МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет імені В.Н. Каразіна

Факультет: **ННІ Каразінський банківський інститут**
Кафедра: **Інформаційних технологій та математичного моделювання**
Спеціальність: **122 Комп'ютерні науки**
Освітня програма: **Комп'ютерні науки та інформаційні технології в бізнесі**

Група: АК-41б **денна форма навчання**

КВАЛІФІКАЦІЙНА БАКАЛАВРСЬКА РОБОТА

на тему:

«РОЗРОБКА МОБІЛЬНОГО ДОДАТКА ДЛЯ УПРАВЛІННЯ ФІНАНСОВИМИ ЗАОЩАДЖЕННЯМИ З ФУНКЦІЄЮ РОЗРАХУНКУ СКЛАДНИХ ВІДСОТКІВ»
ЗА НАКАЗОМ № 4601-5/335 ВІД 07 ЛЮТОГО 2025 РОКУ

здобувача вищої освіти **Житушкіної Вікторії Андріївни**

Робота допущена до захисту в ЕК
протокол кафедри ІТММ №__ від _____2025р.

Завідувач кафедри ІТММ
к.п.н., доцент
_____ Н.І. Стяглик

Науковий керівник
к.п.н., доцент
_____ Н.І. Стяглик

м. Харків 2025 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Харківський національний університет імені В. Н. Каразіна

Факультет навчально-науковий інститут "Каразінський банківський інститут"
Кафедра інформаційних технологій та математичного моделювання
Рівень вищої освіти перший (бакалаврський)
Спеціальність 122 Комп'ютерні науки
Освітня програма Комп'ютерні науки та інформаційні технології в бізнесі

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ **Н. І. Стяглик**
Підпис ініціали, прізвище

"08" лютого 2025 року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ (ПРОЄКТ)

ЖИТУШКІНА Вікторія Андріївна
(прізвище, ім'я, по батькові студента)

1. Тема роботи «Розробка мобільного додатка для управління фінансовими заощадженнями з функцією розрахунку складних відсотків»

керівник роботи Стяглик Наталя Іванівна к.п.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від "08" лютого 2025 року № 4601-5/335

2. Строк подання студентом роботи 15 травня 2025 року

3. Перелік питань, які потрібно розробити:
У розділі 1: розглянути теоретичне питання управління фінансовими заощадженнями; обґрунтувати потреби створення мобільного застосунку для фінансового планування з функцією розрахунку складних відсотків.

У розділі 2: виконати проектування та технічну реалізацію мобільного застосунку.

У розділі 3: здійснити аналіз результат та оцінити якість розробленого застосунку.

4. План роботи

№ з/п	Назви етапів роботи
1	Вибір здобувачем теми кваліфікаційної бакалаврської роботи
2	Затвердження плану і завдання кваліфікаційної бакалаврської роботи
3	Здача кваліфікаційної бакалаврської роботи керівнику
4	Підпис кваліфікаційної бакалаврської роботи керівника
5	Підпис кваліфікаційної бакалаврської роботи у нормоконтролера
6	Допуск завідувачем кафедри до захисту кваліфікаційної бакалаврської роботи
7	Захист кваліфікаційної бакалаврської роботи

5. Дата видачі завдання 08 лютого 2025 року

Студент _____ **В.А. Житушкіна**
підпис ініціали, прізвище

Керівник роботи _____ **Н.І. Стяглик**
підпис ініціали, прізвище

РЕФЕРАТ
НА КВАЛІФІКАЦІЙНУ БАКАЛАВРСЬКУ РОБОТУ
«Розробка мобільного додатка для управління фінансовими заощадженнями з функцією розрахунку складних відсотків»
Житушкіної Вікторії Андріївни

Кваліфікаційна бакалаврська робота містить 52 сторінок, 0 таблиць, 8 рисунків, список літератури з 14 найменувань.

Об'єктом дослідження є управління фінансовими заощадженнями.

Предметом дослідження є відслідковування надходжень і витрат з можливістю розрахунку інвестицій.

Мета кваліфікаційної бакалаврської роботи полягає у створенні мобільного застосунку для відслідковування надходжень і витрат з можливістю розрахунку інвестицій.

Завданнями кваліфікаційної бакалаврської роботи є:

У розділі 1: розглянути теоретичне питання управління фінансовими заощадженнями; обґрунтувати потреби створення мобільного застосунку для фінансового планування з функцією розрахунку складних відсотків.

У розділі 2: виконати проектування та технічну реалізацію мобільного застосунку.

У розділі 3: здійснити аналіз результат та оцінити якість розробленого застосунку.

Актуальність дослідження:

Необхідність у інструментах, котрі можуть допомогти в покращенні рівня фінансової грамотності населення.

За результатами дослідження:

Створено застосунок з функціями запису витрат та надходжень та калькуляторами для обчислення складних відсотків.

Одержані результати можуть бути використані:

Для початкового ознайомлення з фінансовими інструментами, мовами програмування, фреймоворками та середовищами розробки, котрі можна використати для створення фінансового мобільного додатку.

КЛЮЧОВІ СЛОВА:

ІНСТРУМЕНТИ ФІНАНСОВОГО ОБЛІКУ, МОБІЛЬНІ ДОДАТКИ, UNITY, FIGMA, UI

ABSTRACT

AT QUALIFICATION BACHELOR WORK

"Development of a mobile application for managing financial savings with the function of calculating compound interest"

Zhytushkina Victoria Andriivna

The bachelor's thesis contains 52 pages, 0 tables, 8 drawings, a list of references of 14 titles.

The object of research is financial accounting tools.

The subject of research is the creation of a financial application.

The purpose of the bachelor's thesis is to create an application for tracking income and expenses with the ability to calculate investments.

The tasks of the bachelor thesis are:

Chapter 1: justification of the need for a mobile application for financial planning.

Chapter 2: design and technical implementation of the mobile application.

In chapter 3: analyzing the result and evaluating the quality of the application.

Relevance of the study:

The need for tools that can help improve the level of financial literacy of the population.

Research results:

An application with the functions of recording expenses and income and calculators for calculating interest has been created.

The results can be used:

For an initial introduction to financial instruments, programming languages, frameworks and development environments that can be used to create a financial mobile application.

KEYWORDS:

FINANCIAL ACCOUNTING TOOLS, MOBILE APPLICATIONS, UNITY, FIGMA, UI

ЗМІСТ

ВСТУП

Гроші є однією з основ сучасного суспільства, вони відкривають велику кількість дверей та можливостей. Більшість людей дуже мають за ціль заробити великий статок для того, щоб покращити своє становище, але чому не у всіх це вдається зробити, що відрізняє багатих людей від більшості.

Старше покоління може сказати, що вони все це отримали не чесно, ті молодіші можуть сказати, що це все їх батьки бо вони теж багаті, так інколи це може впливати, але основною відмінністю насправді успішних людей є одна навичка. Цією навичкою є фінансова грамотність, бо недаремно кажуть, що якщо всім дати однакову кількість грошей, через рік все повернеться до того стану, який був перед цим. Раніше, людям можна було спустити з рук те, що вони не можуть добре керувати своїм бюджетом, освіта була рідкістю, інструментів керування фінансами було мало. Але з розвитком суспільства це повинно було змінитися, освіта пішла в маси, фізичні інструменти для керування стали доступніші, потім з’явилися цифрові, що зробило поріг входження для користування ще нижчим. Маючи всі ці покращення ситуація не дуже змінилися. Тому все, що пов’язано з фінансовою грамотністю населення є дуже важливим. Саме ці фактори роблять актуальною тему «Розробка мобільного додатка для управління фінансовими заощадженнями з функцією розрахунку складних відсотків»

Для того, щоб зробити більш цікавим та зручним роботу з фінансами були створенні інструменти, у наш час вони мають найчастіше цифровий вигляд, для того, щоб людина мала до них доступ у будь-який час. Тому створення мобільного додатку, який буде використовуватися різними людьми, може впливати на рівень освіти у цій сфері у позитивний бік.

Об’єктом дослідження є управління фінансовими заощадженнями

Предметом є відслідковування надходжень і витрат з можливістю розрахунку інвестицій.

Метою роботи є створення додатку, який можна використовувати у повсякденному житті, для відслідковування надходжень і витрат з усіх банківських рахунків та джерел, з послідуючою можливістю розрахунку прибутку від використання отриманих грошей, як інвестиційний фонд.

Робота буде цікава широкому колу користувачів, які прагнуть ефективно керувати особистими фінансами — від студентів і фрілансерів до підприємців та інвесторів. Особливо актуальною вона стане для тих, хто має декілька банківських рахунків або джерел доходу та бажає мати зручний інструмент для контролю надходжень, витрат і прибутковості власних інвестицій. Мобільний застосунок поєднає щоденну зручність з елементами фінансового планування, що зробить його корисним як у побуті, так і для прийняття стратегічних фінансових рішень.

РОЗДІЛ 1 ТЕОРЕТИЧНІ ЗАСАДИ ТА ОБҐРУНТУВАННЯ ПОТРЕБИ У МОБІЛЬНОМУ ЗАСТОСУНКУ ДЛЯ ФІНАНСОВОГО ПЛАНУВАННЯ

З розвитком суспільства змінювалися умови для успішного життя та виживання людини. У минулому людині необхідна була проста кооперація з іншими людьми, для розподілу обов’язків, що покращило якість життя. На відміну від цього у сучасному світі все більше починає залежати від самої людини, з переходом людства до капіталістичного устрою, рівень життя людини почав залежати від її вміння розпоряджатися своїми ресурсами, особливо це стосується грошей, як основної обмінної одиниці. Тому можна сказати, що у сьогоденні, можливості людини залежать на пряму від її фінансової грамотності. Але чи все добре з цим у населення нашої країни?

Згідно з дослідженням фінансової грамотності підлітків, що було проведене НБУ у квітні 2024[1], у якому взяли участь 3139 школярів, середня кількість правильних відповідей становила 46,6%, що в переводі на 12-бальну шкалу становить 3 балу. З цього можна зробити висновок, що рівень навичок користування та заощадження статків у молоді, що є майбутнім нашої країни дуже низький, тому для того, щоб покращити цю ситуацію, треба вносити конкретні зміни. У зв’язку з тим, що сучасна молодь виросла у добу цифровізації всього від математичних розрахунків до електронного розкладу дня, необхідно грати за правилами цього тренду та переносити обробку фінансових заощаджень з записників в сучасні пристрої, створюючи цифрові додатки для моніторингу та обробки власних фінансів. Розробка додатків, які можуть допомогти у керуванні заощадженнями є однією з найбільш перспективною сферою у фінансовому секторі інформаційних технологій.

1.1 Поняття фінансових заощаджень.

Перш за все необхідно визначитися, що саме мається на увазі під поняттям: «Фінансові заощадження», які в них цілі, та види, бо саме для роботи з ними найчастіше створюють фінансові додатки. Згідно з визначенням наданим Національним Банком України у рамках презентації «Основи фінансового планування та заощадження коштів»[2], що була підготовлена Центром фінансових знань НБУ «Талан», заощадження – це частина грошових доходів, яку людина відкладає для задоволення якихось конкретних потреб у майбутньому – для досягнення своїх фінансових цілей. З цього твердження можна зробити висновок, що фінансові заощадження це певний профіцит, отриманий в деякому проміжку часу, що може бути витрачений на отримання певної послуги або товару, що людина не може собі дозволити, в короткостроковій перспективі.

1.2 Основні цілі заощаджень.

Далі необхідно визначити, які основні цілі існують. Їх можна поділити на 3 категорій, а саме

- Резервний фонд
- Фінансове планування
- Інвестування

Розглянемо їх детальніше для того, щоб пояснити, що саме мається на увазі під цими категоріями

Почнемо з резервного фонду, до нього входять фінанси, що використовуються тільки у випадку непередбачуваних ситуацій, таких як: хвороба, звільнення, сімейні проблеми тощо. Бажаний обсяг коштів, які до нього входять – це сума, що може покрити собою півроку ваших витрат.

Фінансове планування, до нього входять заплановані витрати на покриття якоїсь послуги або купівлі. Найчастіше поділяються на:

короткострокові (купівля побутової техніки, подарунки), середньострокові (відпустка, ремонт, внесок кредиту), довгострокові (купівля житла, транспорту освіта).

Інвестування, до нього входять кошти, які мають на меті примножити капітал. Прикладами інструментів інвестування є: депозити в банках, військові ОВДП.

1.3 Методи управління заощадженнями

Повертаючись до основної теми дипломної роботи, а саме до розробки додатка для управління фінансовими заощадженнями, необхідно дізнатися які основні методи використовуються для управління капіталом. Методи поділяються на дві категорії:

- Класичні
- Цифрові

Класичні методи – це методи, які виконуються за допомогою ручних записів та зберігаються на фізичних накопичувачах, наприклад, зошити, журнали, фінансові щоденники.

Цифрові методи – виконуються за допомогою спеціальних цифрових застосунків та фінансових моделей, які підтримують автоматичне обчислення та заповнення. Дані, які обробленні цим методом зберігаються локально на комп’ютері або на хмарному середовищі. Прикладами інструментів, що використовуються тут є: Excel, Google Sheets, Monefy.

У наш час цифрові методи активно замінюють класичні, тому необхідно детальніше розібратися з чим це пов’язано, тим паче застосунок створений у рамках цієї роботи відноситься до саме цього методу. Основними перевагами над класичним методом є:

- Автоматизація обчислень, що зменшує ризик помилки
- Зручність використання та передачі інформації

- Можливість візуалізації інформації, у багатьох додатках та застосунках є можливість створити графік на основі наведених даних.

Класифікація мобільних додатків та розгляд їх функціоналу

Наступним кроком є огляд існуючих мобільних додатків, що використовуються для фінансового планування або частково мають такий функціонал. Проте для кращої структуризації інформації буде краще спочатку поділити їх на класи по їх меті.

Класифікація додатків

Отже, мобільні додатки для фінансового планування можна розділити, за їх функціоналом на наступні групи:

- Трекери витрат
- Інвестиційні трекери
- Менеджери бюджету
- Калькулятори заощаджень
- Банківські додатки

Давайте розглянемо кожен з них окремо

Трекери витрат – це додатки, які дозволяють відслідковувати щоденні витрати. З допомогою цих інструментів можна виконувати наступне: класифікувати витрати, встановлювати ліміти, отримати інформацію по витратам на певний період. Також деякі з цих трекерів витрат можна від’єднати до банківських рахунків для автоматичного отримання інформації про ваші фінансові дії. Приклади: Expensify, PocketGuard.

Інвестиційні трекери – використовуються для моніторингу та аналізу і моніторингу акцій, облігацій, криптовалют тощо. Цей тип має вбудований доступ до багатьох фондів, що робить його зручним інструментом для людей, які мають певний досвід у цій сфері. Приклад: SigFig.

Менеджери бюджету – використовуються для реалізації бюджетного плану, відстежування доходів та витрат за бажаними категоріями. Функціонал вбудований в такий тип додатків часто має наступний можливості:

- Створення бюджету
- Планування фінансових цілей
- Прогнозування фінансового стану

Прикладами додатків такого типу: Mint, You Need a Budget.

Калькулятори заощаджень – використовуються для планування і прогнозування заощаджень та накопичень. У своєму арсеналі має: декілька формул серед яких ті, що враховують складні відсотки, що дозволяє використовувати ці додатки для обчислення прибутку, наприклад, від повторюваних інвестицій. Прикладами таких калькуляторів є: Compound Interest Calculator та Savings Goals.

Банківські додатки – це додатки, що використовуються для управління рахунками та здійснення фінансових операцій, але вони також мають функціонал, який можна використовувати, як інструмент для фінансового планування. До цих функцій відносяться:

- Моніторинг втрат
- Налаштування бюджету на певні витрати
- Створення накопичувальних рахунків для певних цілей
- Отримання фінансової аналітики
- Інвестиції, наприклад, купівля військових ОВДП

Прикладами таких додатків є: Privat24, monobank.

Отже, класифікувавши додатки для фінансового заощадження, можна віднести додаток, який створений у межах цієї дипломної роботи до одного з них. Найближчим типом до мого проекту є калькулятори заощаджень, бо основним функціоналом мого додатку є робота з складними відсотками. Тому розглянемо детальніше, що це таке, коли використовується, та порівняємо його з простим відсотком.

1.4 Складні відсотки

Детальнішого розгляду складних відсотків треба почати з основного, що таке складні відсотки. Складний відсоток – це метод нарахування відсотків, при якому відсотки з n-ним періодом нараховуються на загальну суму і на наступний період відсотки отримуються від збільшеної суми.

$$A = P\times\left(1 + \frac{n}{r}\right)^{nt} \quad (1)$$

Формула складних відсотків

A – кінцева сума (з урахуванням відсотків),

P – початкова сума інвестиції (principal),

r – річна процентна ставка (у вигляді десяткового дробу),

n – кількість нарахувань відсотків на рік,

t – кількість років.

Складні відсотки найчастіше використовують для обчислення довгострокових заощаджень. Це пов’язано із збільш сильнішим проявом ефекту складного відсотку протягом великого проміжку часу

Порівняння простого та складного відсотків

Для порівняння необхідно записати формулу простого відсотка

$$A = P(1 + rt) \quad (2)$$

Формула простого відсотка

A – кінцева сума (з урахуванням відсотків),

P – початкова сума інвестиції (principal),

r – річна процентна ставка (у вигляді десяткового дробу),

t – кількість років.

Для порівняння возьмимо за початкову суму інвестиції 10 000 грн, відсоток – 10%, термін 10 років, для складних відсотків кількість нарахувань за рік дорівнює 3.

Обчислення простих відсотків.

$$10000(1 + 0.1 \times 10) = 20000 \quad (3)$$

Обчислення складних відсотків

$$10000 \times (1 + 0.1/3)^{3 \times 10} = 26743 \quad (4)$$

З отриманих результатів обчислень можна побачити, що кількість коштів отриманих з відсотків завдяки складним відсоткам більше за прості на 6 743, що є майже в 1.7 разів більше.

1.5 Сфера застосування та цільова аудиторія фінансових додатків

Коли розібрали основні теоретичні питання, що можуть виникнути при створенні фінансових додатків, залишилося остання, але не по важливості, а саме зрозуміти, де можна використовувати ці додатки та яка верства населення зацікавлена у них

Почнемо з сфери застосування. Основними секторами для використання є:

- Побутове використання
- Освітня діяльність
- В підприємництві

Розглянемо детальніше кожен з них

Побутове використання

У побуті може використовуватися для контролю над домашнім бюджетом, відслідковування витрат на покупки, відкладання коштів на

довгострокові цілі. Завдяки цьому середньостатистичні користувачі можуть покращити свої навички фінансової дисципліни.

Освітня підготовка

Частково цієї теми торкнулися у попередньому абзаці, але тепер розглянемо її більш детально. Фінансові додатки можна використовувати, як інструмент навчання дітей та молоді, таким навичкам:

- Основам управління грошима
- Інвестуванню на прикладі готових додатків та платформ.

Так само ці інструменти можна використовувати для більш поглибленого ознайомлення з ними для спеціальностей, які напряму пов’язані з фінансовим сектором.

Підприємництво та робота

Ці інструменти стануть у нагоді для фінансового обліку підприємцям та бухгалтерам. Вони можуть їх використовувати для

- Планування грошових потоків
- Розрахунок доходу/розходів
- Прогнозування довгострокових витрат/ доходів підприємства
- Накопичення коштів для інвестицій, великих покупок.

Коли ми розібралися, щодо сфер, можна перейти до опису цільової верстви населення цих додатків. У зв’язку з тим, що кожен додаток має свою цільову категорія будуть описуватися узагальнено усі на кого націлені ці додатки.

Отже користувачами, на яких націлені фінансові додатки є:

- Фінансові спеціалісти
- Молодь зацікавлена своїм майбутнім
- Люди, які керують сімейним бюджетом
- Викладачі фінансових установ
- Люди з нестабільним заробітком
- Люди передпенсійного віку
- Інвестори

Висновок до розділу 1.

Підсумовуючи усю теорію описану у першому розділі, можна сказати, що у наш час є велика необхідність у знаннях про те, як треба розпоряджатися фінансами, що постає більш гостро у зв’язку з низьким рівнем фінансової грамотності. На допомогу у вирішенні цього питання можна використати додатки для фінансового планування, що є інструментом цифрового методу управління заощадженнями. Але ці додатки можуть стати у нагоді не тільки новачкам, але і професіоналам у даній сфері. Тому, вважаємо, що створення мобільних додатків, що можуть допомогти у керуванні фінансами, є актуальним завданням у наш час.

РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА ТЕХНІЧНА РЕАЛІЗАЦІЯ МОБІЛЬНОГО ЗАСТОСУНКУ

Першим, що треба зробити у ході проєктування застосунку – це визначитися із:

- Мовою програмування
- Середовищем розробки
- Середовищем створення UI
- Побудовою архітектури проєкту.

Чималу увагу треба приділити і технічній реалізації проєкту, нюансам розробки, можливим викликами тощо.

Почнемо з аналітичних питань.

2.1 Вибір мови програмування.

Пропонуємо почати саме з вибору мови програмування, по деяким причинам:

- Вибір мови визначає технологічний стек, бо вони пов’язані з фреймворками, IDE та середовищами розробки, що може вплинути на подальші етапи розробки
- Застосунки створені на деяких мовах програмування працюють лише на конкретних платформах
- З використанням деяких мов можуть виникнути труднощі в реалізації функціоналу

Почнемо з того, які мови програмування використовуються для розробки мобільних застосунків.

Програмні застосунки для пристроїв на платформі Android можна створювати такими мовами:

- Dart
- JavaScript

- C#
- Kotlin
- Java

Застосунки для пристроїв на платформі iOS можна створювати за допомогою таких мов:

- Dart
- JavaScript
- C#
- Swift

Як можна помітити деякі мови програмування повторюються на обох платформах, що робить їх кросплатформними та більш бажаними у розробці програмного продукту по причині потенційного залучення більшої кількості користувачів, це є причиною чому перевага надана цим мовам програмування, тому давайте поговоримо більш детально про кожну з них окремо

Почнемо з Dart. Згідно вікіпедії, Dart – мова програмування, яку розробляє компанія Google, позиціонуючи як мову структурованого програмування для Веб. Була створена 10 жовтня 2011 року, що робить її найновішою серед мов, про які будуть описані далі. При роботі пов’язаній зі створенням інтерфейсів використовує фреймворк Flutter. Ця мова є статичною та стврого типізованою, але також може підритмувати перехід до динамічної типізації.

Dart підтримує таку парадигму програмування, як об’єктно орієнтоване програмування, окрім цього вона має декілька особливостей, про які ми поговоримо детальніше. Головною з них є система null safety, яка гарантує те, що значення змінних не може бути null, крім випадків, коли null явно задається в коді, що допомагає уникати помилок типу null dereference, що виникають при звертанні до методу або змінної зі значенням null. Основними принципами роботи null safety є:

Всі змінні не є nullable по замовчуванню

Можливість окремо зробити змінну nullable. Це робиться шляхом додавання «?» після типу змінної

«Повна звукоізоляція» - це гарантія того, що під час виконання програми не буде виникнення null dereference.

Null safety має наступні інструменти реалізації:

- Проведення аналізу під час редагування, перевіряється інструментом «dart analyze» на наявність помилок пов’язаних з null
- Ініціалізація змінних
- Оператори ? та !. Оператор ? використовується, коли треба зробити змінну з можливістю мати значення null. Оператор ! використовується для обходу перевірки на null.
- Ключове слово late можна використовувати, як механізм для створення змінних з ініціалізацією пізніше, але з необхідністю внесення значення до використання.

Серед переваг Dart можна зазначити:

- Наявність функції Got Reload, що дозволяє бачити внесені зміни без перезапуску додатку.
- Висока продуктивність, яка досягається завдяки компіляції мови в машинний код або JavaScript, що допомагає досягти високої продуктивності
- Наявність готових віджетів і компонентів у бібліотеках.

Недоліками цієї мови є:

- Залежність від Flutter. Більшість бібліотек створюється саме для цього фреймворка, що зменшує універсальність даної мови програмування
- Підтримка лише Google, що створює проблеми за межами екосистеми Google
- Обмежена вибір бібліотек та фреймворків. Хоча зазначено у перевагах наявність готових компонентів, але у зв’язку з тим, що

ця мова є ще молодою кількість інструментів, які можуть допомогти у розробці обмежена.

Наступною мовою про яку більш детально оглянемо це JavaScript, створена у 1995 році була, як мова сценаріїв браузера, тобто її основною задачею на той час було створення взаємодій веб-сторінок, інтерактивних елементів на сайтах. Але у наш час її можна використовувати в мобільній розробці, у випадку коли необхідно додати інтеграцію або взаємодію з веб-сервісами. На допомогу в цьому приходять її фреймворки, серед яких

- React Native
- Ionic
- NativeScript

Розглянемо детальніше кожен з них.

React Native є найпопулярнішим фреймворком, що використовується для створення мобільних додатків за допомогою JavaScript. Для забезпечення зв’язку між кодом написаним цією мовою та нативними модулями використовує систему мостів, основна відмінність цієї системи - розподіл задач на дві частини або потоки, JavaScript та нативну.

JavaScript відповідає за логіку додатку, обробку даних, бізнес логіку та рендеринг інтерфейсу

Нативна частина відповідає за рендеринг UI користувача та виконання специфічних операцій платформи, таких як: доступ до камери, гіроскопу, gps.

Механізм мостів є елементом, який передає інформацію між двома потоками, виконуючи це асинхронно, завдяки чому додаток може реагувати на дії користувача без необхідності очікувати відповіді від іншого потоку.

З цього випливає головна перевага цього фреймворку - можливість використання нативних компонентів додатків

Ionic - фреймворк для розробки кросплатформених мобільних додатків, що працює на основі веб-технологій, що дозволяють портувати додатки на різні пристрої.

Ionic поднює в собі WebView та нативні модулі, що дозволяє створення додатку з використанням веб-технологій. З цього можна зазначити, що для створення додатку потрібна використовувати також мову стилю сторінок CSS та мову розмітки сторінок HTML.

Для повного розуміння принципів роботи проектів створених на цьому інтерфейсі необхідно сказати декілька слів про WebView. WebView - це компонент для відображення веб-сторінок у додатках. У своїй суті його можна порівняти з емулятором для сторінок сайту, бо у своїй основі цей Ionic є гібридним фреймворком, що запускає веб-сторінку у вигляді додатку.

Перевагою цього фреймворку є перенесення на іншу платформу. У зв’язку з тим, що цей фреймворк є гібридом, для перенесення програмного продукту на іншу платформу, єдине що треба - адаптування інтерфейсу під потреби.

Недоліком є обмежений доступ до нативних функцій. У Ionic немає прямого доступу до функцій пристрою, тобто для роботи з ними необхідно завантажувати плагіни, що робить роботу з якимись можливостями, які рідко використовуються більш складною.

Описавши всі попередні переходимо до останнього. NativeScript – фреймворк з відкритим кодом, який використовується для розробки кросплатформених мобільних додатків. Він дозволяє створювати нативний інтерфейс, як і React Native.

Головною відмінністю від його конкурента є кількість фреймворків, котрі підтримуються. До них відносяться:

- Angular
- Vue
- Svelte

У той час, як єдиним фреймворком, який працює з React Native є сам React. Це робить його більш гнучким у виборі інструментів для створення додатку. До переваг, крім наведеної вище можна ще віднести прямий доступ

до нативних API, без написання коду мовами на кшталт Java або Swift, чи без використання мостів, як у попередньому.

Основним недоліком може стати набагато менша спільнота користувачів, що може створити проблеми під час пошуку рішень або необхідних плагінів.

Останньою мовою програмування, про буде розказано – C#. Ця мова є компільованою мовою програмування, тобто перед відтворенням скрипту перетворює її в машинний код. Це робить швидше за інтерпретовані мови, які не перетворюються у машинний код, а виконується інтерпретатором.

C# має прямий доступ до вбудованих функцій пристрою, завдяки бібліотекам та API кожної платформи, що дозволяє використовувати їх функції без необхідності інтегрування додаткових плагінів, або написання коду іншою мовою.

Ця мова програмування має чітку типізацію та сувору структуру коду, що надає змогу знаходити помилки на етапі компіляції, отримати кращу продуктивність кінцевого продукту і останнє, але не менш важливе це краще читання коду, завдяки чіткій структурі.

C# є частиною екосистеми Microsoft, що надає гарантії стабільної та довготривалої підтримки з їх сторони. Це може стати причиною для вибору її для проектів, які планують оновлювати та довгий час супроводжувати.

На основі вище описаної інформації можна виокремити переваги та недоліки даної мови. До переваг можна віднести:

- Продуктивність та оптимізація
- Довготривалу підтримку
- Підтримка нативних функцій пристрою
- Зручність у створенні та реалізації логіки розрахунків, що робить цю мову корисною у створенні фінансових застосунків.

До недоліків можна віднести:

- Низька гнучкість
- Час компіляції та розмір файлів

Тепер розглянемо всі мови, котрі були проаналізовані вище та знайдемо найбільш доречну нам. Почнемо з того, що нам треба від зробити за допомогою цієї мови. Це має бути додаток, який буде працювати лише на мобільних пристроях, задачею якого буде запис даних та реалізація калькуляторів. На основі цього стислого опису буде обрана мова на якій він буде реалізований.

Почнемо з того, що він повинний працювати на тільки мобільних девайсах і за умовою не передбачена взаємодія з веб-ресурсами, що перекриває головний сенс від використання JavaScript, а саме те, що він зручний в інтегруванні з веб-сервісами.

Наступним важливий момент - наявність корисної інформації, щодо вирішення потенційних проблем та інструментів, які можна запозичити. В цьому більш кращим буде C#, який є старшим за Dart та має більшу базу користувачів, які забезпечують наповненість інструментарію.

Отже беручи до уваги це, C# було обрано мовою програмування для цієї роботи.

2.2 Вибір інструмента розробки

Коли вже була обрана мова програмування, необхідно детально оглянути середовища розробки, бо вибір найбільш влучного інструменту дуже важливий з тієї ж причини, що і вибір мови, бо від нього залежить функціонал, який можна зробити у додатку та наскільки легко це буде створюватися та з якими проблемами під час розробки можна зустрітися.

Основними інструментами розробки мобільних додатків мовою C# можна вважати:

- Xamarin
- .NET MAUI
- Unity

Для того, щоб зробити найбільш влучний вибір, потрібно розглянути їх більш ретельно для того, щоб знайти їх головні переваги та недоліки, що може зіграти вирішальне значення у виборі.

Почнемо з першим у списку - Xamarin. Xamarin – це фреймворк для розробки мобільних додатків, у своїй основі використовує платформу .NET. Працює він наступним чином, Xamarin компілює код в нативні збірки, в результатом чого є висока продуктивність та доступ до API платформ Android та iOS. Цей фреймворк має два підходи до створення додатку: Xamarin.Native та Xamarin.Forms.

У першому підході інтерфес окремий для кожної платформи, з спільною логікою.

У другому UI спільний для обох, завдяки рендерингу відповідних компонентів на кожній окремо.

Перевагами цього фреймворка є:

- Дозвіл використовувати нативні функції дивайсу.
- Підтримка бібліотек для роботи з базами даних та підтримка хмарних сервісів. Це дає змогу для простої інтеграції з базами даних і сервісами, що знаходяться в мережі.
- C# є єдиною мовою, яка використовується.

До основним недоліком є застарілість. Підтримка Xamarin припинилася в 2024 році, у зв’язку з чим він може не відповідати тим потребам ринку, які є зараз та можуть виникнути у майбутньому. Головною причиною відмови Microsoft від цього середовища вважається, потреба в специфічних налаштувань для кожної платформи окремо.

Наступним фреймворком для аналізу, є .NET MAUI від Microsoft. Він дозволяє створювати додатки на платформах: Android, iOS, Windows, macOS. Його можна вважати нащадком Xamarin.Forms, але основні покращеннями в порівнянні з попередником є:

- Спрощена архітектура проекту. У тепер розробка не вимагає наявності окремого підпроекту для кожної платформи

- Використовує новіші версії .NET, від шостої до новіших.
- Краща продуктивність. Це пов’язано з використанням нової моделі тримерів, що видаляє невикористаний код під час компіляції.

Крім наведених вище переваг, до плюсів використання .NET MAUI можна віднести:

- Виконується в Visual Studio 2022 та вище, що робить можливим використання всього функціоналу, який надається цим інтегрованим середовищем розробки.
- Можливість створювати інтерфейс через XAML. Це додає гнучкості в компонуванні, анімаціях та стилях.
- Інтеграція продуктів Microsoft

До недоліків можна віднести:

- Середовище є новим, реліз його був у 2022 році, тому є можливість виникнення багів під час роботи застосунку та сам додаток може працювати нестабільно.
- Порівняно з іншими має меншу спільноту. Це пов’язано з першим недоліком і робить складнішим пошук рішень та готових компонентів.

Тепер, коли були оглянуті два фреймворка можна перейти до єдиного середовища розробки з цього списку, а саме Unity. Unity - це кросплатформений рушій, який використовується для розробки ігор, симуляцій, хоча завдяки ньому можна розробляти також, фінансові додатки, нотатки тощо. Мовою програмування, яку найчастіше використовують є C#.

Завдяки наявності великої кількості наборів інструментів для створення додатків під кожну платформу, розробник має можливість без додаткових викликів змінювати платформу натисканням однієї кнопки. Серед цих платформ:

- Windows
- Android

- iOS
- macOS
- Linux
- PlayStation
- tvOS
- visionOS

Серед переваг Unity, як середовища розробки є:

- Можливість гнучкого налаштування UI. Це можливо завдяки Canvas-системі, що збільшує варіативність налаштування інтерфейсу у застосунку.
- Підтримка створення анімацій та наявність різного відеоконтенту.
- Наявність власної мережі ассетів. Завдяки Unity Asset Store, можна знайти готові рішення для інтерфейсу, завантажити плагіни, інструменти тощо.
- Легка інтеграція сторонніх сервісів. Unity має інструмент для інтеграції сервісів напряму з GitHub, або у випадку завантаження спеціального файлу з офіційних сайтів, який можна відкрити у самому середовищі розробки, що завантажить його одразу у проект.

При усіх цих перевагах є недоліків, а саме:

- Великий розмір файлів завантаження для мобільних пристроїв.
- Складніша адаптація інтерфейсу під різні екрани.

Тепер, коли були оглянуті вище перечислені варіанти інструментів, які можна використовувати для створення фінансового мобільного додатку необхідно визначитися, який саме буде використовуватися для розробки

Першим фреймворком від використання якого було вирішено відмовитися став Xamarin, причин на то було декілька.

По-перше, його застаріла архітектура, яка базується на рендерах окремих компонентів кожної платформи, в кінцевому випадку призводить до складнішого процесу розробки та повільнішої роботи додатку.

По-друге, хоча на даний момент цей фактор не відіграє великого значення, але оновлення для кожної платформи стають доступними пізніше порівняно з іншими інструментами.

По-третє, складне створення інтерфейсу, ще більше уповільнює процес розробки додатку.

Беручи все це до уваги можна зазначити, що інші два варіанти є кращими за Xamarin.

Коли було прийняте остаточне рішення про відмову від цього застарілого фреймворка, стало зрозуміло, що основний «двобій» буде між .NET MAUI та Unity. Для того, щоб вирішити, який варіант краще, треба порівняти їх переваги та на основі цього зробити висновок, який саме інструмент більше підходить для виконання завдання.

Почнемо з кросплатформеності. Як було зазначено у .NET MAUI виправили необхідність наявності підпроектів для кожної платформи, що робить легшим зміну її. Але все одно є деякі проблеми з цим, наприклад, для того, щоб створити проект для iOS треба мати macOS, на який треба завантажити Visual Studio for Mac або використовувати комбінацію з Visual Studio, Xcode, CLI. В той час, як для створення додатку в Unity, якщо він не має сторонніх плагінів, треба дивайс з встановленим Unity + будь яка IDE.

Продуктивність. .NET MAUI використовує модель тригерів, що покращує оптимізацію додатку. На відміну від нього Unity немає вбудованого рішення, але надає можливість його створити. Цьому випадку продуктивність в лабораторних умовах може бути гірше, але цей результат не буде набагато гіршим за той, який пропонує інструмент від Microsoft.

Інструменти. .NET MAUI має повну підтримку продуктів Microsoft, найважливішим з них є Visual Studio, який дозволяє використовувати всі його функції, серед яких NuGet та Git. В Unity це середовище розробки також підтримується, але крім нього він має свій власний AssetStore, що збільшує можливості для пошуку необхідних компонентів.

Інтерфейс. Найбільша різниця в них полягає в процесі створення інтерфейсу. У той час як MAUI використовує XAML, Unity використовує свою власну систему, як полягає в розміщенні компонентів по сцені, що надає можливість легшого створення UI з використанням кастомного інтерфейсу, у той час, як фреймворк краще справляється з адаптацією під різні екрани.

Порівнявши ці 2 інструмента, було обрано Unity. З наступних причин: у разі необхідності буде доступ до більшої кількості готових компонентів, створення під різні платформи потребує менше ресурсів, перенесення інтерфейсу виконується набагато швидше.

2.3 Середовище створення UI

Дизайн є важливою частиною застосунку, бо саме на нього першого звертають увагу користувачі і від нього залежить чи забажає він використовувати або завантажувати цей додаток.

Коли вже мова програмування та середовище розробки для фінансового додатку були обрані необхідно обрати де саме буде створюватися прототип дизайну для застосунку. Основні інструменти, які використовуються для виконання поставленого завдання:

- Figma
- Adobe XD
- Sketch

Розглянемо кожного з них детальніше.

Почнемо з Sketch – це векторний графічний редактор, який використовують для створення інтерфейсу додатків, веб-сайтів, а також підтримує роботу з дизайн-системами, які використовуються для уніфікації кінцевого дизайну та прискорення розробки шляхом створення заготовок. Єдиною платформою на якій доступний цей редактор є macOS. Інструмент розповсюджується на платній основі.

До переваг його можна віднести:

- Інтуїтивно зрозумілий інтерфейс
- Векторна графіка
- Можливість робити анімації
- Можливість зробити прототип, який можна запустити на будь-якому девайси Apple.

До його мінусів можна віднести:

- Доступність. Цей додаток підтримується лише macOS та для використання потребує підписки.

Adobe XD – інструмент для дизайну, компанія-власник якої Adobe Systems. Відмінністю від попереднього є можливість використання на двох основних десктопних платформах: Windows та macOS, що є створює більший обхват потенційних користувачів.

Перевагами цього додатку є:

- Кросплатформеність
- Інтеграція з іншими продуктами Adobe, такими як Photoshop та Illustrator
- Автоматичне оновлення компонентів завдяки дизайн-системі.

Недоліками є:

- Платна модель розповсюдження
- Менший вибір плагінів та шаблонів за конкурентів.
- Знаходиться у режимі обслуговування. Це означає, що компанія не інвестує у постійний розвиток додатку, що ставить під сумнів можливість додавання нового функціоналу.

І останнім інструментом з цієї трійки є Figma. Figma - це онлайн-платформа для створення дизайну.

Перевагами є:

- Наявність безкоштовного тарифу
- Можливість спільної роботи у реальному часі
- Наявність шаблонів
- Можливість використовувати на будь-якому пристрої

Основними незручностями з якими можна зіштовхнутися під час роботи з цим інструментом є:

- Необхідність інтернет-з’єднання
- Менший функціонал ніж в аналогах

Ознайомившись з усіма середовищами для проектування інтерфейсу необхідно обрати найкраще з запропонованих. Для цього порівняємо їх між собою по наступним критеріям:

- Функціонал
- Доступність

Почнемо порівняння з функціональності. Всі ці три інструменти працюють з векторною графікою, що дозволяє їм отримувати гарне зображення в незалежності від приближення. Основною різницею між ними є можливість підтримки плагінів. Sketch має найбільшу бібліотеку плагінів, що надає можливість більш гнучкої та специфічної роботи. Після нього йде Figma, яка також підтримує велику кількість плагінів. Найменшу бібліотеку має Adobe XD, це пов’язано з меншою підтримкою зі боку розробників та спільноти.

Порівнюючи доступність треба порівняти платформи, на яких доступні додатки та цінову політику. Починаючи з доступних платформ найкраще себе показує Figma, як веб-інструмент, що дозволяє використовувати її на більшості платформ. З приводу ціни, по переду так само Figma, бо це єдине середовище створення дизайну з запропонованих, яке більшість свого функціоналу надає безкоштовно.

Беручи все це до уваги було обрано саме Figma, як інструмент, у якому буде розроблятися дизайн додатку.

2.4 Побудова архітектури проекту

Останнім пунктом теоретичної частини другого розділу є побудова архітектури проекту. Розробка архітектури проекту є ключовою частиною у підготовці до створення застосунку тому, що вона задає структуру, логіку, порядок взаємодії компонентів. Основними компонентами архітектури мого проекту є:

- Інтерфейс користувача
- Логіка взаємодії
- Логіка фінансових алгоритмів
- Збереження даних

Тепер опишемо більш детальніше кожен з них.

Інтерфейс користувача. Його завданням є взаємодія з користувачем, показ та можливість введення даних. Компонентами цього рівня є:

- Canvas, він необхідний для показу UI елементів.
- Інтерактивні елементи інтерфейсу, до них будуть відноситися такі елементи, як кнопки, текстові поля, слайдери тощо. Тобто всі елементи після взаємодії з якими буде виконуватися якась логіка додатку.

Логіка взаємодії. Завдання: обробка введеної інформації з подальшою передачею її до логіки фінансових алгоритмів або для подальшого збереження.

Логіка фінансових обчислень буде виконувати реалізацію фінансових розрахунків, які будуть представленні для вибору користувачем. Прикладом задач, які будуть виконуватися ними:

- Обчислення фінансових формул
- Обчислення статистики користувача

Збереження даних. Цей рівень відповідає за зберігання та зчитування даних. Основним компонентом цього рівня є PlayerPrefs - це вбудований метод зберігання інформації користувача.

Отже, закінчивши теоретичний блок другого розділу, була обрана мова програмування, а саме c#, яка буде використовуватися в середовищі розробки Unity в якому буде створений додаток на основі дизайну створеному в Figma та архітектурі проекту, побудованому у рамках цієї половини розділу. У наступній частині буде детально показано процес створення застосунку та проблеми з якими зіткнулися під час розробки під час розробки.

2.5 Практична частина

Переходячи до виконання технічної частина перше, що було вирішено зробити, – це створити прототип зовнішнього вигляду додатку у веб-інструменті Figma, про який вже розповідалося раніше, але перед тим треба зрозуміти, яка задача стоїть перед додатком. Так як тема моєї дипломної роботи «Розробка мобільного додатка для управління фінансовими заощадженнями з функцією розрахунку складних відсотків», треба зробити інтерфейс, який має можливість додати логіку, яка буде виконувати необхідні дії такі як:

— Список витрат
— Список надходжень
— Запис витрат
— Запис надходжень
— Калькулятори інвестицій

Отже тепер необхідно придумати, як розмістити весь цей функціонал в інтерфейсі. Почнемо з запису витрат та надходжень.

Запис витрат на надходжень буде оформлюватися завдяки клавіатурі додатку а не пристрою бо вона краще підходить під загальний стиль додатку. Запис витрат та надходжень будуть мати схожу структуру, яка складається з:

— Поля вводу суми
— Типу витрат
— Вбудованої клавіатури

— Кнопок відміни та збереження

Єдиними відмінностями між цими екранами будуть:

— Колір, жовтий та зелений відповідно

— Назва екрану

У кінцевому результаті екран запису витрат та екран надходження будуть мати вигляд, як на рисунках, (Рис 2.1) та (Рис 2.2) відповідно.

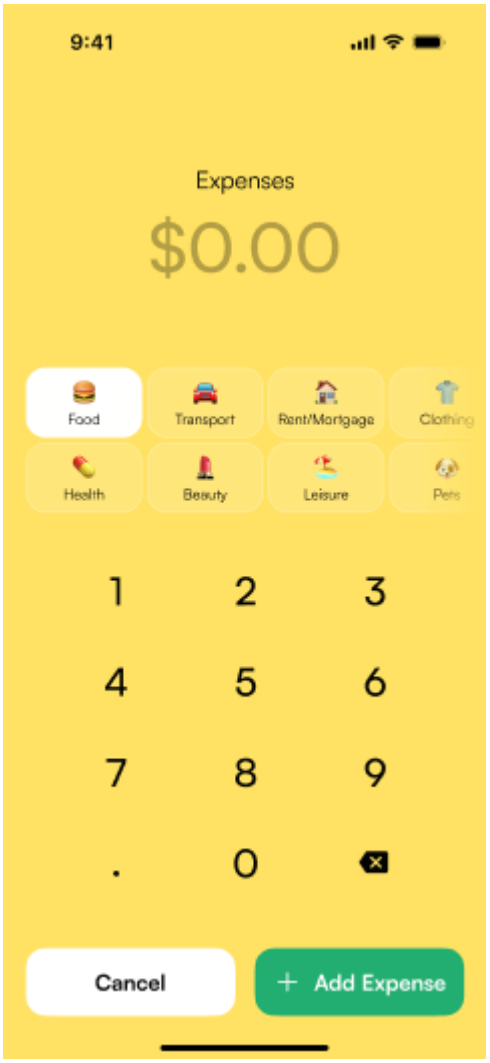

Рис 2.1 Екран витрат

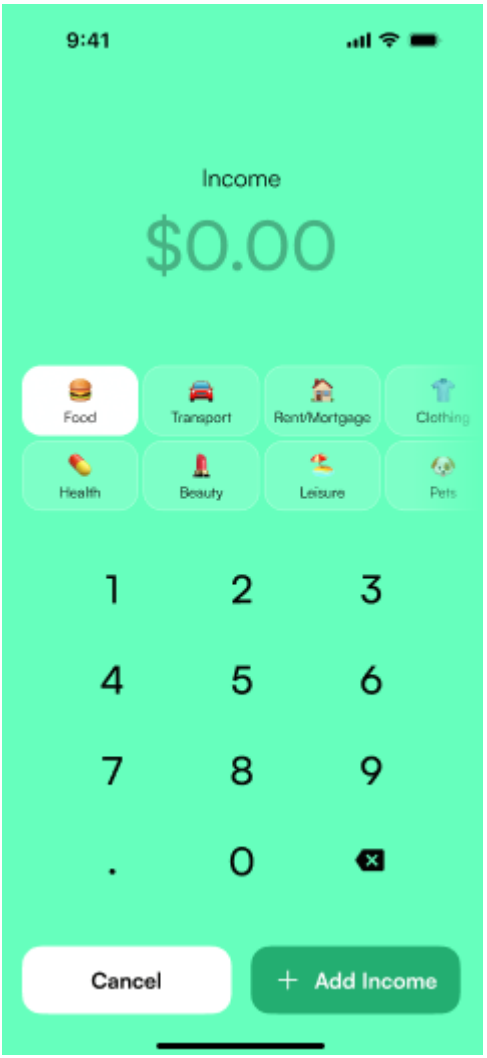

Рис 2.2 Екран надходження

Наступним екраном, над яким була пророблена праця це екран, який має зберігати список витрат та надходжень. Для цього екрану була придумана наступна ідею для реалізації логіки. В ньому будуть зберігатися одразу і надходження і витрати, він буде функціонувати на кшталт історії в звичайних мобільних банкінгах, але у нього буде ще можливість обрання окремо списку надходжень та списку витрат, що буде супроводжуватися відповідними змінами. Тепер розглянемо все це детальніше, кожен з випадків змін екрану.

Пустий екран (Рис 2.3). При старті роботи з додатком, у користувача ще немає історії, котру можна показати на цій сторінці тому, йому відкриті для перегляду наступні пункти:

- Меню додатку. Воно знаходиться в самому низу та містить кнопки переходу на: екран запису витрат, запису надходжень, на цей екран, який вже відкритий та екран калькулятора складних відсотків.
- Список операцій. На даний момент він пустий, крім цього він містить кнопку «All» яка надає можливість перегляду усієї історії детальніше.
- Баланс. Тут записується дельта між надходженням та витратами.

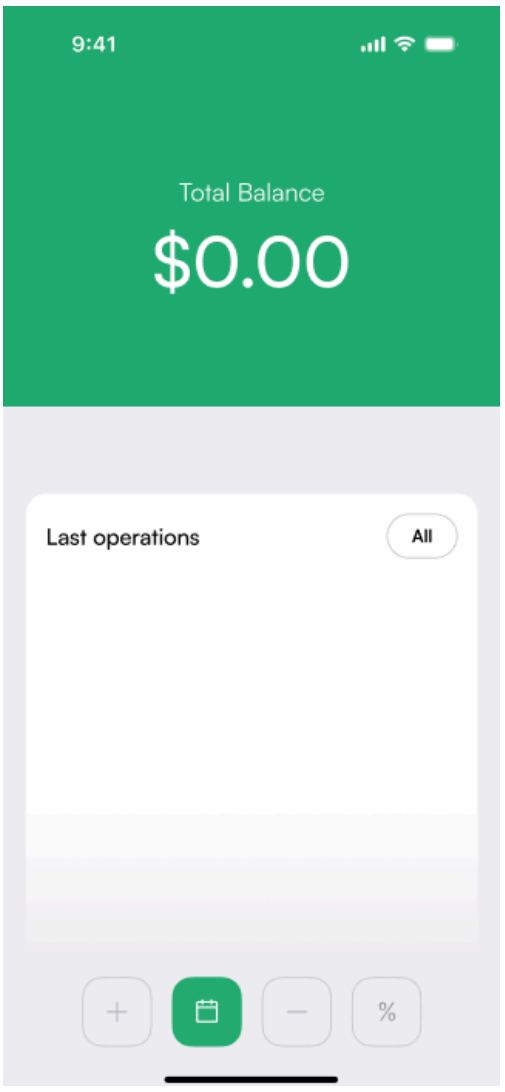

Рис 2.3 Порожній екран витрат

Екран витрат з історією. Цей екран відрізняється від попереднього, тим що в ньому є список операцій проведених користувачем. Новими на цій сторінці є:

- Інформаційне вікно. В цьому вікні можна побачити деяку інформацію про історію користування, скільки всього було надходжень, скільки витрат, середньомісячна інформація та інформація за останній місяць
- Зміна в балансі. Тут вже можна побачити, дельту витрат та надходжень в залежності від значення колір фону змінюється, при позитивній стає зеленим (рис 2.4) при негативній червоним (рис 2.5).

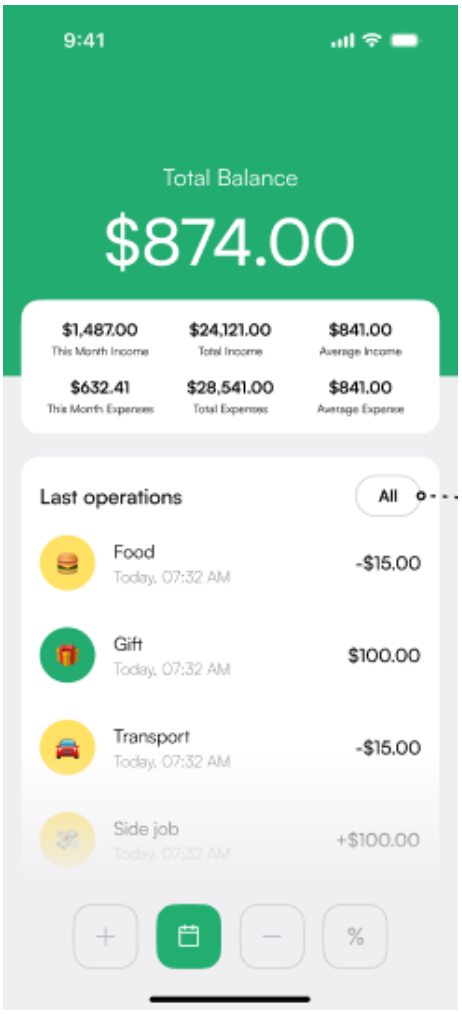

Рис 2.4 Екран витрат та надходжень з позитивною дельтою

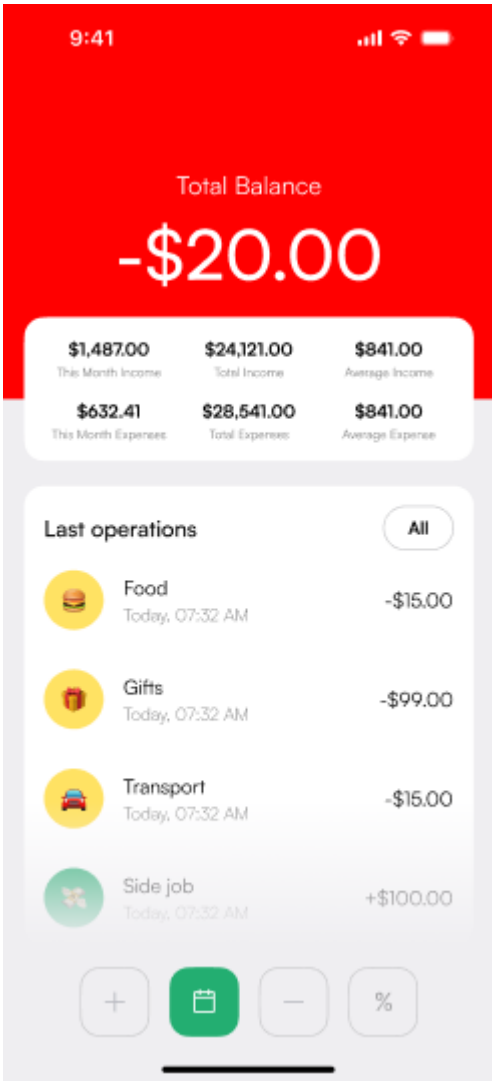

Рис 2.5 Екран витрат та надходжень з негативною дельтою

Екран калькуляторі (Рис 2.6). На цьому екрані знаходяться калькулятори, які будуть реалізовані у цьому додатку, а саме:

- Інвестиції з одним платежем
- Інвестиції з повторюваними платежами

На цьому екрані будуть знаходитися такі компоненти:

- Меню навігації. Воно подібне до того, що знаходиться і на попередньому екрані, але відрізняються тим, що виділена кнопка цього екрану.
- Список калькуляторів. Тут знаходяться калькулятори названі вище.

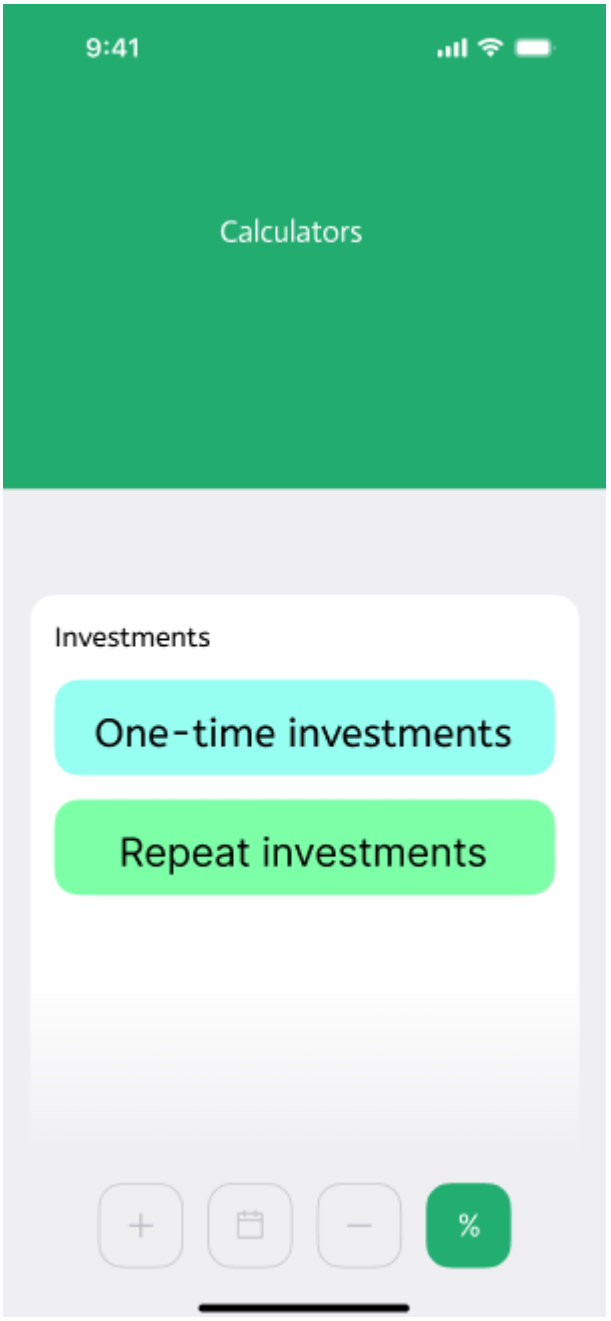

Рис 2.6 Екран калькуляторів

Екрани калькуляторів (Рис 2.7). Калькулятори інвестицій з одним платежем та інвестицій з періодичним платежами мають схожу структуру відмінність лише в кольорі, бірюзовий та зелений колір фону відповідно. Основними компонентами є:

- Поля вводу. Тут знаходяться поля для введення значень для формули.

— Клавіатура. До цієї частини відносяться кнопки вводу чисел, допоміжні кнопки.

— Кнопки закриття, обчислення та переходу на наступне поле

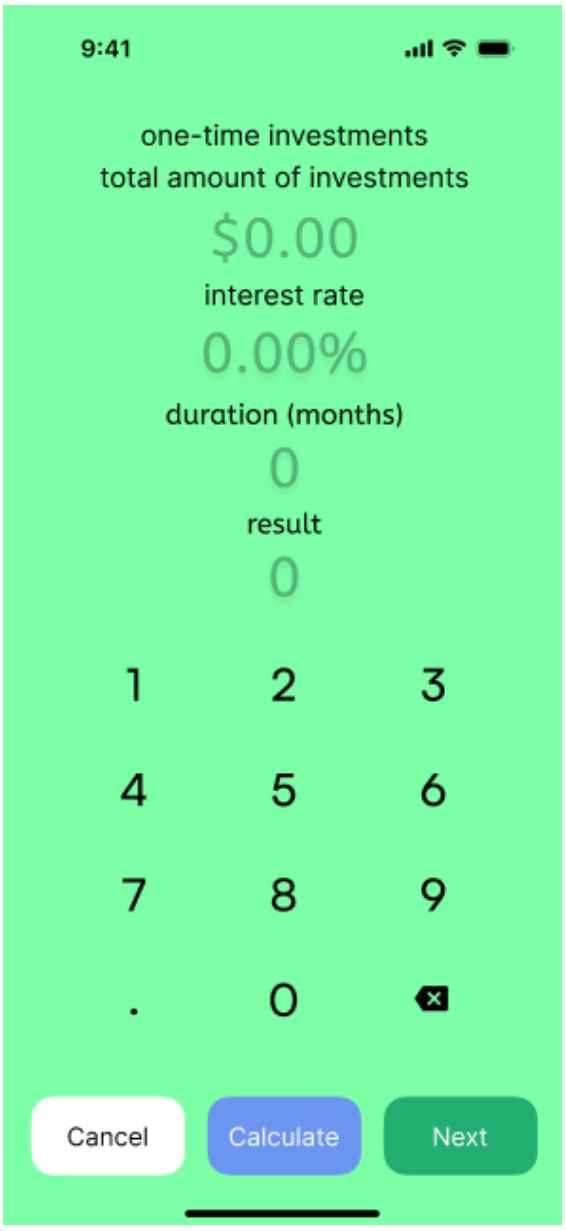

Рис 2.7 Екран калькулятора «Інвестиції з одним платежем»

Тепер коли, були створені всі прототипи інтерфейсу необхідно перенести їх в Unity, для цього треба експортувати всі елементи UI, після чого перенести їх в рушій. Коли в папки проекту були перенесені всі зображення та шрифти, необхідно їх розмістити на Canvas для коректного

переносу необхідно зробити роздільну здатність(розмір екрану) в сцені такою ж, як і в Figma. Для цього треба ввести в параметр Reference Resolution, що знаходиться в компоненті Canvas Scaler і належить Canvas, відповідне значення з прототипу. Після цього розмістити всі зображення та тексти, як прототипі, це можна зробити за допомогою параметру (трансформ) та перенесенню значень з прототипу.

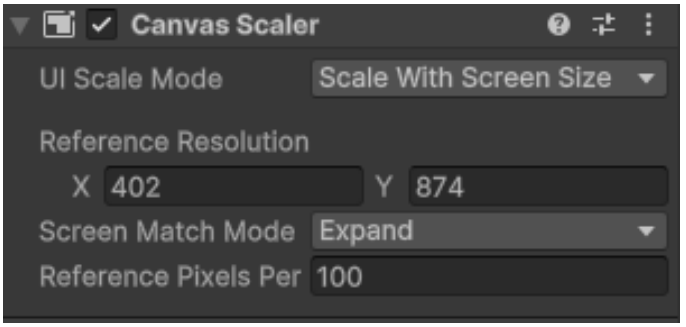

Рис 2.8 Заповнений Canvas Scaler

Тепер коли перенесення інтерфейсу в рушій було завершено необхідно переходити до реалізації логіки.

Першим з чого почнемо, це логіка вікон введення нарахування та витрат. Для цього нам необхідні скрипти:

- Скрипт, який буде виконувати введення значень
- Скрипт, який зберігає введене значення
- Скрипт, який буде відповідати за вибір іконки, яка буде відповідати за відображення категорії

Першим було створено скрипт з назвою «Keypad». Він відповідає за кнопки клавіатури, які відповідають за внесення змін у текстове поле. Основними частинами його є: кнопки цифр, які відповідають за введення відповідного значення, кнопка крапки, яка дозволяє записувати дроби, кнопка видалення, що видаляє останній введений символ.

Другим був, «PrefabDataManager», який реалізовував збереження даних на основі PlayerPrefs, яка використовується для збереження та загрузки даних. Основними функціями цього скрипту були:

- InitializeButtons. Ініціалізація кнопок, які відповідають за іконки.
- Перевірка наявності ключів в PlayerPrefs для цього місяця для надходжень та витрат, методи CheckNewMonthIncome та CheckNewMonthExpenses відповідно
- Створення префабу з унікальним ключем, який зберігає дані про: тип позики, тег (надходження чи витрати), дату та час, значення. CreatePrefabData
- Видалення даних про префаб. DeletePrefabData
- Завантаження даних, для подальшого показу. LoadPrefabData

Для обох вікон, логіка виконується однаково, з єдиною відмінністю, у полі Prefab Tag, компоненту «Prefab Data Manager» вводиться Expenses або Income в залежності від вікна.

Наступною частиною є екран зі списком-статистикою. Тут необхідно створити логіку, яка буде опрацьовувати наступні задачі:

- Виведення на екран списку операцій
- При наявності операцій вивести статистику

Першу задачу виконує «PrefabInitializer», який, методом GetAllPrefabKeys, отримує список ключів всіх збережених елементів, після чого SpawnAndInitializePrefab створює інстанси префабу для кожного збереженого елементу, встановлюючи значення в поля префабу:

- Price – сума операції
- NameLoan – назва операції
- DateLoan – дата запису

З ним взаємодіє DateTabInitializer, що відповідає за створення вкладок для дат та додає префаби до кожної вкладки, завдяки ключам відповідним цій даті.

Ще одним скриптом, який працює «TextToSprite», який знаходиться на логотипі в середині заготовки. Його основна задача перетворити індекс зображення іконки в іконку, це необхідно, бо PlayerPrefs може приймати лише дані типу float, int, string. За виведення статистики відповідає одразу декілька скриптів: «MonthlyExpensesManager» та «MonthlyIncomeManager», вони діють схожим чином, виконуючи наступні дії:

- Підраховують дохід/витрати за останній місяць
- Повертають загальну суму за весь час
- Знаходять середньомісячне значення

Екран детального огляду історії. На цьому екрані необхідно реалізувати:

- Переключення між вкладками: надходження, витрати, все
- Вивід потрібної категорії

Ці дві функції реалізовані в «ButtonToggleManager», детальніше про це:

- ActivateExpensesAndDeactivateIncome. Цей метод вимикає всі об’єкти витрат та залишає лише дохід
- ActivateIncomeAndDeactivateExpenses() - працює навпаки, щодо попереднього методу.
- ResetIncomeAndExpenses() - включає всі об’єкти
- SetActiveObjectsByTag - змінює стан об’єкта в залежності від тегу та булі, яка відповідає за необхідний стан.

Останнім основним скриптом використаним на цьому екрані був GeneralAmount, що відповідає за виведення дельти операцій.

Отже коли логіка екранів пов’язаних з записом операцій створена можна переходити до екрану калькуляторів, але перед цим необхідно створити скрипти які будуть дозволяти перехід між екранами. Ними стали ScreenSwitcher та ButtonManager. Для розуміння чого використовуються два ці скрипта розглянемо їх детальніше.

ScreenSwitcher відповідає за перемикання між екранами, завдяки методам Show та Hide. Він прив’язаний до кнопок повернення до головного меню, що робить його більш вузьконаправленим.

ButtonManager відповідає за керування вкладками, такими, як Історія операцій та Вікно калькуляторів. Його логіка виконується наступним чином:

1. Виділяє кнопку вкладки
2. Переходить на неї
3. Змінює колір кнопки в меню

Дія, яку необхідно виконати у вікні вибору калькулятора - це перехід на екран потрібного калькулятора, це можна виконати за допомогою ButtonManager, що дозволить зробити це наявними ресурсами без створення нового скрипту.

Екран калькулятора. Екран калькулятора працює на кшталт екранів введення операцій, але має зміни такі, як:

- Зміна поля введення значень
- Обчислення результату
- Виведення результату на екран

Вирішення першої задачі було створено InputFieldChanger. Він відповідає за те, щоб змінити поле, в яке буде вводитися значення, це необхідна міра, бо у зв’язку з тим, що значення водиться не нативною клавіатурою, а кастомною, необхідно якось змінювати поле. Для цього був створений метод ChangeInputField, який при натисненні на кнопку буде змінювати поля по кругу завдяки циклу.

Для вирішення другої та третьої задач, розроблено по скрипту для кожного калькулятора. Виконувати свою задачу вони будуть наступними кроками:

1. Перевірка значень на прийнятність
2. Обчислення
3. Виведення на екран

Після цього можна почистити поля та ввести нові значення для перевірки.

Окрім цих скриптів були ще маленькі допоміжні, ось їх перелік:

- UIAnimator, виконує прості анімації
- TextColorChanger, змінює колір в залежності від того, чи негативне значення
- Rebuilder – оновлює значення
- ParentActivator – відповідає за активацію об’єкта в залежності від наявності дочірніх елементів
- FormatCurrencyOnStart – відповідає за форматування чисел в валюту при старті
- FormatCurrency – виконує схожу дію, але ще взаємодіє з кнопкою прийняття, яка стає після введення значення
- EraseText – очищає поле вводу
- DateChecker – видаляє день у випадку, якщо він сьогоднішній
- ChangeColorByParentTag – змінює колір в залежності від тегу батьківського об’єкту

Висновок до розділу 2

Отже, цим можна закінчити опис створення фінансового додатку. У цьому розділі було обрано мову на якій написано цей проект, а саме C#, на основі цього обрано інструмент для створення додатку, з усіх перелічених варіантів зупинившись на рушії Unity, окрім цього для створення дизайну, для цього використано веб-інструмент Figma, бо в порівнянні з іншими він найбільше підходить саме для цієї ситуації, беручи до уваги можливість безкоштовного використання. Після знаходження потрібних інструментів, було перейдено до практичного створення. Успішно перемістивши інтерфейс з Figma в Unity, приступили до створення логіки самого додатку, створивши можливість запису доходу/витрат, їх збереження та можливості перегляду

історії усіх операцій. Фінальною частиною було реалізація калькуляторів. В результаті отримано фінансовий додаток, який, можна використовувати у побуті для відслідковування витрат та надходжень, наприклад загальні витрати по всім рахункам та готівці. Крім цього застосунок містить калькулятори, які можна використати, наприклад, для обчислення потенційного прибутку від інвестиції тільки що записаного надходження коштів.

РОЗДІЛ 3. АНАЛІЗ РЕЗУЛЬТАТІВ ТА ОЦІНКА ЯКОСТІ ЗАСТОСУВАННЯ КАЛЬКУЛЯТОРА

3.1 Аналіз результатів

Отже тепер коли був створений додаток необхідно провести аналіз та оцінку якості застосування. Як додаток з функціями керування бюджету та обчислення інвестицій, завдяки калькуляторам, які для обчислення використовують відповідні формули, він повинен виконувати ці задачі давайте розглянемо детальніше чи справляється він з поставленим завданням.

Почнемо з частини застосунку, яка відповідає за керування бюджетом. Для цього цей застосунок має можливість запису змін в бюджеті та зберігання історії операцій, завдяки чому виконується поставлена перед ним задача. Бонусом до цього є наявність статистики, опрацювання інформації якої дозволяє коригувати бюджет користувача в залежності, від отриманого результату, наприклад для того, щоб порівняти середньомісячні дохід та витрати. Ці можливості можуть бути корисними для користувачів у побуті, для планування сімейного бюджету або просто для відслідковування витрат у повсякденні.

Другою частиною є калькулятори інвестицій, ця частина відтворює роботу двох основних формул для вкладу коштів, а саме: повторювані та одноразові інвестиції. Цей функціонал є корисний у використанні в комбінації з статистикою по витратам та надходження, бо дозволяє обчислити потенційний прибуток від інвестиції частину або усього надлишку отриманого в результаті обчислення дельти по операціям, цю інформацію можна побачити на екрані витрат.

Єдиним недоліком, який можна виокремити є відсутність можливості використовувати один і той самий профіль на декількох платформах, хоча цей момент є не дуже важливим, але він може трохи впливати на якість досвіду користування застосунком.

Резюмуючи досвід від використання застосунку, можна стверджувати, що основна перевага цього додатку не в окремих функціях, а в їх комбінації. Окремий запис доходів та витрат не дасть стільки користі, як їх об’єднання завдяки чому можна проаналізувати весь грошовий потік, який пройшов через запис у цьому додатку, а калькулятори інвестицій, заохочують до інвестицій профіциту бюджету, який ви можете побачити на першому екрані, яким не випадково є саме екран зі статистикою витрат надходжень.

Беручи до уваги вище написаний аналіз можна оцінити якість використання цього додатку на 9/10, один бал знятий за відсутність підтримки декількох платформ.

3.2 Майбутні покращення

Основуючись на вище згаданому опису роботи, додаток хоч і виконує поставлене завдання, але є деякі моменти, які можуть бути покращенні у майбутньому, серед них були виділені наступні:

- Розширення функціоналу застосунку
- Використання одного акаунту на кількох девайсах
- Інтеграція з іншими фінансовими додатками
- Додати навчальний туториал

Розглянемо це детальніше. Почнемо з розширення функціоналу застосунку. Наразі він складається з запису фінансових операцій та калькуляторів для обчислення інвестицій. Хоч цього функціоналу вже достатньо для того, щоб бути корисною для користувача програмою у майбутньому можна додати функціонал, який може стати у нагоді людям, які використовують цей додаток. Серед цих оновлень можна виокремити наступне:

- Можливість поділитися статистикою
- Створення спільного простору для керування бюджетом з іншими користувачами

— Збільшення можливості для фільтрації операцій

Ці додаткові можливості можуть гарно влитися у цей додаток завдяки гарної інтеграції з наявними функціями.

Використання одного акаунти на кількох девайсах, може стати у нагоді користувачам, які використовують багато пристроїв у повсякденні. Це може покращити досвід користувача та дасть більшу гнучкість, завдяки відсутності необхідності використовувати лише один пристрій для внесення нотаток.

Додавання навчального туторіалу є важливим пунктом серед майбутніх оновлень. Це пов’язано з тим, що незважаючи на свою користь він може бути заважким у використанні для деяких верств населення. Задля допомоги їм необхідно буде створити навчальних гайд з керування додатком, який простою мовою зможе пояснити, як виконуються його функції.

Інтеграція з іншими фінансовими додатками. Деякі додатки, що схожі на застосунок, зроблений у рамках кваліфікаційної бакалаврської роботи мають можливість інтегрувати в них дані з банківських рахунків на пряму, що полегшує ведення статистики у додатку, завдяки отримання інформації з одного або декількох рахунків. У майбутньому це можна зробити, але для введення цієї функції треба забезпечити надійність зберігання даних користувачів.

ВИСНОВОК

Перед створенням проекту з дипломної роботи, було виконано детальне ознайомлення з станом навичок пов’язаних з фінансовою грамотністю у населені. Дізнавшись, що рівень знання у цій сфері дуже низький, особливо у молодого покоління, було вирішено створити мобільний застосунок, що є інструментом для розрахунку бюджету, який націлений саме на цю версту населення, бо вони більш часу приділяють. Оглянувши теоретичну інформацію пов’язану з цією сферою, були обрані необхідні формули для додатку та визначилася з класифікацією додатку, ним став калькулятор заощаджень, у основі якого лежить запис доходів та витрат, інформація про які використовується для статистики додатку такої, як:

- Дельта операцій
- Дохід/витрати за місяць, за весь час, та середньомісячні

Наступним, що було зроблено у ході виконання цієї дипломної роботи, стало вибір інструментів необхідних для виконання цієї роботи. Ними стали:

- Мова – C#
- Середовище розробки – Unity
- Інструмент для створення дизайну – Figma

Розібравшись з тим, які інструменти використовуються для створення додатку, почалася робота над технічною частиною. Перенісши інтерфейс з інструменту в рушій, почалися обміркування над структурою та логікою роботи додатку. У кінцевому результаті був створений екрани введення надходжень та витрат, значення яких записуються у PlayerPrefs після чого стають доступні у списку всіх операцій. Крім цього було створено вікно калькуляторів, у якому є можливість обрати один з доступних. Робота калькуляторів була перевірена та не було виявлено явних проблем в її роботі.

В межах останнього розділу був проведений аналіз додатку та оцінка якості застосування калькуляторів.

Підсумовуючи виконану дипломну роботу, можна стверджувати, що набуті навички зі створення фінансових додатків та розробки користувацького інтерфейсу є цінним досвідом, який може бути успішно застосований у майбутній професійній діяльності.

ВИКОРИСТАНІ ДЖЕРЕЛА

1. Національний банк України. Результати опитування з фінансової грамотності для підлітків 11–18 років [Електронний ресурс] / Національний банк України. – 2024. – Режим доступу: https://talan.bank.gov.ua/novyny/rezultati-opituvannia-z-finansovoyi-gramotnosti-dlia-pidlitkiv-11-18-rokiv

2. Презентація «Основи фінансового планування та заощадження коштів» [Електронний ресурс] / Національний банк України. – 2024. – Режим доступу: https://talan.bank.gov.ua/uploads/navchalni_materialy/Презентація_Основи%20фінансового%20планування%20та%20заощадження%20коштів.pdf

3. Best Investment Apps Of May 2024 [Електронний ресурс] // Forbes Advisor. – Режим доступу: https://www.forbes.com/advisor/investing/best-investment-apps/

4. How Mobile Banking Apps Are Transforming Personal Finance Management [Електронний ресурс] // CEO Review. – Режим доступу: https://www.ceo-review.com/how-mobile-banking-apps-are-transforming-personal-finance-management/

5. Compound Interest [Електронний ресурс] // Investopedia. – Режим доступу: https://www.investopedia.com/terms/c/compoundinterest.asp

6. Dart language overview [Електронний ресурс] // Dart.dev. – Режим доступу: https://dart.dev/language

7. Null safety in Dart [Електронний ресурс] // Dart.dev. – Режим доступу: https://dart.dev/null-safety

8. React Native – Build native apps using React [Електронний ресурс] // ReactNative.dev. – Режим доступу: https://reactnative.dev/

9. Ionic – Cross-Platform Mobile App Development [Електронний ресурс] // Ionicframework.com. – Режим доступу: https://ionicframework.com/

10. NativeScript [Електронний ресурс] // NativeScript.org. – Режим доступу: https://nativescript.org/

11. Xamarin Documentation [Електронний ресурс] // Microsoft Docs. – Режим доступу: https://docs.microsoft.com/en-us/xamarin/

12. NET MAUI documentation [Електронний ресурс] // Microsoft Learn. – Режим доступу: https://learn.microsoft.com/en-us/dotnet/maui/?view=net-maui-9.0

13. Install the Unity Hub and Editor [Електронний ресурс] // Unity Learn. – Режим доступу: https://learn.unity.com/tutorial/install-the-unity-hub-and-editor

14. UI Toolkit manual [Електронний ресурс] // Unity Manual. – Режим доступу: https://docs.unity3d.com/Manual/UIToolkits.html