі реального часу.

6. Створення та виконання запитів з БД і виведення їх результатів.

7. Передбачити висновок деякою інформацією у вигляді діаграм.

8. Забезпечення можливості налаштування системи за бажанням користувача.

1.5. Вимоги до програми або програмного виробу

1.5.1. Вимоги до функціональних характеристик

Мобільний додаток призначений для спрощення ведення фінансового обліку. Програма повинна мати великий функціонал для детального ведення

бюджету, поглиблення в фінансову грамотність і поліпшення матеріального становища.

Для того, щоб почати роботу з програмою користувачу необхідно створити власний аккаунт або авторизуватися. Для реєстрації необхідно вказати адресу пошти, обрати власний логін та пароль для подальшого входу.

Після авторизації робота з програмою повинна розпочинатися із заповнення відомостей по доходам, а саме користувачу необхідно ввести перший дохід. Після цього йому необхідно обрати необхідні витрати чи ввести ще раз доходи за категоріями, які у програмі наявні за замовчуванням, але з можливістю додавати власні. Для створення власної категорії, користувачу потрібно буде ввести назву категорії. Така ж можливість повинна бути передбачена і для витрат.

Для того, щоб додавати витрати необхідно вказати суму та категорію.

Окрім базових функцій, необхідно передбачити і додаткові:

– діаграми: можна переглянути і проаналізувати свої витрати, за день, місяць, рік;

– доступ з інших пристроїв: щоб мати можливість стежити за своїм бюджетом з будь-якого пристрою (процес синхронізації).

Для виконання всього переліку вимог до системи і забезпечення повної функціональності програми, програма повинна виконувати наступні функції:

1. Переглядати, вводити, видаляти, редагувати дані доходи.

2. Додавати нові види доходів.

3. Переглядати, вводити, видаляти, редагувати дані про витрати.

4. Додавати нові види витрат.

5. Проводити аналіз фінансових надходжень і розходів.

6. Виконувати фільтрацію даних з різних видів витрат, доходів.

7. Групувати дані по нарахуваннях або розтрат по днях, тижнях, місяцях, роках.

8. Підводити підсумки по балансу.

9. Виводити відфільтровану інформацію у вигляді діаграм.

1.5.2. Вимоги до інформаційної безпеки

До розроблювального додатка сформовані наступні вимоги забезпечення її безпеки та цілісності:

— при відмові системи, данні, які були введені в систему, не повинні бути пошкоджені;

— як система, так і оброблені дані, повинні легко переноситися;

— час відновлення після збою 5 хвилин.

Надійність роботи розроблюваного програмного забезпечення залежить від надійності операційної системи, під управлінням якої вона буде функціонувати, і розроблюваного ПЗ.

1.5.3. Вимоги до складу та параметрів технічних засобів

Для коректної роботи на мобільних пристроях необхідно:

– версія Android 5.0 і вище;

– процесор – Qualcomm Snapdragon 439;

– кількість ядер – 4 ядра;

– частота процесора – 4 × 1,45 ГГц;

– графічний процесор – Adreno 505;

– оперативна пам’ять – 2 Гб.

1.5.4. Вимоги до інформаційної та програмної сумісності

Для розробленої програми додаткові програмні засоби не потрібні, бо вона має власний інтерфейс і працює у своїх межах. Для роботи програми необхідне підключення до мережі Internet зі швидкістю 20 Мбіт/с.

РОЗДІЛ 2
ПРОЕКТУВАННЯ ТА РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1. Функціональне призначення системи

Мобільний додаток призначений для спрощення ведення фінансового обліку. Програма має великий функціонал для детального ведення бюджету, поглиблення в фінансову грамотність і поліпшення матеріального становища.

Для того, щоб почати роботу з програмою користувачу необхідно створити власний акаунт або авторизуватися. Для реєстрації необхідно вказати адресу пошти, обрати власний логін та пароль для подальшого входу.

Після авторизації робота з програмою розпочинається із заповнення відомостей по доходам, а саме користувачу необхідно ввести перший дохід. Після цього необхідно обрати необхідні витрати чи ввести ще раз доходи за категоріями, які у програмі наявні за замовчуванням, але є можливість додавати власні. Для створення власної категорії, користувач повинен ввести назву категорії. Така ж можливість наявна і для витрат.

Для того, щоб додавати витрати необхідно вказати суму та категорію.

Окрім базових функцій, наявні і додаткові:

– діаграми – можна переглянути і проаналізувати свої витрати, за день, місяць, рік;

– доступ з інших пристроїв – щоб мати можливість стежити за своїм бюджетом з будь-якого пристрою (присутня синхронізація).

2.2. Опис застосованих математичних методів

Під час проєктування та розробки даного мобільного додатку використовувалися прості арифметичні операції над даними системи (розрахунок суми доходів, розходів, середньомісячні та середньорічні

розрахунки).

2.3. Опис використаних технологій та мов програмування

Ринку мобільних додатків вже більше десяти років, однак він досі бурхливо розвивається. Попит на створення мобільних додатків постійно зростає і він все ще помітно перевищує пропозицію. Для створення таких додатків є досить великий вибір платформ та інструментів. Одним з таких представників є платформа компанії Microsoft – Xamarin. У ній використовується стандартна для Enterprise – розробки мова програмування C#, кросплатформне середовище розробки – Visual Studio. На виході – нативний додаток для Android. Для розробки додатка використовуємо – Visual Studio 2019, Xamarin, C# [1].

Xamarin – це фреймворк для розробки мобільних додатків (iOS, Android, Windows Phone) з використанням мови C#, із застосуванням всіх звичних мовних особливостей таких як LINQ, лямбда-виразів, Generic і async. При цьому мається повний доступ до всіх можливостей SDK платформи і рідного механізму створення UI, отримуючи на виході додаток, який, нічим не відрізняється від нативних і не поступається їм у продуктивності.

Фреймворк складається з декількох основних частин:

– Xamarin.IOS – бібліотека класів для C#, що надає розробнику доступ до iOS SDK;

– Xamarin.Android – бібліотека класів для C#, що надає розробнику доступ до Android SDK;

– компілятори для iOS і Android;

– IDE Xamarin Studio;

– плагін для Visual Studio.

Для виконання додатків в Android використовується віртуальна Java-машина Dalvik. Нативні додатки, які пишуться на Java, компілюються в якийсь

проміжний байт-код, який інтерпретується Dalvik в команди процесора у момент виконання програми. Це так звана Just-in-time компіляція (компіляція на льоту, рис. 2.1).

Xamarin враховує різницю між платформами для розробки, надаючи окремі компілятори для кожної з цих платформ, які дозволяють на виході отримувати справжні, нативні додатки, що виконуються поза контекстом браузера і можуть використовувати всі апаратні і програмні ресурси платформи.

Рис. 2.1. Виконання програм Xamarin для Android

Для Xamarin як середовище розробки використовують або власну IDE – Xamarin Studio, або Visual Studio. Xamarin пропонує можливість вести розробку в Visual Studio після встановлення спеціального плагіна. Плюси очевидні – становлення розробником мобільних додатків, не змінюючи місця дислокації, і можна використовувати всю важку артилерію в особі Resharper та інших плагінів. Після установки плагіна для Visual Studio необхідно налаштувати з'єднання з ПК, яке буде використано при запуску проекту на виконання. Тобто після запуску, додаток автоматично пересилається на ПК, де компілюється і завантажується або на симулятор, або на пристрій, при цьому сам процес, процес налагодження, розстановка брейкпоінтів тощо, буде відбуватися в

Visual Studio. На поточний момент технологія Xamarin є серйозним інструментом для вирішення складних завдань в області розробки мобільних додатків [2].

На сьогоднішній момент мова програмування C# одна з найпотужних, що швидко розвиваються і затребуваних мов в IT-галузі. На даний момент на ній пишуться різноманітні програми: від невеликих десктопних програм до великих веб-порталів і веб-сервісів, які обслуговують щодня мільйони користувачів.

C# – об'єктно-орієнтована мова програмування. Створено як мову розробки додатків для платформи Microsoft .NET Framework. C# відноситься до сім'ї мов з C-подібним синтаксисом, з них його синтаксис найбільш близький до C++ і Java. Мова має статичну типізацію, підтримує поліморфізм, перевантаження операторів (в тому числі операторів явного і неявного приведення типу), делегати, атрибути, події, властивості, узагальнені типи і методи, ітератори, анонімні функції з підтримкою замикань, LINQ, виключення, коментарі у форматі XML.

Перейнявши багато від своїх попередників – мов C++, Pascal, Модула, Smalltalk і, особливо, Java – C#, спираючись на практику їх використання, виключає деякі моделі, що зарекомендували себе як проблематичні при розробці програмних систем, наприклад, C# на відміну від C ++ не підтримує множинне успадкування класів (між тим допускається множинне спадкування інтерфейсів).

C# є об'єктно-орієнтованим і в цьому плані багато перейняв у Java C++. Наприклад, C# підтримує поліморфізм, успадкування, перевантаження операторів, статичну типізацію. Об'єктно-орієнтований підхід дозволяє вирішити завдання з побудови великих, але в той же час гнучких, масштабованих і розширюваних додатків. І C# продовжує активно розвиватися, і з кожною новою версією з'являється все більше цікавого функціоналу, як,

наприклад, лямбда, динамічне зв'язування, асинхронні методи тощо. C# стандартизований в ECMA (ECMA-334) і ISO (ISO/IEC 23270).

Відомо як мінімум про три незалежні реалізації C#, що базуються на цій специфікації і знаходяться в даний час на різних стадіях розробки:

– Mono, початa компанією Ximian, продовжена її покупцем і наступником Novell, а потім Xamarin;

– dotGNU і Portable.NET, що розробляються Free Software Foundation.

Коли говорять C#, нерідко мають на увазі технології платформи .NET (Windows Forms, WPF, ASP.NET, Xamarin). І, навпаки, коли говорять .NET, нерідко мають на увазі C#. Однак, хоча ці поняття пов'язані, ототожнювати їх невірно. Мова C# була створена спеціально для роботи з фреймворком .NET, але саме поняття .NET дещо ширше.

Якось Білл Гейтс сказав, що платформа .NET – це найкраще, що створила компанія Microsoft. Можливо, він мав рацію. Фреймворк .NET представляє потужну платформу для створення додатків. Можна виділити наступні її основні риси:

– підтримка декількох мов – основою платформи є загальномовне середовище виконання Common Language Runtime (CLR), завдяки чому .NET підтримує кілька мов: поряд з C# це також VB.NET, C++, F#, а також різні діалекти інших мов, прив'язані до .NET, наприклад, Delphi. При компіляції код, на будь-якій з цих мов, компілюється в збірку спільною мовою CIL (Common Intermediate Language) – свого роду асемблер платформи .NET. Тому ми можемо зробити окремі модулі однієї програми на окремих мовах;

– кросплатформність – .NET є платформою (з деякими обмеженнями). Наприклад, остання версія платформи на даний момент .NET Core підтримується на більшості сучасних ОС Windows, MacOS, Linux. Використовуючи різні технології на платформі .NET, можна розробляти

програми на мові C# для різних платформ – Windows, MacOS, Linux, Android, iOS, Tizen;

– потужна бібліотека класів – .NET представляє єдину для всіх підтримуваних мов бібліотеку класів. І який б додаток ми не збиралися писати на C# – текстовий редактор, чат або складний веб-сайт – так чи інакше ми задіємо бібліотеку класів .NET;

– різноманітність технологій – загальномовне середовище виконання CLR і базова бібліотека класів є основою для цілого стека технологій, який розробники можуть задіяти при побудові тих чи інших додатків. Наприклад, для роботи з базами даних в цьому стеку технологій призначена технологія ADO.NET і Entity Framework Core. Для побудови графічних додатків з багатим насиченим інтерфейсом – технологія WPF і UWP, для створення більш простих графічних додатків – Windows Forms. Для розробки мобільних додатків – Xamarin. Для створення веб-сайтів – ASP.NET тощо.

Також ще слід відзначити таку особливість мови C# і фреймворк .NET, як автоматичне прибирання сміття. А це означає, що нам в більшості випадків не доведеться, на відміну від C++, піклуватися про звільнення пам'яті [3].

У Xamarin застосовується мова розмітки XML.

XML (eXtensible Markup Language) – це спрощений діалект мови SGML, призначений для опису ієрархічних структур даних в World Wide Web. Він розробляється робочою групою W3C в 1996 р.; в даний час прийнятої рекомендацією є друга редакція мови XML 1.0 XML, безсумнівно, входить в обойму найбільш перспективних технологій WWW, чим пояснюється інтерес, який приділяється йому і корпораціями-розробниками, і широкою публікою.

Перш ніж перейти до його опису, видається доречним обговорити причини його появи та подальшого бурхливого розвитку. Спробуємо для цього поглянути на ті проблеми WWW, які повинні бути вирішені засобами нового покоління Веб-технологій.

XML – це спроба вирішити перераховані проблеми шляхом створення простої мови розмітки, що описує довільні структуровані дані. Точніше кажучи, це метамова, на якій пишуться спеціалізовані мови, що описують дані певної структури. Такі мови називаються XML-словниками. На відміну від HTML, XML не містить ніяких вказівок на те, як описані в XML-документі дані повинні відображатися. Спосіб відображення даних для різних пристроїв задається мовою опису стилів XSL, який грає для XML приблизно ту ж роль, що CSS дл HTML.

Інша принципова її відмінність від HTML полягає в тому, що XML може містити будь-які теги, які вважатимуть за потрібне використовувати творці XML-словника.

Процес обробки XML-документа полягає в наступному. Його текст аналізується спеціальною програмою, яка називається XML-процесором. XML-процесор нічого не знає про семантику даних у документі, він тільки виробляє синтаксичний розбір (parsing) тексту документа і перевіряє його правильність з точки зору правил XML. Якщо документ правильно оформлений (well-formed), то результати розбору тексту передаються XML-процесором прикладній програмі, яка виконує їх змістовну обробку, якщо ж документ оформлений невірно, тобто містить синтаксичні помилки, то XML-процесор повинен повідомити про них користувачеві. HTML не висловлює змісту документів [4].

В операційних системах: Android, починаючи із версії 5.0 і до останньої. Android – це одна з найпопулярніших і найбільш перспективних операційних систем для різних мобільних пристроїв. Система пропонує дуже зручний інструментарій і максимальну гнучкість налаштувань, що дозволяє кожному користувачеві смартфона або планшета на Android налаштувати його повністю під свої потреби. Розробка даної операційної системи стартувала в далекому 2003-му році, але по-справжньому відомою вона стала лише через 2 роки - після придбання компанією Google. Переломний момент в історії Android стався восени 2008-го року. Тоді компанія Google продемонструвала T-Mobile

G1, який став першим смартфоном під керуванням Android. Саме в той момент багато світових виробників звернули увагу на перспективну операційну систему.

Вже згадана ОС завжди позиціонувалася своїми розробниками як система з відкритим кодом. Це дозволяє будь-якому охочому створювати свої програми, ігри та інші додатки для розширення можливостей Android-гаджетів. Розробники спочатку продумали все так, щоб операційна система працювала максимально швидко навіть на самому «бюджетному» пристрої і це є безсумнівним плюсом.

Відкритістю системи охоче користуються і виробники мобільної електроніки, випускаючи власні призначені для користувача інтерфейси, наприклад, Sense від компанії HTC. Це робить гаджети на Android від різних виробників несхожими один на одного. Любителям класичного і «чистого» Android слід звернути свою увагу на пристрої Nexus. Саме вони традиційно першими отримують оновлення. Терміни оновлення інших пристроїв зазвичай затягуються через необхідність доопрацювання фірмових оболонок відповідно до особливостей нових версій ОС.

В цілому ж пристрої на Android мають всі функції, якими за негласними вимогам повинні володіти сучасні смартфони і планшети. Крім цього, функціонал з легкістю розширюється за допомогою додатків, віджетів або сторонніх прошивок.

Бажаючим придбати смартфон на Android необхідно пам'ятати, що багато функцій і програми орієнтовані на роботу з інтернетом. При відсутності доступу до Wi-Fi рекомендується підключити вигідний тариф для інтернет-користувачів або ж відключити деякі функції, що вимагають виходу в мережу.

Також не можна не відзначити обмежену автономність пристроїв на Android, особливо представників попередніх поколінь. При активному використанні смартфон або планшет доведеться заряджати щодня, а в деяких випадках і по 2 рази на добу. Виробники активно працюють над виправленням

цього недоліку і останнім часом на ринку стали з'являтися цікаві пристрої з помітно збільшеною автономністю. Не залишаються осторонь і розробники. З кожною новою версією операційна система «їсть» все менше заряду.

На підставі наведеної інформації можна виділити ключові переваги і недоліки даної операційної системи. Так, безперечними перевагами ОС Android є:

– відкритість програмного коду, що дає можливості для розробки практично будь-яких додатків та ігор;

– невимогливість до «заліза» пристрою;

– величезний асортимент додатків і захоплюючих ігор на будь-який смак;

– повна відповідність сучасним уявленням про функціонал смартфона;

– свобода дій для користувача. При бажанні система і її інтерфейс з легкістю налаштовуються під потреби власника;

– популярність операційної системи серед переважної більшості сучасних виробників. Це дозволяє користувачеві з практично будь-яким бюджетом купити функціональний і сучасний смартфон.

Є і свої недоліки. Ключовими є:

– відкритість програмного коду. Так, її можна одночасно віднести і до переваг, і до недоліків. Недоліком це є з тієї причини, що виробники мобільних пристроїв в більшості своїй вважають за краще створювати власні оболонки. Через це з'являються істотні тимчасові затримки між офіційним виходом оновлення ОС і її отриманням на різні пристрої;

– порівняно низька автономність. Останнім часом в даному напрямку був зроблений величезний крок вперед [5].

2.4. Опис структури системи та алгоритмів її функціонування

2.4.1. Проєктування системи

Use-case діаграма (діаграма прецедентів) необхідна для відображення різних сценаріїв взаємодії між акторами і прецедентами (рис. 2.2).

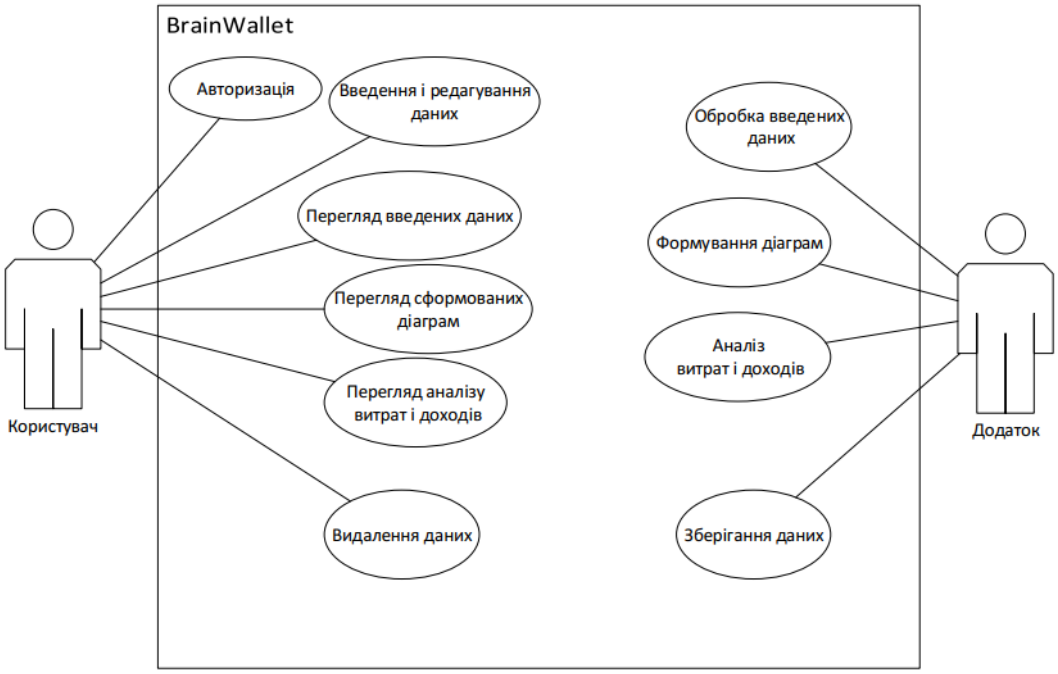

Рис. 2.2. Use-case діаграма додатку

При проектуванні мобільного додатку «BrainWallet» була розроблена БД, термінологія якої використовуються в даній предметній області:

– користувачі (users) – перелік інформації про користувачів;

– категорії доходів (income_category) – перелік інформації про категорії доходів;

– категорії витрат (consumption_category) – перелік інформації про категорії витрат;

– доходи (income) – перелік інформації про доходи користувача;

– витрати (consumption) – перелік інформації про витрати користувача;

– грошові символи (money) – перелік грошових символів.

При проектуванні використалась реляційна модель бази даних. Всі данні зберігаються у таблицях бази даних, які в свою чергу містять поля. Кожне поле таблиці має свій тип.

При проектуванні таблиць були використані такі типи даних та позначення:

– VARCHAR – символьне значення до 255 символів, кодування utf8_general_ci;

– DATA – дата;

– DECIMAL – число з плаваючою точкою;

– INT – ціле значення.

При встановленні зв’язку між таблицями використалися наступні позначення:

– PK – (Primary key) первинний ключ – унікальне поле;

– FK – (Foreign key) зовнішній ключ – зовнішній ключ дозволяє описати залежність ключа однієї таблиці від ключа іншої таблиці.

Нижче наведений опис сутностей у вигляді реляційних таблиць (табл. 2.1– 2.6)

Таблиця 2.1

Сутності «Користувачі» (users)

Призначення поля	Назва поля	Ключ	Тип	Довжина
Код користувача	id_user	PK	INT	11
Код валюти	id_money	FK	INT	11
Логін	user_login		VARCHAR	30
Пароль	user_pass		VARCHAR	50
Ім’я	user_name		VARCHAR	60
Прізвище	user_surname		VARCHAR	100
Електронна пошта	user_email		VARCHAR	50

Таблиця 2.2

Сутності «Категорії доходів» (income_category)

Призначення поля	Назва поля	Ключ	Тип	Довжина
Код категорії доходів	id_cat_income	PK	INT	11
Категорія доходів	cat_income		VARCHAR	50

Таблиця 2.3

Сутності «Категорії витрат» (consumption_category)

Призначення поля	Назва поля	Ключ	Тип	Довжина
Код категорії витрат	id_cat_consumption	PK	INT	11
Категорія витрат	cat_consumption		VARCHAI	50

Таблиця 2.4

Сутності «Доходи» (income)

Призначення поля	Назва поля	Ключ	Тип	Довжина
Код доходу	id_income	PK	INT	11
Код користувача	id_user_inc	FK	INT	11
Код категорії доходів	id_inc_cat	FK	INT	11
Сума	income		DECIMAL	10,2
Дата доходу	date_income		DATE	

Таблиця 2.5

Сутності «Грошові символи» (money)

Призначення поля	Назва поля	Ключ	Тип	Довжина
Код символу	id_money	PK	INT	11
Позначка символу	money		VARCHAR	20

Таблиця 2.6

Сутності «Витрати» (consumption)

Призначення поля	Назва поля	Ключ	Тип	Довжина
Код витрати	id_consumption	PK	INT	11
Код користувача	id_user_cons	FK	INT	11
Код категорії витрат	id_cons_cat	FK	INT	11
Сума	consumption		DECIMAL	10,2
Дата витрати	date_cons		DATE	

ER-модель – це семантична модель даних, яка призначена для спрощення процесу проектування бази даних. З ER-моделі можуть бути породжені всі види баз даних: реляційні, ієрархічні, мережні, об’єктні. В основі ER-моделі лежать поняття “сутність”, “зв’язок” та “атрибут” [6].

Первинний ключ – це поле, яке використовується для забезпечення унікальності даних в таблиці (PK).

Зовнішній ключ – це одне або декілька полів (атрибутів), які є первинними в іншій таблиці і значення яких замінюється значеннями первинного ключа іншої таблиці (FK) [6].

На схемі зображені зв’язки один-до-багатьох між батьківськими та дочірніми таблицями. Зв’язок типу один-до-багатьох означає, що один екземпляр першої сутності зв’язаний з декількома екземплярами другої (рис. 2.3).

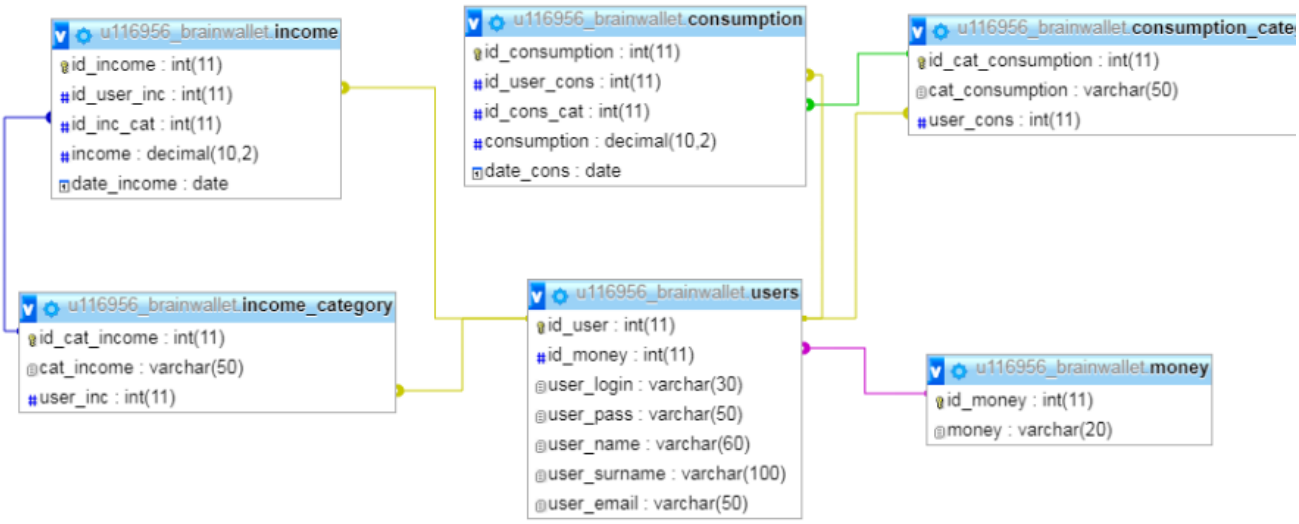

Рис. 2.3. ER-діаграма БД системи

2.4.2. Опис алгоритму функціонування програми

Для забезпечення всіх зазначених в призначенні розробки вимог до змісту і функціональних характеристик, що пред'являються до мобільного додатку , розроблений алгоритм функціонування програми (рис.2.4.):

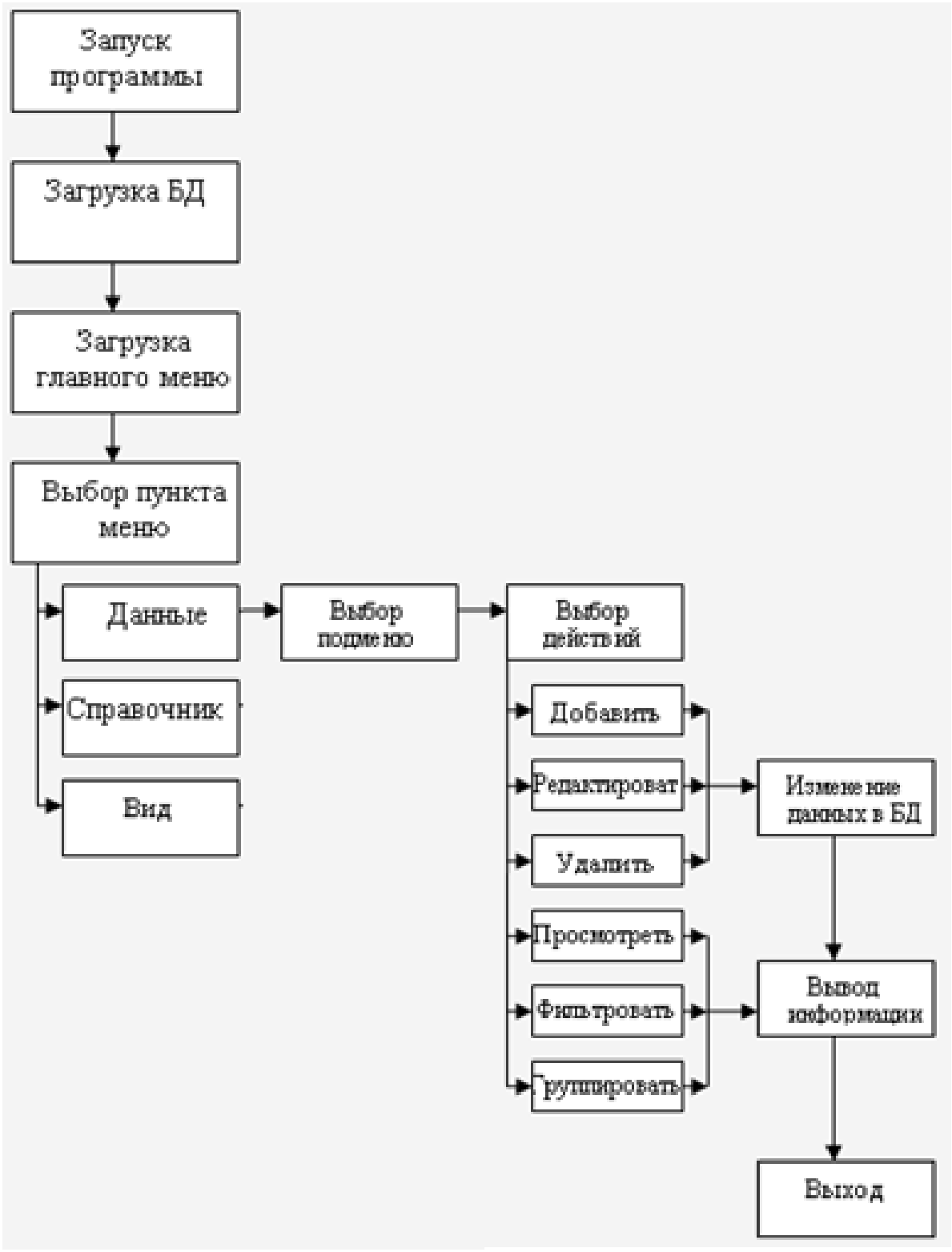

Рис. 2.4. Схема алгоритму функціонування програми

2.4.3. Створення сторінок

При запуску програми «WelcomePage» (рис. 2.5), події компонентів сторінки відображено у табл. 2.7.

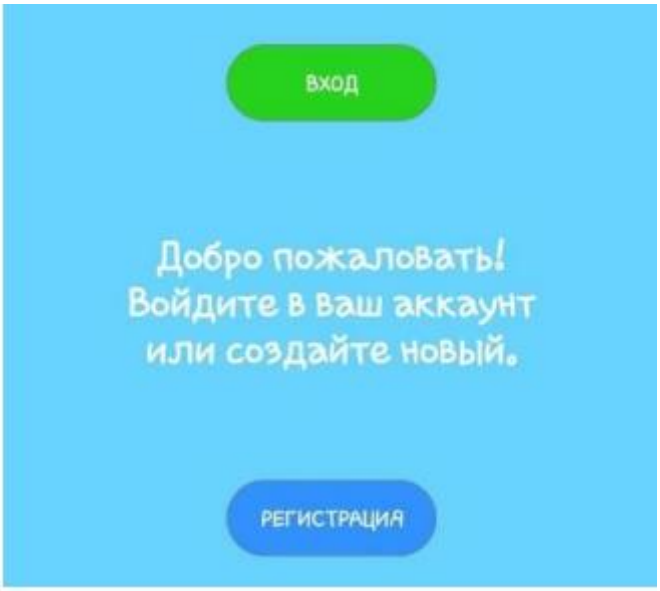

Рис. 2.5. Ескіз сторінки «WelcomePage»

Таблиця 2.7

Події сторінки «WelcomePage»

Подія	Призначення
btn_Signin_Clicked()	Перехід на сторінку авторизації
btn_register_Clicked()	Перехід на сторінку реєстрації
OnBackButtonPressed()	Вихід із програми

Для реєстрації користувача у додатку передбачена сторінка «RegisterPage» (рис. 2.6). Події компонентів сторінки відображено у таблиці 2.8.

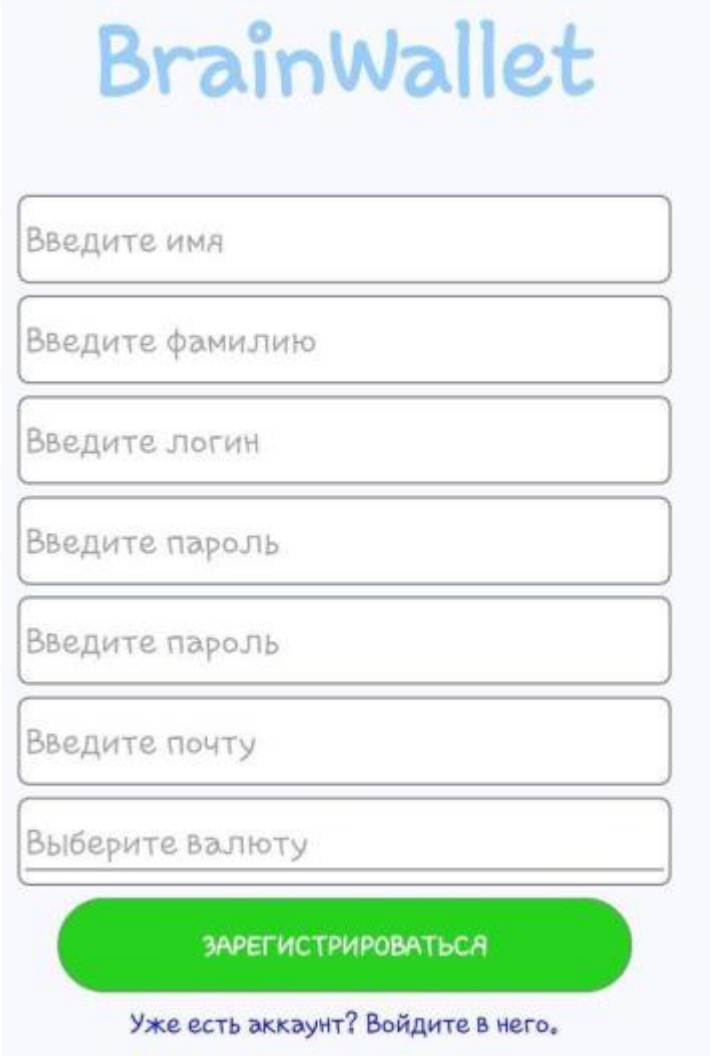

Рис. 2.6. Ескіз сторінки «RegisterPage»

Таблиця 2.8

Події сторінки «RegisterPage»

Подія	Призначення
currency()	Завантажує валюту з серверу
btn_register_Clicked()	Приймає введенні користувачем дані і реєструє у системі
IsUserExists()	Перевірка існування логіну

Для авторизації користувача у додатку створена сторінка «LoginPage» (рис. 2.7). Події компонентів сторінки відображено у таблиці 2.9.

Рис. 2.7. Ескіз сторінки «LoginPage»

Таблиця 2.9

Події сторінки «LoginPage»

Подія	Призначення
btn_Signin_Clicked()	Приймає від користувача та дозволяє або забороняє вхід логін і пароль

Після того, коли користувач авторизується на сторінці «LoginPage» йому відкриється головна сторінка програми «MainPageDetail» (рис. 2.8). Події компонентів сторінки відображено у таблиці 2.10.

Рис. 2.8. Ескіз сторінки «MainPageDetail»

Таблиця 2.10

Події сторінки «MainPageDetail»

Подія	Призначення
currency()	Завантажує валюту для відображення на головній
money()	Виконує підрахунок грошей
btn_income_Clicked()	Відкриває сторінку додання доходів
btn_cons_Clicked()	Відкриває сторінку додання витрат

На головній сторінці розміщена кнопка «Новый доход», яка відкриває сторінку для заповнення доходів «IncomePage» (рис. 2.9). Події компонентів сторінки відображено у таблиці 2.11.

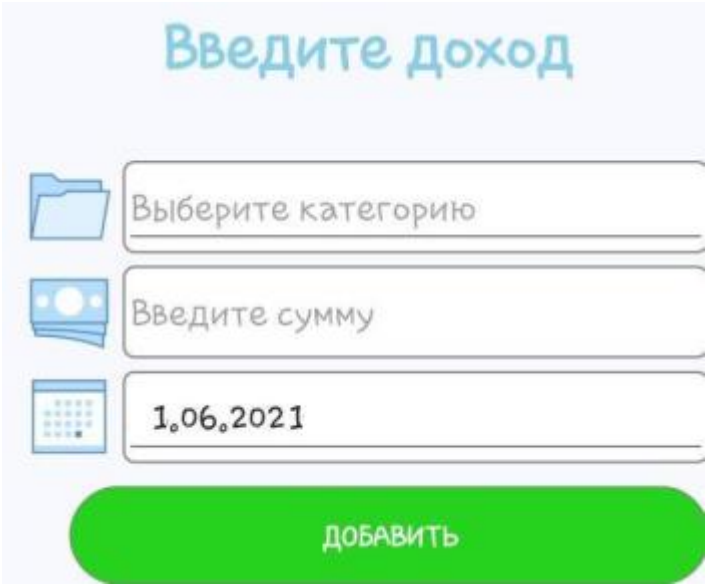

Рис. 2.9. Ескіз сторінки «IncomePage»

Таблиця 2.11

Події сторінки «IncomePage»

Подія	Призначення
category()	Завантажує список категорій доходів
btn_income_Clicked()	Додає дохід

На головній сторінці розміщена кнопка «Новый расход», яка відкриває сторінку для заповнення доходів «ConsumptionPage» (рис. 2.10). Події компонентів сторінки відображено у таблиці 2.12.

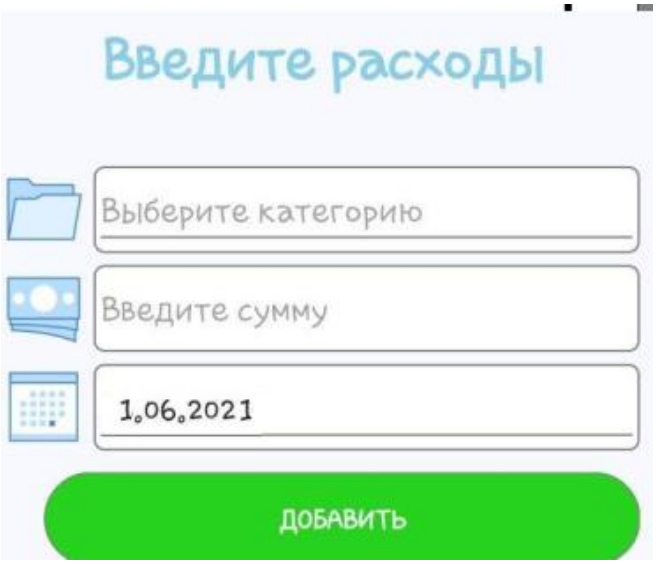

Рис. 2.10. Ескіз сторінки «ConsumptionPage»

Таблиця 2.12

Події сторінки «ConsumptionPage»

Подія	Призначення
category()	Завантажує список категорій доходів
btn_cons_Clicked()	Додає витрати

У бічному меню є сторінки «DayPage», «MonthPage», «YearPage», «AllReportsPage», які відповідають за формування звітів відповідно за день, місяць, рік та увесь час. Усі ці сторінки подібні одна до одної (рис. 2.11). Події компонентів сторінок відображено у таблиці 2.10.

Рис. 2.11. Ескіз сторінок

Таблиця 2.13

Події сторінок «DayPage», «MonthPage», «YearPage», «AllReportsPage»

Подія	Призначення
Pustota()	Функція, яка перевіряє, щоб списки не були порожніми
MainDayHelper()	Заповнює першу сторінку доходами і витратами за певну дату
IncomeDayHelper()	Заповнює другу сторінку доходами за певну дату
ConsumptionDayHelper()	Заповнює третю сторінку витратами за певну дату
OnItemSelected()	Перехід на редагування доходів
OnItemSelected1()	Перехід на редагування витрат
DateSelected()	Вибір дати для заповнення даними, оновлення даних

Сторінки «DetailPage» і «DetailPage1» подібні одна до одної, вони необхідні для редагування категорій (рис. 2.12). Події компонентів сторінок відображено у таблиці 2.14.

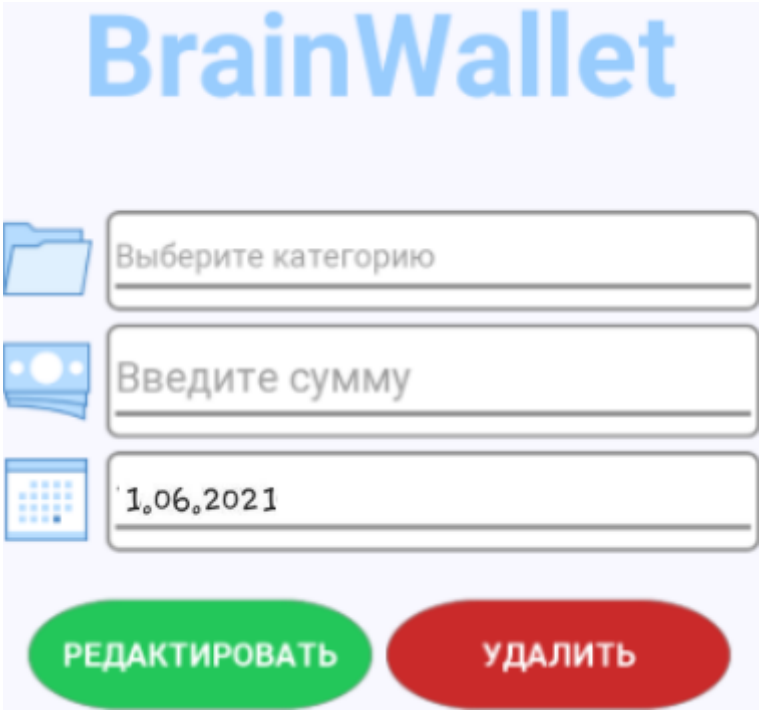

Рис. 2.12. Ескіз сторінок «DetailPage» і «DetailPage1»

Таблиця 2.14

Події сторінок «DetailPage» і «DetailPage1»

Подія	Призначення
categoryinc()	Завантажує список категорій
update_Clicked()	Оновлює дані
delete_Clicked()	Видаляє дані

На сторінці налаштувань «Settings» можна редагувати свій аккаунт або додати нові категорії (рис. 2.13). Події компонентів сторінок відображено у таблиці 2.15.

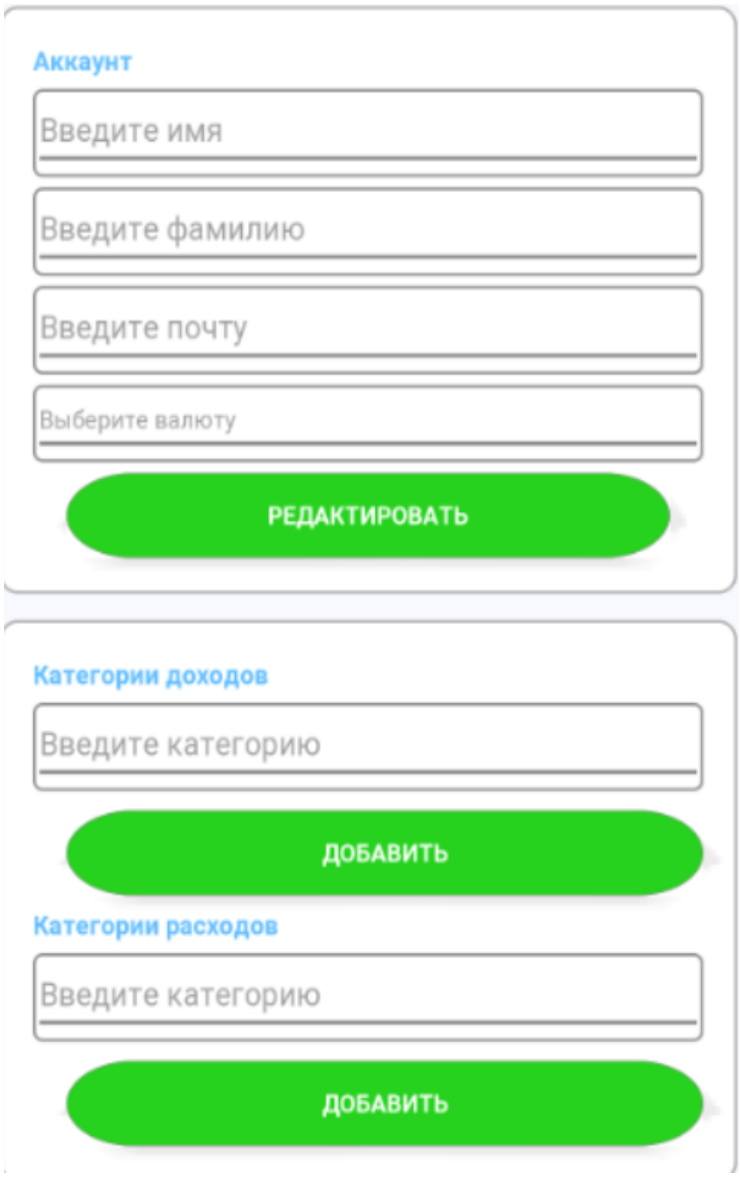

Рис. 2.13. Ескіз сторінки «Settings»

Таблиця 2.15

Події сторінки «Settings»

Подія	Призначення
currency()	Завантажує валюту
UserUpd()	Оновлює дані користувача
btn_cons_Clicked()	Додає нову категорію доходів
btn_income_Clicked()	Додає нову категорію витрат

2.4.4. Опис файлової структури додатку

Програмний додаток розроблений у середовищі програмування MS Visual Studio, фреймворк Xamarin, на мові програмування C# і мовою розмітки сторінки XML. Програма складається із сторінок, де кожна сторінка має свої файли опису – .xaml і .xaml.cs.

У даному програмному додатку використовуються файли з таким розширення як:

– .cs – містить допоміжний код, який використовуються програмою;

– .xaml – описує компоненти сторінки і їх розміщення на ній;

– .xaml.cs – містить код, який необхідний для коректної роботи сторінок програми.

У програмі були використані наступні компоненти, які описані у файлах з розширенням .xaml:

– Button – кнопка, необхідна для певних дій; DatePicker – компонент для вибору дати; Entry – поле для введення даних;

– Image – для відображення малюнків;

– Label – для виведення тексту або допоміжної інформації на екран;

– ListView – список елементів; Picker – для вибору елементу; Frame – контейнер;

– ScrollView – контейнер з прокруткою; Grid – сітка.

Файлова структура програмного додатку представлена на рис. 2.14.

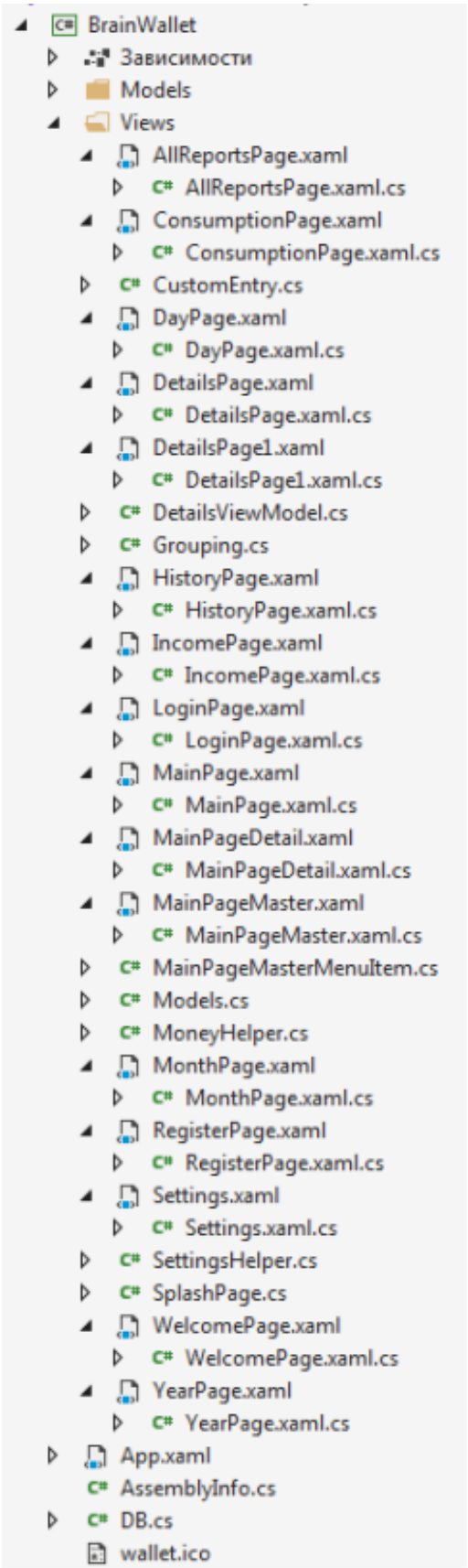

Рис. 2.14. Файлова структура програми BrainWallet

2.5. Обґрунтування та організація вхідних та вихідних даних програми

Вхідними даними програми є: логін, пароль і пошта користувача, відомості по рахунку, доходи, витрати, власні категорії.

Вихідними даними програми є: аналіз витрат за день, місяць, рік.

Всі дані зберігаються в базі даних проєкту.

2.6. Опис роботи розробленої системи

2.6.1. Використані технічні засоби

Для роботи даного додатку необхідно мобільний пристрій (наприклад, смартфон), що відповідає таким вимогам:

- операційна система Android версії 2.2 (Froyo) і вище;
- оптимальна роздільна здатність дисплею – 1280*720;
- доступ до мережі Інтернет.

Для тестування та демонстрації роботи даного додатку використовувався мобільний пристрій з наступними характеристиками:

Смартфон Huawei Y8 2020 (CRO-U00) DualSim з наступними характеристиками: екран (5", TFT, 854x480)/ MediaTek MT6580M (1.3 ГГц)/ / RAM 4 ГБ/ 128 ГБ вбудованої пам'яті + підтримка microSD/SDHC (до 256 ГБ)/ 3G/ GPS/ підтримка 2-х SIM-карток (Micro-SIM)/ Android 6.0 (Marshmallow).

2.6.2. Використані програмні засоби

Програмний додаток розроблений у середовищі програмування MS Visual Studio, фреймворк Xamarin, на мові програмування C# і мовою розмітки сторінки XML. База даних конструювалася в СКБД Workbench 5.12.

2.6.3. Виклик та завантаження програми

Для того, щоб почати працювати у додатку його необхідно встановити на телефон або планшет на базі ОС Android. Файл встановлення має назву «com.companyname.brainwallet-Signed.apk» і важить 10 мб. Файл необхідно запустити і встановити програму на свій пристрій. Після встановлення програми вона буде важити 35 мб.

2.6.4. Опис інтерфейсу користувача

Після запуску програмного додатку BrainWallet з'являється привітальне вікно, у якому необхідно обрати подальший шлях роботи з ним (рис. 2.15).

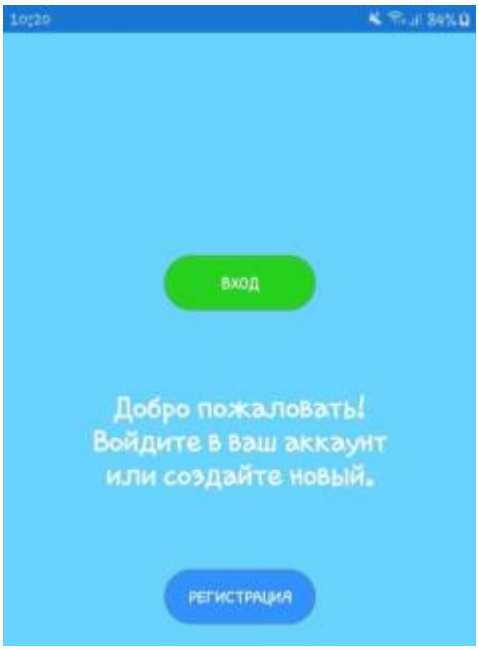

Рис. 2.15. Привітальна сторінка

Спочатку необхідно зареєструватися, тому натискаємо на кнопку «Регистрация». Після того, як ми натиснули на кнопку «Регистрация», з'являється вікно реєстрації (рис. 2.16) в якому необхідно ввести коротку

інформацію про себе, логін, пароль – це необхідно для зручності використання додатку у подальшому.

Рис. 2.16. Вікно реєстрації

Якщо помилитись і неправильно ввести повторний пароль, то додаток повідомить про це за допомогою відповідного повідомлення (рис. 2.17).

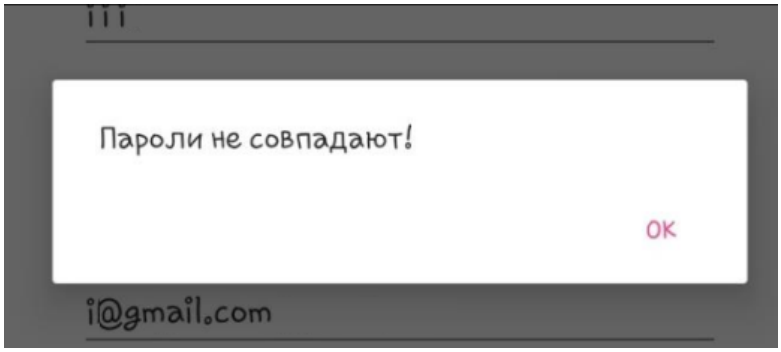

Рис. 2.17. Помилка введення пароля

Після того, якщо дані були правильно введені, то реєстрація завершується успішно і додаток видає відповідне повідомлення (рис. 2.18).

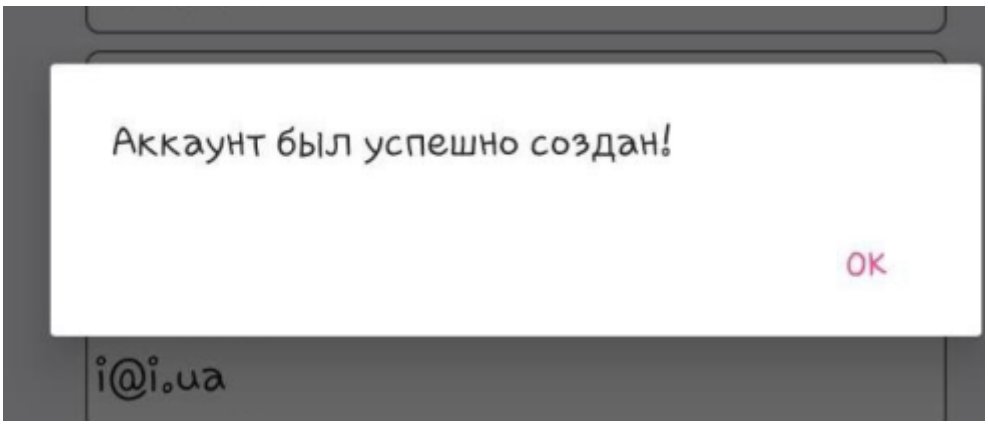

Рис. 2.18. Успішна реєстрація

Якщо інший користувач, при реєстрації, спробує ввести логін, який вже є у системі, то додаток повідомить про це за допомогою відповідного повідомлення (рис. 2.19).

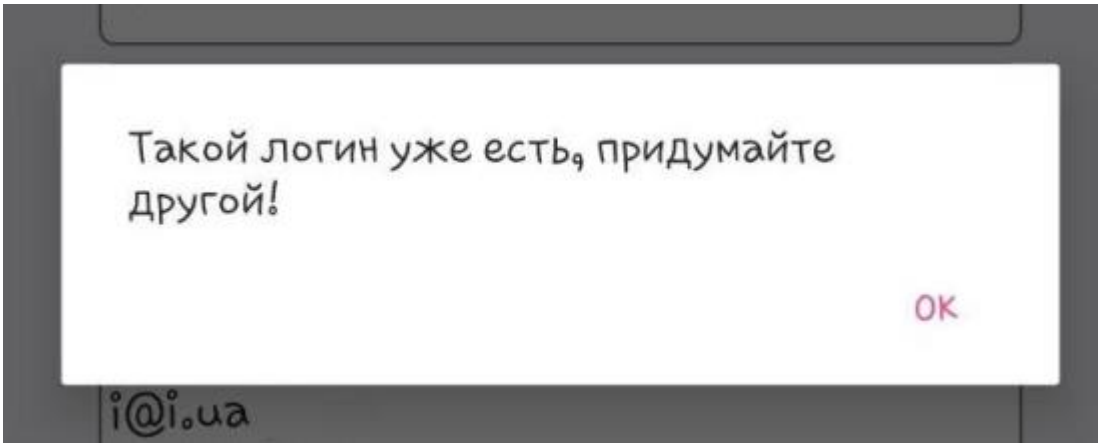

Рис. 2.19. Попередження про однаковість логінів

Після реєстрації можна успішно ввійти до власного аккаунту заповнивши відповідні поля: логін та пароль, який за замовчуванням прихований (рис. 2.20).

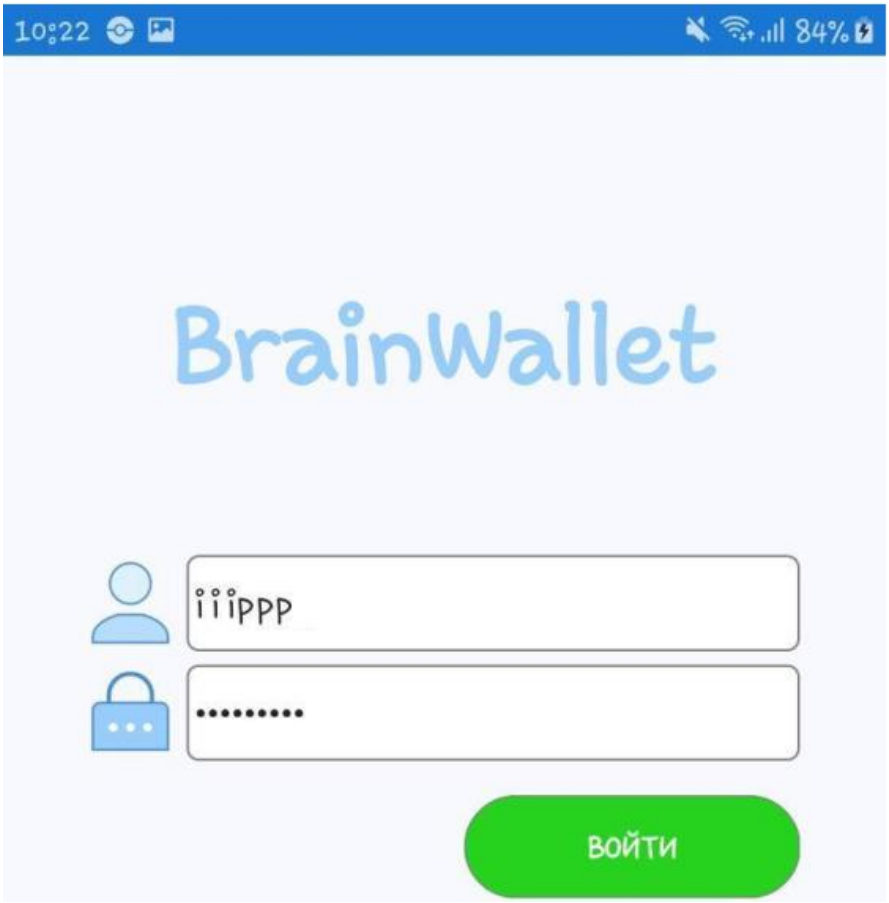

Рис. 2.20. Вхід до акаунту

Але якщо ввести неправильно логін чи пароль, то система видасть відповідне повідомлення (рис. 2.21).

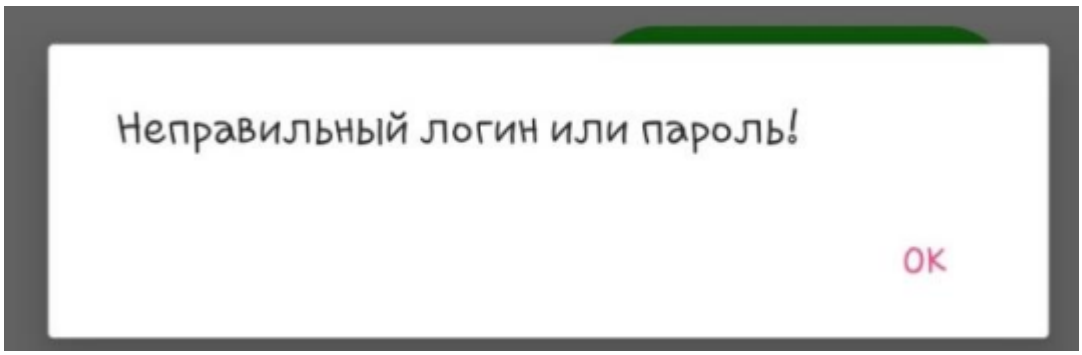

Рис. 2.21. Неуспішний вхід до аканту

Після авторизації користувач потрапляє на головну сторінку програми, на якій розташовані кнопки для додання доходів і витрат, інформація про ваш баланс та вкладка з меню (рис. 2.22).

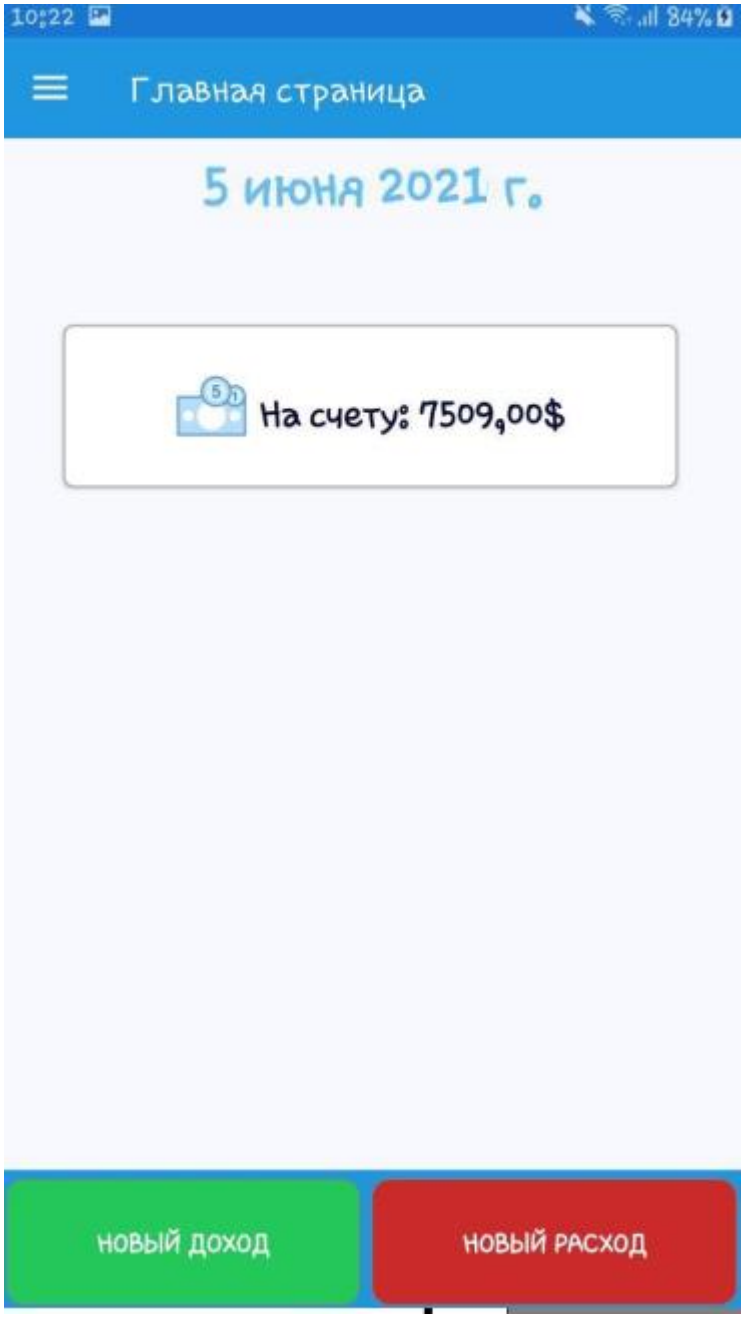

Рис. 2.22. Головна сторінка

Для того, щоб додати новий дохід необхідно натиснути на кнопку «Новый доход». Після того, коли відкриється сторінка необхідно обрати

категорію із випадаючого списку, ввести необхідну суму і вказати дату, натиснути кнопку «Добавить» (рис. 2.23 – 2.25).

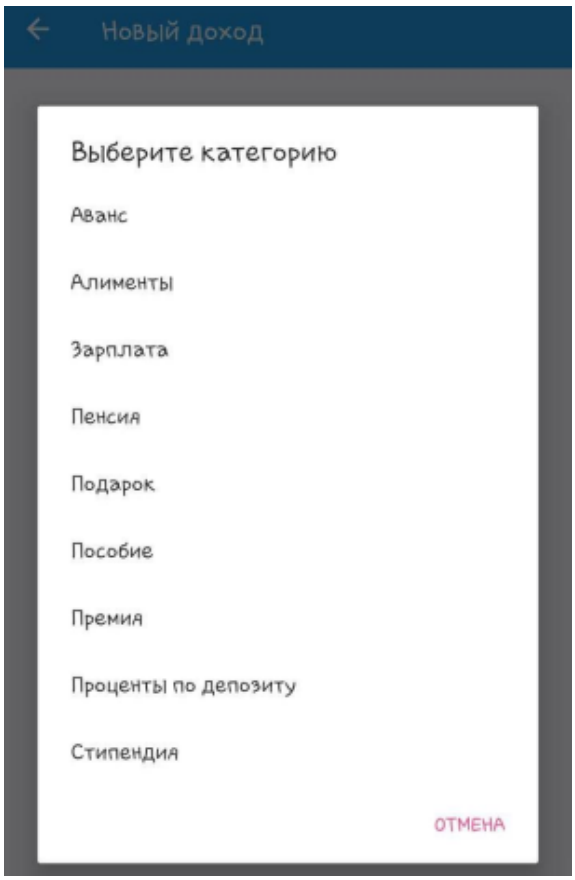

Рис. 2.23. Категорії доходів

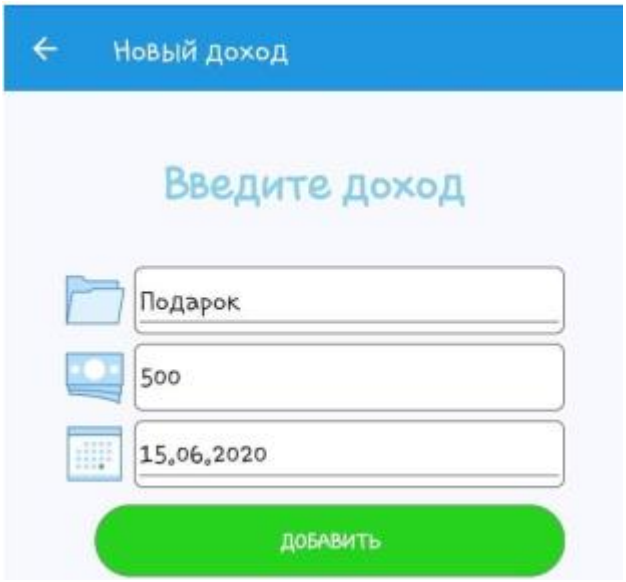

Рис. 2.24. Вікно введення доходів

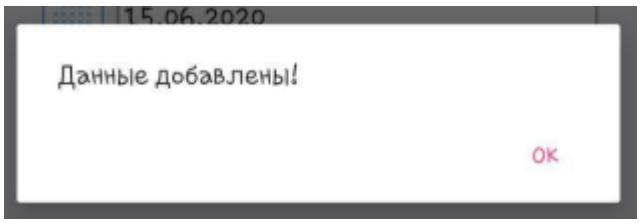

Рис. 2.25. Повідомлення при доданні

Для того, щоб додати нові витрати необхідно натиснути на кнопку «Новый расход». Після того, коли відкриється сторінка необхідно обрати категорію із випадаючого списку, ввести необхідну суму і вказати дату, натиснути кнопку «Добавить» (рис. 2.26 – 2.28).

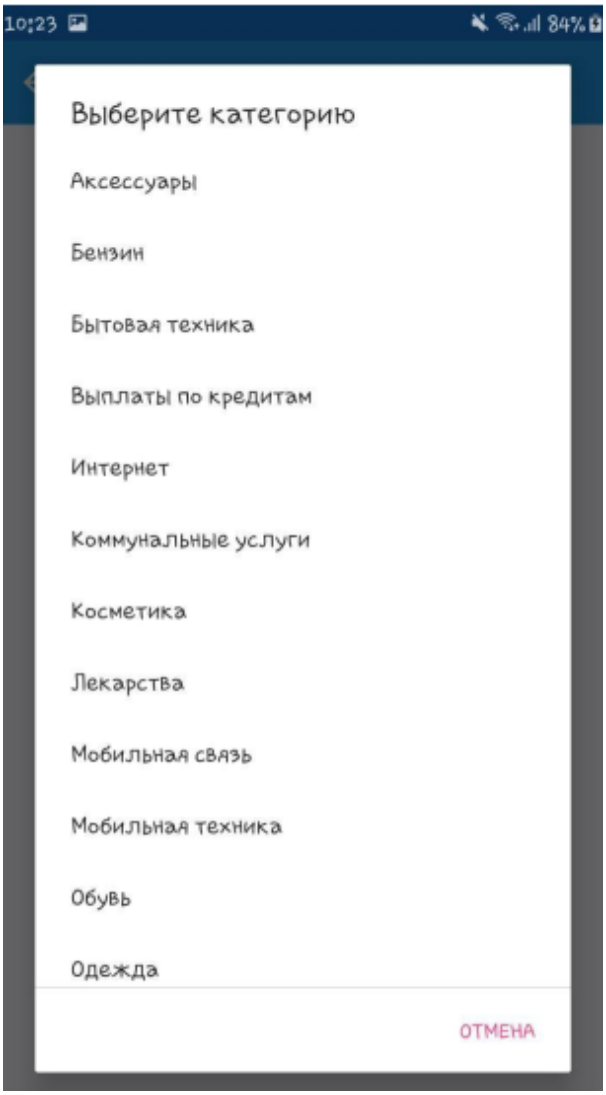

Рис. 2.26. Категорії доходів

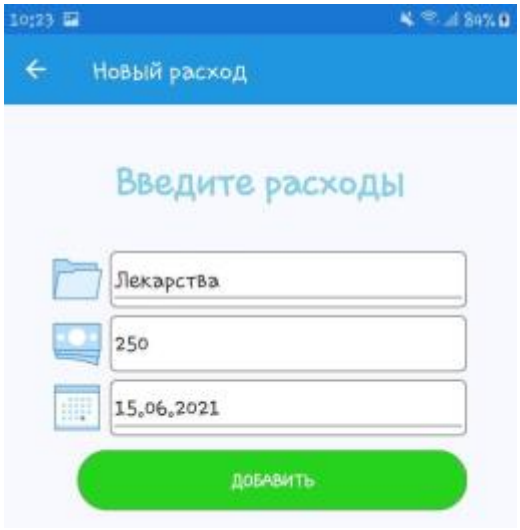

Рис. 2.27. Вікно введення витрат

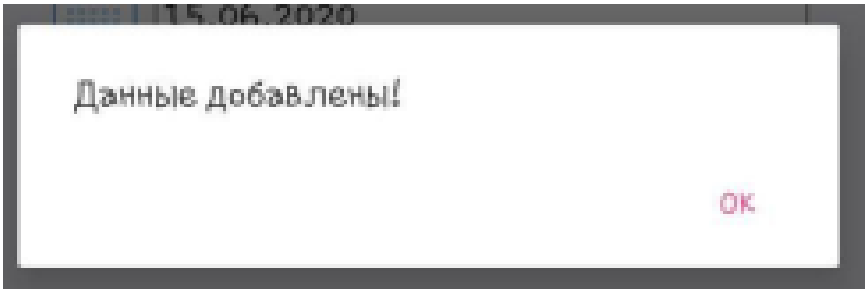

Рис. 2.28. Повідомлення при доданні

У програмі передбачено бічне меню, що містить вкладки, які необхідні для роботи з програмою (рис. 2.29).

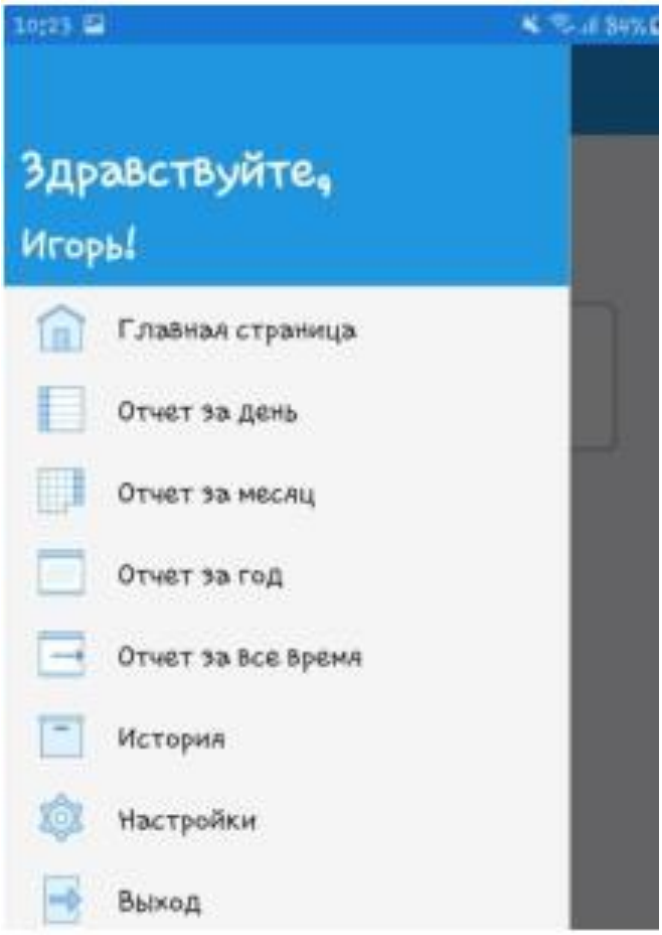

Рис. 2.29. Меню програми

Вкладки «Отчет за день», «Отчет за месяц», «Отчет за год», «Отчет за все время» подібні одна до одної, вони містять інформацію про витрати і доходи відповідно за день, місяць, рік, увесь час користування програмою. При чому, можна обрати будь-який день, місяць чи рік. Перегляд звітів на цих сторінках поділяється на 3 панелі: разом, окремо доходи, окремо витрати. Інформація на панелі «Разом» представляє собою список суми категорій доходів і суми категорій витрат, виходячи з чого виконується розрахунок суми за певний період (рис. 2.30). Інформація на панелі «Доходи» представляє собою список усіх доходів, а також відображення у вигляді діаграми за певний період (див. рис. 2.31 – 2.32). Інформація на панелі «Витрати» представляє собою список

усіх витрат і відображення у вигляді діаграми за певний період відповідно (див. рис. 2.33 – 2.35).

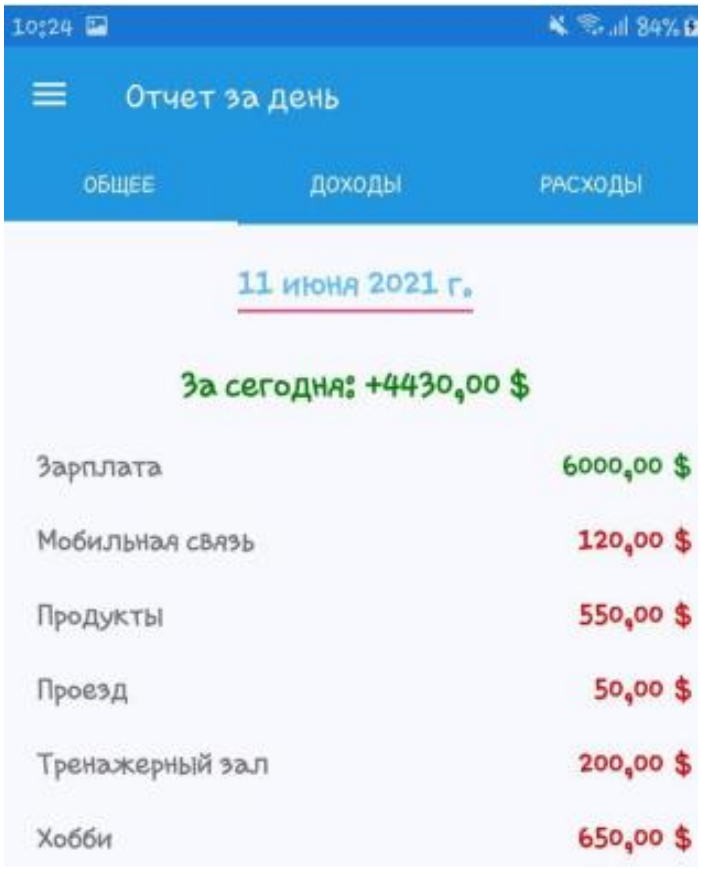

Рис. 2.30. Звіт за день

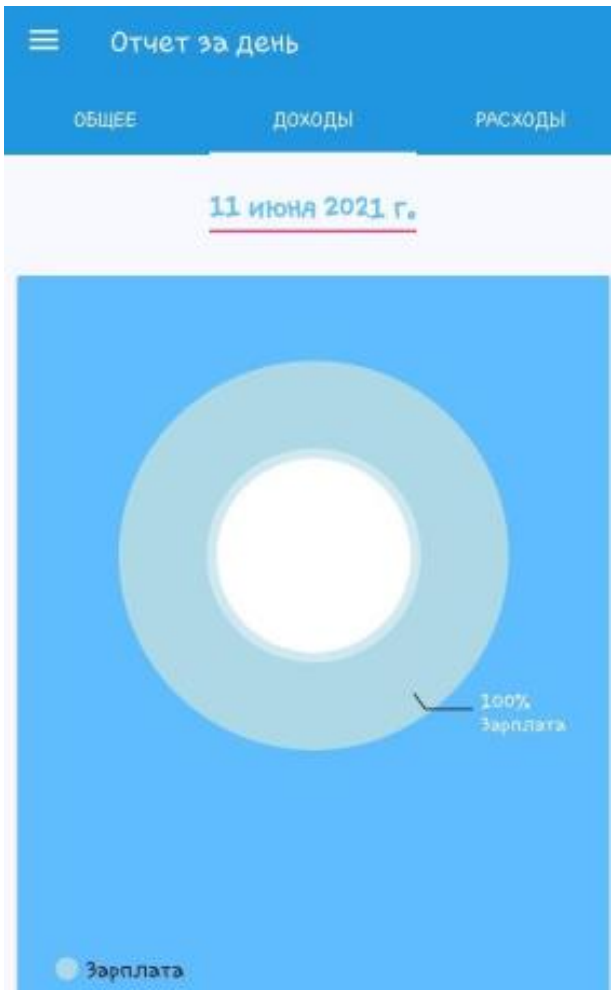

Рис. 2.31. Доходи за день (діаграма)

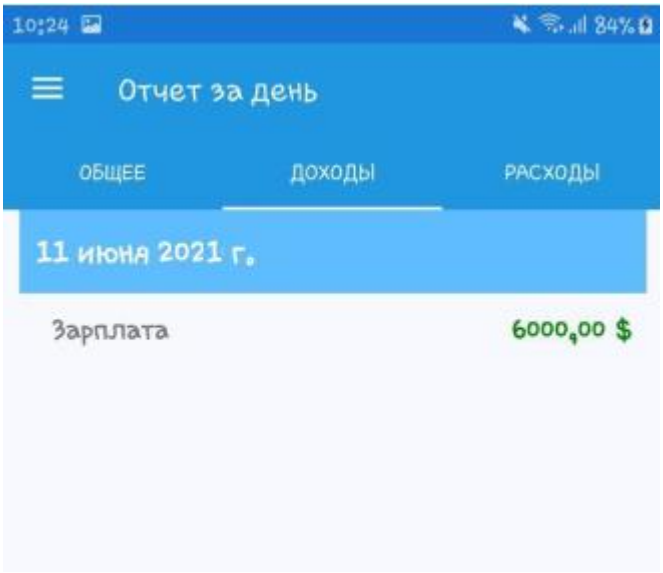

Рис. 2.32. Доходи за день

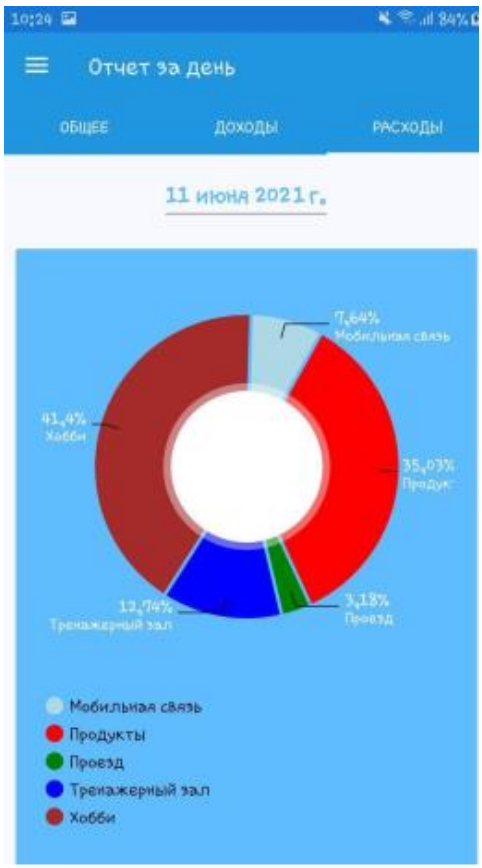

Рис. 2.33. Витрати за день (діаграма)

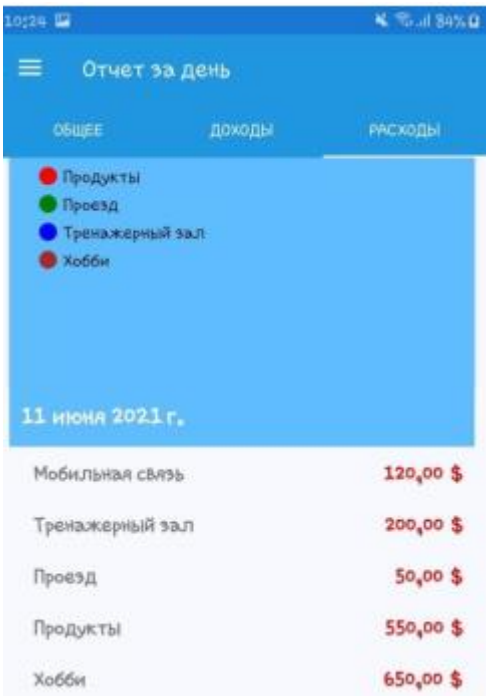

Рис. 2.34. Витрати за день

На вкладці «История» можна переглянути усі додані витрати та доходи, з можливістю видалення або редагування, для цього необхідно натиснути на потрібне поле (рис. 2.35 – 2.37).

Рис. 2.35. Історія доходів

Рис. 2.36. Історія витрат

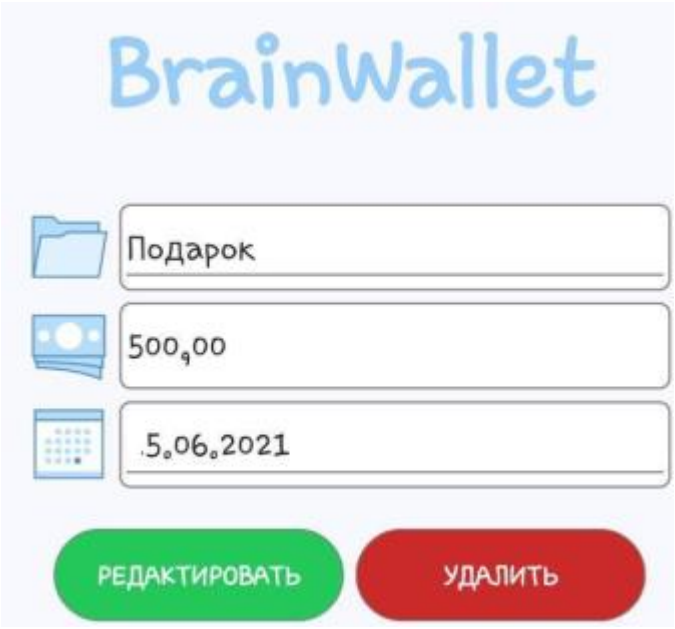

Рис. 2.37. Вікно редагування

Після того, як користувач потрапляє на сторінку редагування необхідно ввести нові дані та натиснути кнопку «Редактировать», якщо треба видалити дохід чи витрати, то відповідно можна натиснути на кнопку «Удалить».

На вкладці «Настройки» (рис. 2.38) можна виконати редагування профілю, для цього необхідно ввести ті дані, які користувач хоче оновити. Також на цій сторінці можна додавати власні категорії, для цього у відповідне поле необхідно ввести категорію і натиснути кнопку «Добавить» (рис. 2.39).

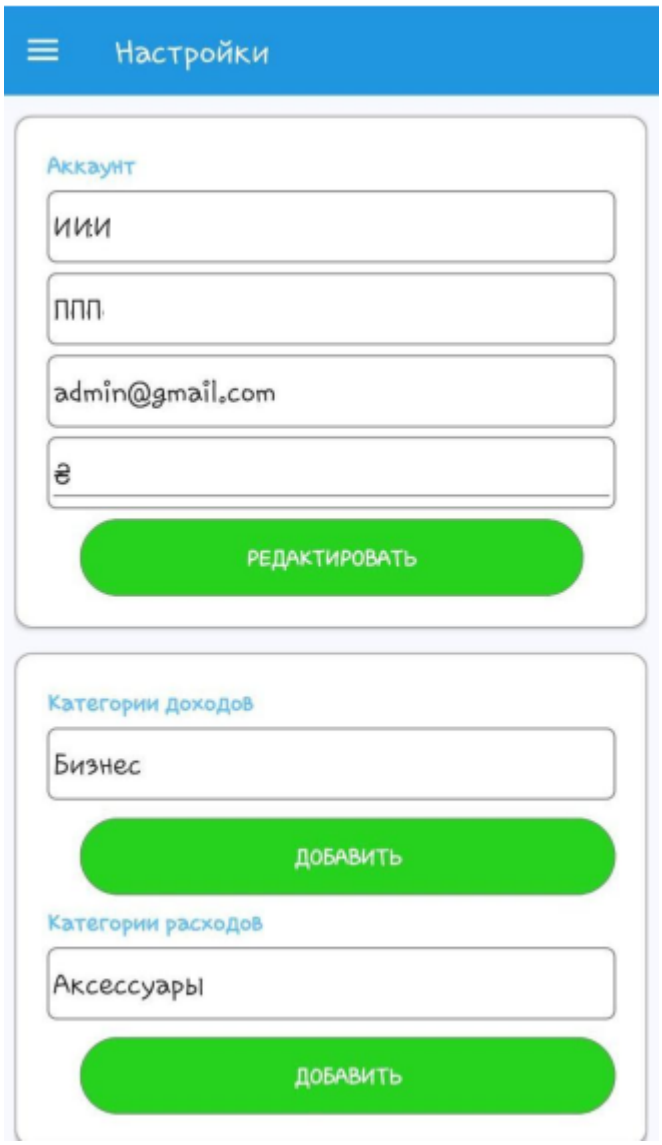

Рис. 2.38. Вікно налаштувань

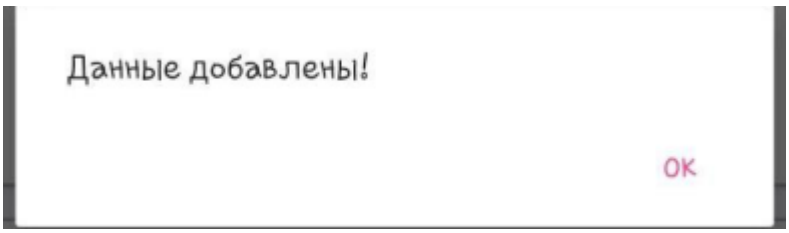

Рис. 2.39. Успішне додання даних

Якщо користувачу потрібно вийти з акаунту, то він може натиснути на кнопку із меню «Выход» (рис. 2.40).

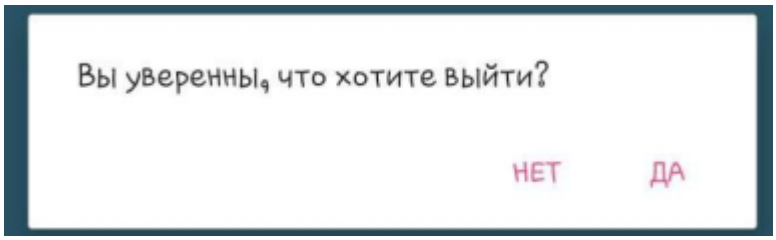

Рис. 2.40. Вкладка вихід з програми

РОЗДІЛ 3
ЕКОНОМІЧНИЙ РОЗДІЛ

3.1. Розрахунок трудомісткості та вартості розробки програмного продукту

Вхідні дані розробки програмного забезпечення:

- передбачуване число операторів – 860;
- коефіцієнт складності програми – 1,25;
- коефіцієнт кореляції програми в ході її розробки - 0,1;
- середня годинна заробітна плата програміста, грн/год – 40;
- вартість машино-години ЕОМ, грн/год – 7.

Нормування праці в процесі створення ПЗ істотно ускладнено в силу творчого характеру праці програміста. Тому трудомісткість розробки ПЗ може бути розрахована на основі системи моделей з різною точністю оцінки.

Трудомісткість розробки ПЗ можна розрахувати за формулою:

$$t = t_o + t_и + t_a + t_п + t_{отл} + t_д \text{, людино-годин,} \quad (3.1)$$

де t_o – витрати праці на підготовку й опис поставленої задачі (приймається 50);

$t_и$ – витрати праці на дослідження алгоритму рішення задачі,

t_a – витрати праці на розробку блок-схеми алгоритму,

$t_п$ – витрати праці на програмування по готовій блок-схемі,

$t_{отл}$ – витрати праці на налагодження програми на ЕОМ,

$t_д$ – витрати праці на підготовку документації.

Складові витрати праці визначаються через умовне число операторів у ПЗ, яке розробляється.

Умовне число операторів (підпрограм):

$$Q = q \cdot C(1 + p), \tag{3.2}$$

де q – передбачуване число операторів,

С– коефіцієнт складності програми,

р– коефіцієнт кореляції програми в ході її розробки.

$$Q = 860 \cdot 1{,}25 \cdot (1 + 0{,}1) = 1182;$$

Витрати праці на вивчення опису задачі $t_и$ визначається з урахуванням уточнення опису і кваліфікації програміста:

$$t_u = \frac{Q \cdot B}{(75...85)K}, \text{ людино-годин}, \tag{3.3}$$

де В – коефіцієнт збільшення витрат праці внаслідок недостатнього опису задачі, В=1.2 ... 1.5,

К – коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності. до 2 – 0,8.

$$t_u = \frac{1182 \cdot 1{,}2}{85 \cdot 0{,}8} = 22, \text{ людино-годин.}$$

Витрати праці на розробку алгоритму рішення задачі:

$$t_a = \frac{Q}{(20...25)K}; \tag{3.4}$$

$$t_a = \frac{1182}{20 \cdot 0{,}8} = 73, \text{ людино-годин.}$$

Витрати на складання програми по готовій блок-схемі:

$$t_n = \frac{Q}{(20...25)K}; \quad (3.5)$$

$$t_n = \frac{1182}{25 \cdot 0{,}8} = 59, \text{ людино-годин.}$$

Витрати праці на налагодження програми на ЕОМ:

- за умови автономного налагодження одного завдання:

$$t_{отл} = \frac{Q}{(4...5)K}; \quad (3.6)$$

$$t_{отл} = \frac{1182}{4 \cdot 0{,}8} = 369, \text{ людино-годин,}$$

- за умови комплексного налагодження завдання:

$$t^{к}_{отл} = 1{,}2 \cdot t_{отл}; \quad (3.7)$$

$$t^{к}_{отл} = 1{,}2 \cdot 369 = 443, \text{ людино-годин}$$

Витрати праці на підготовку документації:

$$t_д = t_{др} + t_{до}; \quad (3.8)$$

де $t_{др}$ – трудомісткість підготовки матеріалів і рукопису

$$t_{др} = \frac{Q}{(15...20)K}; \quad (3.9)$$

$$t_{др} = \frac{1182}{15 \cdot 0{,}8} = 98, \text{ людино-годин.}$$

$t_{до}$ – трудомісткість редагування, печатки й оформлення документації:

$$t_{до} = 0{,}75 \cdot t_{др}; \quad (3.10)$$

$$t_{до} = 0{,}75 \cdot 98 = 73$$, людино-годин.

$$t_{д} = 98 + 73 = 172$$, людино-годин.

Отримаємо трудомісткість розробки програмного забезпечення:

t = 50 +22 + 73 +59 + 443 +172 = 821, людино-годин.

В результаті ми розрахували, що в загальній складності необхідно 821 людино-годин для розробки даного програмного забезпечення.

3.2. Розрахунок витрат на створення програми

Витрати на створення ПЗ Кпо включають витрати на заробітну плату виконавця програми Зз/п і витрат машинного часу, необхідного на налагодження програми на ЕОМ.

$$K_{ПО} = З_{ЗП} + З_{МВ}, \text{ грн,} \quad (3.11)$$

де $З_{ЗП}$ – заробітна плата виконавців, яка визначається за формулою:

$$З_{ЗП} = t \cdot C_{ПР}, \text{ грн,} \quad (3.12)$$

де t – загальна трудомісткість, людино-годин,

$С_{ПР}$ – середня годинна заробітна плата програміста, грн/година.

$$З_{ЗП} = 821 \cdot 40 = 32843, \text{грн.}$$

$З_{МВ}$ – Вартість машинного часу, необхідного для налагодження програми на ЕОМ:

$$З_{МВ} = t_{отл} \cdot C_{М}, \text{ грн,} \quad (3.13)$$

де $t_{отл}$ – трудомісткість налагодження програми на ЕОМ, год.

$С_{Мч}$ – вартість машино-години ЕОМ, грн/год.

$$З_{МВ} = 443 \cdot 7 = 3104 \text{, грн.}$$

$$К_{ПО} = 32843 + 3104 = 35947 \text{ , грн.}$$

Очікуваний період створення ПЗ:

$$T = \frac{t}{B_k \cdot F_p} \text{, мес. ,} \qquad (3.14)$$

де B_k - число виконавців;

$F_р$ - місячний фонд робочого часу (при 40 годинному робочому тижні Fр=176 годин).

$$T = \frac{821}{1 \cdot 176} = 4{,}6 \text{ міс.}$$

Таким чином, очікувана тривалість розробки складе 4,6 місяця, а витрати на створення програмного забезпечення 35947 грн.

ВИСНОВКИ

Життя сучасної людини динамічне. Людина ставить перед собою безліч цілей і прагне до їх досягнення. Динаміка і ритм сучасного життя, прагнення до успіху – вимагають від людини постійної зібраності, концентрації уваги, мобілізації всіх сил. Планування і грамотне ведення бюджету є однією з основних методик досягнення успіху. Однак, з огляду на потреби сучасної людини, метод виконання даних дій, якому буде віддаватися перевага, повинен бути простим у використанні, ефективним, а головне, не повинен вимагати великих затрат часу. Виходячи з цього, додаток "BrainWallet", який розроблявся в процесі кваліфікаційної роботи є актуальним і є відмінним засобом досягнення матеріального благополуччя.

Програмний додаток розроблений у середовищі програмування MS Visual Studio, фреймворк Xamarin, на мові програмування C# і мовою розмітки сторінки XML. База даних конструювалася в СКБД Workbench 5.12.

Додаток дозволяє скоротити час для ведення домашньої бухгалтерії, автоматизувати процес організації ведення обліку поточних коштів, отримувати візуальні характеристики руху ресурсів і підведення підсумків балансу сімейного бюджету.

Розроблена система може бути використана користувачами, зацікавленими в скороченні часу для ведення домашньої бухгалтерії і отриманні візуальних характеристик потоків фінансових ресурсів з метою найбільш правильного їх використання і ефективного розподілу.

В економічному розділі були проведені обчислення трудомісткості та витрат для розробки програмного забезпечення. Для створення даної інформаційної системи знадобиться 4,6 місяця, трудомісткість розробки ПЗ – 821 людино-годин, а витрати на її створення програмного забезпечення - 35947 грн.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. На чем лучше писать мобильные приложения URL: https://livetyping.com/ru/. дата звернення: 15.01.2020.
2. Подробно о Xamarin URL: https://habr.com/ru/post/188130/. дата звернення: 15.01.2020.
3. Язык программирования C# UML: https://metanit.com/. дата звернення: 1.03.2021.
4. Що таке XML URL: https://www.taina.com.ua. дата звернення: 18.01.2021.
5. Гриффитс Д. Head First. Программирование для Android. – Санкт-Петербург: Питер – 2016 – 912 с.
6. Фрайли К. SQL / Крис Фрайли. – Москва, 2003. – 456 с. – (ДМК Пресс). – (Quick Start).
7. Сколько украинцы ежемесячно тратят денег на еду URL: https://kp.ua/life/646329-polovyna-zarplaty-ukrayntsev-ukhodyt-na-edu. дата звернення: 15.01.2021.
8. Бібліографічний запис. Бібліографічний опис. Загальні вимоги та правила складання : (ГОСТ 7.1-2003, IDT) : ДСТУ ГОСТ 7.1:2006. – Чинний з 2007–07–01. – К. : Держспоживстандарт України, 2007. – 47 с. – (Система стандартів з інформації, бібліотечної та видавничої справи) (Національний стандарт України).
9. Бусигін Б.С., Коротенко Г.М., Коротенко Л.М. Прикладна інформатика. Підручник для студентів комп'ютерних спеціальностей. – Дніпропетровськ: Видавництво НГУ, 2004. – 559 с. URL: http://www.programmer.dp.ua/book-ua-k01.php. дата звернення: 15.03.2019.
10. Бусыгин Б.С., Дивизинюк М.М., Коротенко Г.М., Коротенко Л.М. Введение в современную информатику. Учебник. – Севастополь: Издательство СНУЯЭиП, 2005. – 644 с. / URL: http://www.programmer.dp.ua/book-ru-k02.php. дата звернення: 15.01.2021.

11. Документація. Звіти у сфері науки і техніки. Структура і правила оформлення : ДСТУ 3008-95. – Чинний від 1996–01–01. – К. : Держстандарт України, 1996. – 39 с.

12. ДСТУ 2394-94 Інформація та документація. Комплектування фонду, бібліографічний опис, аналіз документів. Терміни та визначення. – Чинний від 01.01.1995. - Київ : Держстандарт України, 1994. – 88 с.

13. Методичні вказівки з виконання економічного розділу в дипломних проектах студентів спеціальності “Комп’ютерні системи ” / Укладачі О.Г. Вагонова, Нікітіна О.Б. Н.Н. Романюк – Дніпропетровськ: Національний гірничий університет. – 2013. – 23с.

14. Методичні рекомендації до виконання кваліфікаційних робіт бакалаврів напряму підготовки 122 «Комп’ютерні науки» галузі знань 12 Інформаційні технології/, Л.М. Коротенко , О.С. Шевцова; Нац. гірн. ун-т. – Д : ДВНЗ НГУ, 2019. – 65 с.

15. Методичні рекомендації щодо написання, оформлення та представлення учнівських науково-дослідницьких робіт учнів – членів Малої академії наук України / Г.Г. Півняк, Л.М. Коротенко, І.М. Удовик, Є.М. Головня – Д.: ДВНЗ «Національний гірничий університет», 2017. – 24 с.

16. Скорочення слів в українській мові у бібліографічному описі. Загальні правила та вимоги : ДСТУ 3582-97. – Чинний від 1998–07–01. – К. : Держстандарт України, 1998. – 24 с. – (Державний стандарт України).

17. Харди, Б. Программирование под Android. Для профессионалов : [пер. с англ.] / Брайан Харди, Билл Филлипс. – СПб. : Питер, 2014. – 592 с. – (Для профессионалов).

18. Цехнер, М. Программирование игр под Android / Марио Цехнер. – СПб. : Питер, 2013. – 688 с.

19. Benjamin Pierce. Types and Programming Languages. — MIT Press, 2002. – 221с.

20. Raghav Sood, Pro Android Augmented Reality. – Apress, 2012. – 343.

ДОДАТОК А

КОД ПРОГРАМИ

index.html

```
<!doctype html>

<html lang="en">

<head>

<!-- Loading Bootstrap -->

<link href="flatUI/css/bootstrap.css" rel="stylesheet">

<!-- Loading Flat UI -->

<link href="flatUI/css/flat-ui.css" rel="stylesheet">

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0, minimum-scale=1.0">


<title>Bitcoin Cold Wallet</title>

<script>

//Global vars

var addChange = false;

var amountToChange = null;

var changeNumber = null;

var currentPage = "key";

var privateKey = null;

var addressNumber = 0;
```

```
</script>

<script src="js/bigi.js"></script>

<script src="js/bitcoinjs.min.js"></script> <script src="js/qrcode.min.js"></script>
<script src="js/mylib.js"></script>

<!-- Loading material-desing-lite CSS -->

<link rel="stylesheet" href="google/google.css"> <link rel="stylesheet"
href="google/styles.css">

<link rel="stylesheet" href="css/mylib.css"> </head>

<body class="mdl-color--grey-100 mdl-color-text--grey-700 mdl-base"> <div
class="mdl-layout mdl-js-layout mdl-layout--fixed-header">

<header class="mdl-layout__header mdl-layout__header--scroll mdl-color--blue-

700">

<div id ="menu" class="mdl-layout__tab-bar mdl-js-ripple-effect mdl-color-- blue-
700">

<a href="#key" id ="keyButton" style ="width:26.5%;" class="mdl-layout__tab is-
active"onclick ="openPage('key',currentPage);">Ключ</a>

<a href="#help" id ="helpButton" style ="width:26.5%;" class="mdl-
layout__tab"onclick ="openPage('help',currentPage);">Інструкція</a>

<a href="#license" id ="licenseButton" style ="width:26.5%;" class="mdl-
layout__tab"onclick ="openPage('license',currentPage);">Ліцензії</a>

<a href="#receive" id ="receiveButton" style ="width:26.5%;" class="mdl-
layout__tab"onclick ="openPage('receive',currentPage);">Отримати</a>

<a href="#send"id ="sendButton" style ="width:26.5%;" class="mdl-
layout__tab"onclick ="openPage('send',currentPage);">Надіслати</a>

</div>

</header>

<main class="mdl-layout__content">
```

<form name ="upload">

<input type ="file" id ="fileKey" style="height: 600px; width: 100%; font-size:36; background:url(plus.svg); background-repeat:no-repeat;"></input>

<!-- Місце для інструкції -->

 <!-- Місце для ліцензій -->

</br>

<input id ="addressNumber" type ="number" value =0 onchange = "window.addressNumber=this.value;"></input>

Отримати адресу

</br> </br>

Отримати приватний ключ

</br>

```
<div class ="span3">

</br>

<input type ="number" placeholder ="номер гаманця" onchange =
"window.withdrawalNumber = this.value;">

<input type ="text" placeholder ="хеш транзакції-входу" onchange =
"window.previousTxHash = this.value;"><input type ="number" placeholder
="індекс входу в транзакції" onchange ="window.previousTxIndex =
Number(this.value);">

<input type ="text" placeholder ="біткоін адреса отримувача" onchange =
"window.depositAddress = this.value;">

<input type ="number" placeholder ="сума" min =0.00000001 max =21000000 step
=0.00000001 onchange ="window.amountToSend =
parseInt(this.value*100000000);">

<input type ="number" id ="amountToChange" min =0.00000001 max =21000000
step =0.00000001 placeholder ="сума решти" onchange
="window.amountToChange = parseInt(this.value*100000000);"></br>

<input type ="number" id ="changeNumber" placeholder ="номер гаманця для
решти" onchange ="window.changeNumber = this.value;"></br>

<a href="#" id ="changeButton" class="btn btn-large btn-block btn-warning"
onclick ="addChangeForm();">Додати решту</a>

<a href="#" id ="removeChangeButton" class="btn btn-large btn-block btn-
warning" onclick ="removeChangeForm();">Прибрати решту</a>

<a href="#" class="btn btn-large btn-block btn-info" onclick =
"buildTransactionForm();">Сформувати транзакцію</a>

<div id ="qrRawTx"></div>

</div></div>

<script src="js/keyinput.js"></script>

</main>

</div>
```

```
<!-- Loading material-desing-lite JavaScript --> <script
src="google/google.js"></script>

</body>

</html>
```

mylib.js

```
//Functions

function displayId(id,displayType = 'block'){ var id =
document.getElementById(id); id.style.display = displayType;

}

function hideId(id){

var id = document.getElementById(id);

id.style.display = 'none';

}

function recursiveHash(baseData,recursiveNumber = 0){ do{

baseData = bitcoin.crypto.sha256(baseData); recursiveNumber--;

}while(recursiveNumber >= 0);

return baseData;

}

function recursiveKeyPair(baseData,recursiveNumber = 0){ return new

bitcoin.ECPair(bigi.fromBuffer(recursiveHash(baseData,recursiveNumber)));

}

function openPage(pageName,currentPage){
```

```
hideId(currentPage);

displayId(pageName);

window.currentPage = pageName;

}


function getAddressForm(){

addressSubmit = recursiveKeyPair(fileContent,window.addressNumber);
window.currentNumber = window.addressNumber; window.currentAddress =
addressSubmit;

var addressPlace = document.getElementById('qrAddress')

addressPlace.innerHTML = '';

var div = document.createElement('div'); div.innerHTML = ('Address
#'+window.addressNumber+':

'+addressSubmit.getAddress());

addressPlace.appendChild(div);

var QRcode = new QRCode(addressPlace,'bitcoin:' + addressSubmit.getAddress());
displayId('privateKeyButton');

hideId('qrKey');

}

function getKeyForm(){

var currentPrivateKey = window.currentAddress.toWIF(); var keyPlace =
document.getElementById('qrKey') keyPlace.innerHTML = '';
var div = document.createElement('div');

div.innerHTML = ('Key #'+window.currentNumber+': '+currentPrivateKey);
keyPlace.appendChild(div);

var QRcode = new QRCode(keyPlace,currentPrivateKey); displayId('qrKey');
```

```
}

function addChangeForm(){ window.addChange = true; hideId('changeButton');
displayId('amountToChange','inline'); displayId('changeNumber','inline');
displayId('removeChangeButton');

}

function removeChangeForm(){

window.addChange = false;

hideId('removeChangeButton');

hideId('amountToChange');

hideId('changeNumber');

displayId('changeButton');

}

function buildTransactionForm(){

var rawTx = new bitcoin.TransactionBuilder();

rawTx.addInput(window.previousTxHash,window.previousTxIndex);

rawTx.addOutput(window.depositAddress,window.amountToSend);

if(window.addChange && window.amountToChange &&
window.changeNumber){

var changeAddress =
recursiveKeyPair(window.fileContent,window.changeNumber);
rawTx.addOutput(changeAddress.getAddress(),window.amountToChange);}
rawTx.sign(0,recursiveKeyPair(window.fileContent,window.withdrawalNumber));
var readyTx = rawTx.build().toHex();

var rawTxPlace = document.getElementById('qrRawTx')

rawTxPlace.innerHTML = '';

var div = document.createElement('div');
```

```
div.innerHTML = (readyTx);

rawTxPlace.appendChild(div);

var QRcode =        new QRCode(rawTxPlace,readyTx);

}
```
mylib.css

```
.blocks {

display:none;

}

#receiveButton{

display:none;

}

#sendButton{

display:none;

}

#privateKeyButton{

display:none;

}

#amountToChange{

display:none;

}

#changeNumber{

display:none;
```

```
}

#removeChangeButton{

display:none;

}

#key{

display:block;

}

#view-source {

position: fixed;

display: block;

right: 0;

bottom: 0;

margin-right: 40px;

margin-bottom: 40px;

z-index: 900;

}
@font-face {
  font-family: 'Flat-UI-Icons-24';
  src:url('fonts/Flat-UI-Icons-24.eot');
  src:url('fonts/Flat-UI-Icons-24.eot?#iefix') format('embedded-opentype'),
        url('fonts/Flat-UI-Icons-24.woff') format('woff'),
        url('fonts/Flat-UI-Icons-24.ttf') format('truetype'),
        url('fonts/Flat-UI-Icons-24.svg#Flat-UI-Icons-24') format('svg');
  font-weight: normal;
  font-style: normal;
}

/* Use the following CSS code if you want to use data attributes for inserting your
```

```
icons */
[data-icon]:before {
  font-family: 'Flat-UI-Icons-24';
  content: attr(data-icon);
  speak: none;
  font-weight: normal;
  line-height: 1;
  -webkit-font-smoothing: antialiased;
}

/* Use the following CSS code if you want to have a class per icon */
/*
Instead of a list of all class selectors,
you can use the generic selector below, but it's slower:
[class*="fui-"]:before {
*/
.fui-video-24:before, .fui-time-24:before, .fui-settings-24:before, .fui-plus-24:before,
.fui-new-24:before, .fui-menu-24:before, .fui-man-24:before, .fui-mail-24:before,
.fui-lock-24:before, .fui-location-24:before, .fui-heart-24:before, .fui-eye-24:before,
.fui-cross-24:before, .fui-cmd-24:before, .fui-checkround-24:before, .fui-
checkmark-24:before, .fui-calendar-24:before, .fui-bubble-24:before, .fui-volume-
24:before, .fui-camera-24:before {
  font-family: 'Flat-UI-Icons-24';
  speak: none;
  font-style: normal;
  font-weight: normal;
  line-height: 1;
  -webkit-font-smoothing: antialiased;
}
.fui-video-24:before {
  content: "\e000";
}
.fui-time-24:before {
  content: "\e001";
}
.fui-settings-24:before {
  content: "\e002";
}
.fui-plus-24:before {
  content: "\e003";
}
.fui-new-24:before {
  content: "\e005";
}
```

```
.fui-menu-24:before {
  content: "\e006";
}
.fui-man-24:before {
  content: "\e007";
}
.fui-mail-24:before {
  content: "\e008";
}
.fui-lock-24:before {
  content: "\e009";
}
.fui-location-24:before {
  content: "\e00a";
}
.fui-heart-24:before {
  content: "\e00b";
}
.fui-eye-24:before {
  content: "\e00c";
}
.fui-cross-24:before {
  content: "\e00d";
}
.fui-cmd-24:before {
  content: "\e00e";
}
.fui-checkround-24:before {
  content: "\e00f";
}
.fui-checkmark-24:before {
  content: "\e010";
}
.fui-calendar-24:before {
  content: "\e011";
}
.fui-bubble-24:before {
  content: "\e012";
}
.fui-volume-24:before {
  content: "\e013";
}
.fui-camera-24:before {
  content: "\e004";
```

```
}


/*! http://mths.be/placeholder v2.0.7 by @mathias */
;(function(window, document, $) {

  var isInputSupported = 'placeholder' in document.createElement('input'),
      isTextareaSupported = 'placeholder' in document.createElement('textarea'),
      prototype = $.fn,
      valHooks = $.valHooks,
      hooks,
      placeholder;

  if (isInputSupported && isTextareaSupported) {

        placeholder = prototype.placeholder = function() {
              return this;
        };

        placeholder.input = placeholder.textarea = true;

  } else {

        placeholder = prototype.placeholder = function() {
              var $this = this;
              $this
                    .filter((isInputSupported ? 'textarea' : ':input') + '[placeholder]')
                    .not('.placeholder')
                    .bind({
                          'focus.placeholder': clearPlaceholder,
                          'blur.placeholder': setPlaceholder
                    })
                    .data('placeholder-enabled', true)
                    .trigger('blur.placeholder');
              return $this;
        };

        placeholder.input = isInputSupported;
        placeholder.textarea = isTextareaSupported;

        hooks = {
              'get': function(element) {
                    var $element = $(element);
                    return $element.data('placeholder-enabled') &&
```

```
$element.hasClass('placeholder') ? '' : element.value;
		},
		'set': function(element, value) {
			var $element = $(element);
			if (!$element.data('placeholder-enabled')) {
				return element.value = value;
			}
			if (value == '') {
				element.value = value;
				// Issue #56: Setting the placeholder causes problems if the element continues to have focus.
				if (element != document.activeElement) {
					// We can't use `triggerHandler` here because of dummy text/password inputs :(
					setPlaceholder.call(element);
				}
			} else if ($element.hasClass('placeholder')) {
				clearPlaceholder.call(element, true, value) || (element.value = value);
			} else {
				element.value = value;
			}
			// `set` can not return `undefined`; see http://jsapi.info/jquery/1.7.1/val#L2363
			return $element;
		}
	};

	isInputSupported || (valHooks.input = hooks);
	isTextareaSupported || (valHooks.textarea = hooks);

	$(function() {
		// Look for forms
		$(document).delegate('form', 'submit.placeholder', function() {
			// Clear the placeholder values so they don't get submitted
			var $inputs = $('.placeholder', this).each(clearPlaceholder);
			setTimeout(function() {
				$inputs.each(setPlaceholder);
			}, 10);
		});
	});

	// Clear placeholder values upon page reload
	$(window).bind('beforeunload.placeholder', function() {
```

```
            $('.placeholder').each(function() {
                this.value = '';
            });
        });

    }

    function args(elem) {
        // Return an object of element attributes
        var newAttrs = {},
           rinlinejQuery = /^jQuery\d+$/;
        $.each(elem.attributes, function(i, attr) {
            if (attr.specified && !rinlinejQuery.test(attr.name)) {
                newAttrs[attr.name] = attr.value;
            }
        });
        return newAttrs;
    }

    function clearPlaceholder(event, value) {
        var input = this,
           $input = $(input);
        if (input.value == $input.attr('placeholder') &&
$input.hasClass('placeholder')) {
            if ($input.data('placeholder-password')) {
                $input = $input.hide().next().show().attr('id',
$input.removeAttr('id').data('placeholder-id'));
                // If `clearPlaceholder` was called from `$.valHooks.input.set`
                if (event === true) {
                    return $input[0].value = value;
                }
                $input.focus();
            } else {
                input.value = '';
                $input.removeClass('placeholder');
                input == document.activeElement && input.select();
            }
        }
    }

    function setPlaceholder() {
        var $replacement,
           input = this,
           $input = $(input),
```

```
            $origInput = $input,
            id = this.id;
        if (input.value == '') {
                if (input.type == 'password') {
                        if (!$input.data('placeholder-textinput')) {
                                try {
                                        $replacement = $input.clone().attr({ 'type': 'text' });
                                } catch(e) {
                                        $replacement = $('<input>').attr($.extend(args(this),
{ 'type': 'text' }));
                                }
                                $replacement
                                        .removeAttr('name')
                                        .data({
                                                'placeholder-password': true,
                                                'placeholder-id': id
                                        })
                                        .bind('focus.placeholder', clearPlaceholder);
                                $input
                                        .data({
                                                'placeholder-textinput': $replacement,
                                                'placeholder-id': id
                                        })
                                        .before($replacement);
                        }
                        $input = $input.removeAttr('id').hide().prev().attr('id', id).show();
                        // Note: `$input[0] != input` now!
                }
                $input.addClass('placeholder');
                $input[0].value = $input.attr('placeholder');
        } else {
                $input.removeClass('placeholder');
        }
  }

}(this, document, jQuery));

/*
============================================================
 * bootstrap-tooltip.js v2.3.0
 * http://twitter.github.com/bootstrap/javascript.html#tooltips
 * Inspired by the original jQuery.tipsy by Jason Frame
 *
============================================================
```

```
 * Copyright 2012 Twitter, Inc.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 * http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express
or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 * ==========================================================
*/


!function ($) {

  "use strict"; // jshint ;_;


 /* TOOLTIP PUBLIC CLASS DEFINITION
  * ============================== */

  var Tooltip = function (element, options) {
    this.init('tooltip', element, options)
  }

  Tooltip.prototype = {

    constructor: Tooltip

  , init: function (type, element, options) {
      var eventIn
        , eventOut
        , triggers
        , trigger
        , i

      this.type = type
      this.$element = $(element)
      this.options = this.getOptions(options)
```

```
    this.enabled = true

    triggers = this.options.trigger.split(' ')

    for (i = triggers.length; i--;) {
      trigger = triggers[i]
      if (trigger == 'click') {
        this.$element.on('click.' + this.type, this.options.selector, $.proxy(this.toggle,
this))
      } else if (trigger != 'manual') {
        eventIn = trigger == 'hover' ? 'mouseenter' : 'focus'
        eventOut = trigger == 'hover' ? 'mouseleave' : 'blur'
        this.$element.on(eventIn + '.' + this.type, this.options.selector,
$.proxy(this.enter, this))
        this.$element.on(eventOut + '.' + this.type, this.options.selector,
$.proxy(this.leave, this))
      }
    }

    this.options.selector ?
      (this._options = $.extend({}, this.options, { trigger: 'manual', selector: '' })) :
      this.fixTitle()
  }

  , getOptions: function (options) {
    options = $.extend({}, $.fn[this.type].defaults, this.$element.data(), options)

    if (options.delay && typeof options.delay == 'number') {
      options.delay = {
        show: options.delay
      , hide: options.delay
      }
    }

    return options
  }

  , enter: function (e) {
    var self = $(e.currentTarget)[this.type](this._options).data(this.type)

    if (!self.options.delay || !self.options.delay.show) return self.show()

    clearTimeout(this.timeout)
    self.hoverState = 'in'
```

```
    this.timeout = setTimeout(function() {
      if (self.hoverState == 'in') self.show()
    }, self.options.delay.show)
  }

, leave: function (e) {
    var self = $(e.currentTarget)[this.type](this._options).data(this.type)

    if (this.timeout) clearTimeout(this.timeout)
    if (!self.options.delay || !self.options.delay.hide) return self.hide()

    self.hoverState = 'out'
    this.timeout = setTimeout(function() {
      if (self.hoverState == 'out') self.hide()
    }, self.options.delay.hide)
  }

, show: function () {
    var $tip
      , pos
      , actualWidth
      , actualHeight
      , placement
      , tp
      , e = $.Event('show')

    if (this.hasContent() && this.enabled) {
      this.$element.trigger(e)
      if (e.isDefaultPrevented()) return
      $tip = this.tip()
      this.setContent()

      if (this.options.animation) {
        $tip.addClass('fade')
      }

      placement = typeof this.options.placement == 'function' ?
        this.options.placement.call(this, $tip[0], this.$element[0]) :
        this.options.placement

      $tip
        .detach()
        .css({ top: 0, left: 0, display: 'block' })
```

```
      this.options.container ? $tip.appendTo(this.options.container) :
$tip.insertAfter(this.$element)

      pos = this.getPosition()

      actualWidth = $tip[0].offsetWidth
      actualHeight = $tip[0].offsetHeight

      switch (placement) {
        case 'bottom':
          tp = {top: pos.top + pos.height, left: pos.left + pos.width / 2 - actualWidth /
2}
          break
        case 'top':
          tp = {top: pos.top - actualHeight, left: pos.left + pos.width / 2 - actualWidth /
2}
          break
        case 'left':
          tp = {top: pos.top + pos.height / 2 - actualHeight / 2, left: pos.left -
actualWidth}
          break
        case 'right':
          tp = {top: pos.top + pos.height / 2 - actualHeight / 2, left: pos.left +
pos.width}
          break
      }

      this.applyPlacement(tp, placement)
      this.$element.trigger('shown')
    }
  }

 , applyPlacement: function(offset, placement){
    var $tip = this.tip()
      , width = $tip[0].offsetWidth
      , height = $tip[0].offsetHeight
      , actualWidth
      , actualHeight
      , delta
      , replace

    $tip
      .offset(offset)
      .addClass(placement)
```

```
      .addClass('in')

    actualWidth = $tip[0].offsetWidth
    actualHeight = $tip[0].offsetHeight

    if (placement == 'top' && actualHeight != height) {
      offset.top = offset.top + height - actualHeight
      replace = true
    }

    if (placement == 'bottom' || placement == 'top') {
      delta = 0

      if (offset.left < 0){
        delta = offset.left * -2
        offset.left = 0
        $tip.offset(offset)
        actualWidth = $tip[0].offsetWidth
        actualHeight = $tip[0].offsetHeight
      }

      this.replaceArrow(delta - width + actualWidth, actualWidth, 'left')
    } else {
      this.replaceArrow(actualHeight - height, actualHeight, 'top')
    }

    if (replace) $tip.offset(offset)
  }

, replaceArrow: function(delta, dimension, position){
    this
      .arrow()
      .css(position, delta ? (50 * (1 - delta / dimension) + "%") : '')
  }

, setContent: function () {
    var $tip = this.tip()
      , title = this.getTitle()

    $tip.find('.tooltip-inner')[this.options.html ? 'html' : 'text'](title)
    $tip.removeClass('fade in top bottom left right')
  }

, hide: function () {
```

```
    var that = this
      , $tip = this.tip()
      , e = $.Event('hide')

    this.$element.trigger(e)
    if (e.isDefaultPrevented()) return

    $tip.removeClass('in')

    function removeWithAnimation() {
      var timeout = setTimeout(function () {
        $tip.off($.support.transition.end).detach()
      }, 500)

      $tip.one($.support.transition.end, function () {
        clearTimeout(timeout)
        $tip.detach()
      })
    }

    $.support.transition && this.$tip.hasClass('fade') ?
      removeWithAnimation() :
      $tip.detach()

    this.$element.trigger('hidden')

    return this
  }

, fixTitle: function () {
    var $e = this.$element
    if ($e.attr('title') || typeof($e.attr('data-original-title')) != 'string') {
      $e.attr('data-original-title', $e.attr('title') || '').attr('title', '')
    }
  }

, hasContent: function () {
    return this.getTitle()
  }

, getPosition: function () {
    var el = this.$element[0]
    return $.extend({}, (typeof el.getBoundingClientRect == 'function') ?
el.getBoundingClientRect() : {
```

```
      width: el.offsetWidth
    , height: el.offsetHeight
    }, this.$element.offset())
  }

, getTitle: function () {
    var title
      , $e = this.$element
      , o = this.options

    title = $e.attr('data-original-title')
      || (typeof o.title == 'function' ? o.title.call($e[0]) :  o.title)

    return title
  }

, tip: function () {
    return this.$tip = this.$tip || $(this.options.template)
  }

, arrow: function(){
    return this.$arrow = this.$arrow || this.tip().find(".tooltip-arrow")
  }

, validate: function () {
    if (!this.$element[0].parentNode) {
      this.hide()
      this.$element = null
      this.options = null
    }
  }

, enable: function () {
    this.enabled = true
  }

, disable: function () {
    this.enabled = false
  }

, toggleEnabled: function () {
    this.enabled = !this.enabled
  }
```

```
  , toggle: function (e) {
      var self = e ? $(e.currentTarget)[this.type](this._options).data(this.type) : this
      self.tip().hasClass('in') ? self.hide() : self.show()
    }

  , destroy: function () {
      this.hide().$element.off('.' + this.type).removeData(this.type)
    }

  }


 /* TOOLTIP PLUGIN DEFINITION
  * ========================== */

  var old = $.fn.tooltip

  $.fn.tooltip = function ( option ) {
    return this.each(function () {
      var $this = $(this)
        , data = $this.data('tooltip')
        , options = typeof option == 'object' && option
      if (!data) $this.data('tooltip', (data = new Tooltip(this, options)))
      if (typeof option == 'string') data[option]()
    })
  }

  $.fn.tooltip.Constructor = Tooltip

  $.fn.tooltip.defaults = {
    animation: true
  , placement: 'top'
  , selector: false
  , template: '<div class="tooltip"><div class="tooltip-arrow"></div><div
class="tooltip-inner"></div></div>'
  , trigger: 'hover focus'
  , title: ''
  , delay: 0
  , html: false
  , container: false
  }


 /* TOOLTIP NO CONFLICT
```

```
  * ========================== */

  $.fn.tooltip.noConflict = function () {
    $.fn.tooltip = old
    return this
  }

}(window.jQuery);
```

ДОДАТОК Б

ВІДГУК КЕРІВНИКА ЕКОНОМІЧНОГО РОЗДІЛУ

ДОДАТОК В

ПЕРЕЛІК ФАЙЛІВ НА ДИСКУ

Ім’я файлу	**Опис**
Пояснювальні документи	
Диплом_ doc	Пояснювальна записка до кваліфікаційної роботи. Документ Word.
Диплом_.pdf	Пояснювальна записка до кваліфікаційної роботи в форматі PDF
Програма	
Program.rar	Архів. Містить коди програми і откомпільовану програму
Презентація	
Презентація_.ppt	Презентація кваліфікаційної роботи