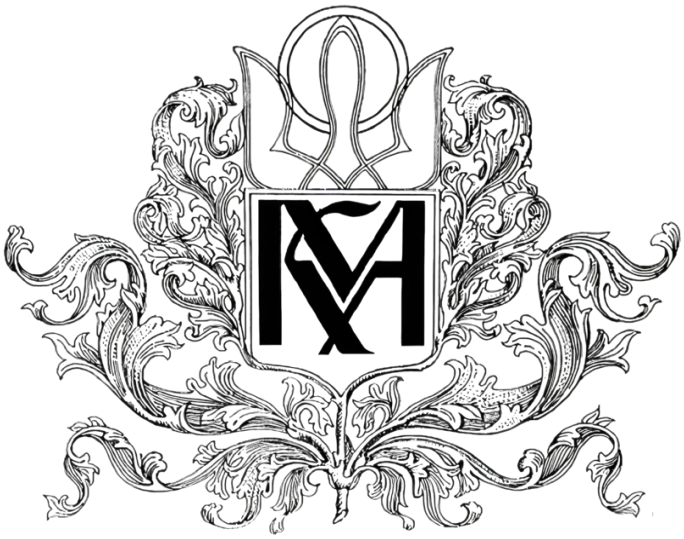
Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра мультимедійних систем



Мобільний застосунок для обліку фінансових витрат на базі Jetpack Compose - Kotlin для Android

Текстова частина до курсової роботи

за спеціальністю «Інженерія програмного забезпечення»

Керівник курсової роботи
ст. викладач Вовк Н. Є.

_____
(підпис)
“____” _____ 2023 р.

Виконав студент 3 р. н.
Мельник І. О.
“____” _____ 2023 р.

Київ – 2023

Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра інформатики факультету інформатики

ЗАТВЕРДЖУЮ

Зав. кафедри мультимедійних систем

доцент, к.ф.-м.н.

О. П. Жежерун

_____2023 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на курсову роботу

студенту Мельник Ілля Олександрович 3-го курсу факультету інформатики

ТЕМА Розробка додатка на базі Jetpack Compose - Kotlin для Android

Зміст ТЧ до курсової роботи:

Індивідуальне завдання

Календарний план виконання роботи

Анотація

Вступ

1 Огляд та особливості створення програм на базі JetPack Compose

2 Проектування та аналіз поставленої задачі

3 Реалізація застосунку

Висновки

Список використаних джерел

Дата видачі „___” _____ 2023 р. Вовк Н.Є. _____

(підпис)

Завдання отримав _____

(підпис)

Календарний план виконання роботи:

№ п/п	Назва етапу	Термін виконання	Примітка
1.	Отримання завдання на курсову роботу	Липень 2022	
2.	Ознайомлення з середовищем розробки Android Studio	Липень 2022	
3.	Детальне ознайомлення з бібліотекою JetPack Compose	Липень-Серпень 2022	
4.	Продумування основних функцій додатка	Серпень 2022	
5.	Проектування архітектури програми	Серпень - Вересень 2022	
6.	Реалізація проекту	Вересень - Листопад 2023	
7.	Написання текстової частини роботи	Березень-Квітень 2023	
8.	Створення презентації роботи	Травень 2023	

Студент Мельник І.О.

Керівник Вовк Н. Є.

“_____”_____

Зміст

Анотація

Основна ідея курсової роботи - це створення додатку для відслідковування фінансових витрат за допомогою бібліотеки JetPack Compose та мови програмування Kotlin для операційної системи Android. Сам процес розробки відбувався в IDE Android Studio від JetBrains.

Результатом роботи є готовий додаток, який дозволяє не тільки вносити дані про витрати, а й слідкувати за тим, на які саме типи товарів були витрачені кошти; також були реалізовані інші додаткові можливості, що передбачалися функціональними вимогами.

Ключові слова: Kotlin, мобільна розробка, Android, JetPack Compose, XML, Java, Android Studio, Rest API.

Вступ

Світ не стоїть на місці, останні роки люди все частіше стараються бути продуктивними та використовувати власні ресурси з розумом. В нагоді стають додатки, що дозволяють відслідковувати витрати коштів.

Вони сприяють кращому управлінню фінансами, в результаті заощаджуючи час та кошти. Зокрема, вони допомагають систематизувати і категоризувати витрати, що полегшує розуміння того, на що ви витрачаєте гроші. Завдяки аналізу витрат, додатки дають змогу виявляти категорії товарів або послуг, які абсорбують більшу частину бюджету.

Робота присвячена створенню подібного додатка для операційної системи Android. У текстовій частині описаний функціонал застосунку, логіка взаємодії вікон та різних функцій.

У першому розділі розповідається про можливості, що надає бібліотека JetPack Compose Kotlin, та порівняння її з XML + Java.

Другий розділ містить інформацію про аналіз подібних застосунків та основні функціональні вимоги, що повинна мати фінальна версія програми.

Третій розділ описує можливості програми та як саме працює той чи інший функціонал.

У висновку містяться практичні навички, які були засвоєні під час написання застосунку.

Основна частина

Розділ 1 : Розробка застосунків у Android Studio та порівняння Kotlin/JetPack Compose з Java/XML

1.1 Порівняння Kotlin/JetPack Compose з Java/XML

JetPack Compose - це сучасна бібліотека від Google, що застосовується для створення графічного інтерфейсу користувача (UI) . Першочергово бібліотека була створення для операційної системи Android , проте за допомогою Jetpack Compose Multiplatform , можна створювати застосунки і для інших операційних систем таких як : Windows, macOS, and Linux.

Стабільна 1.0 версія біблотеки вийшла 28 липня 2021 . Compose був спроектований саме для полегшення розробки інтерфейсів та покращення продуктивності розробників за допомогою сучасного підходу до створення UI. Розробка за допомогою неї відрізняється від класичного підходу до розробки на Android з використанням Java/XML. Можна виділити такі пункти як :

Декларативний підхід: JetPack Compose використовує декларативний підхід до створення UI, що полегшує розробку і забезпечує більш зрозумілу структуру коду. У той час як у Java/XML потрібно описувати логіку створення і зміни UI у відповідних Java-класах, в JetPack Compose це робиться прямо в межах компонентів UI, що дозволяє легше стежити за взаємодією між кодом і візуальною частиною застосунку.

Компоненти: JetPack Compose пропонує модульну структуру, яка заснована на компонентах. Ці компоненти є незалежними один від одного, що дозволяє легко масштабувати і перевикористовувати код. У відміну від Java/XML, де UI-компоненти описуються в окремих XML-файлах, в JetPack Compose вони описуються в межах коду на Kotlin, що полегшує розробку і рефакторинг коду.

Інтеграція з Kotlin: JetPack Compose повністю інтегрований з мовою програмування Kotlin, що дає можливість використовувати всі переваги цієї мови, такі як короткий синтаксис, безпечність типів, розширення функцій та інші . Це дозволяє розробникам створювати чистіший, ефективніший та безпечніший код порівняно з Java/XML. Kotlin також пропонує зручні механізми, такі як корутини, для асинхронного програмування та управління потоками, що сприяє покращенню продуктивності розробників та стабільності застосунків.

Оновлення в реальному часі: Завдяки декларативному підходу JetPack Compose та системі State, зміни стану компонентів автоматично відображаються в UI без додаткових втручань з боку розробника. У той час як у Java/XML необхідно вручну оновлювати UI у відповідних обробниках подій та використовувати системи прив'язки даних.

Анімації та переходи: JetPack Compose спрощує створення анімацій та переходів між екранами, надаючи вбудовані функції та інструменти. У порівнянні з Java/XML, де реалізація анімацій може бути складною та зайняти багато часу, JetPack Compose дозволяє швидко реалізувати складні ефекти та анімації з меншими зусиллями.

Більші можливості для перевикористання коду: Завдяки модульній структурі на основі компонентів, розробники можуть легко перевикористовувати код для створення інших екранів або компонентів у своїх застосунках.

Гаряче перезавантаження (Hot Reload): Завдяки інтеграції з Android Studio, розробники можуть побачити зміни у коді майже відразу після збереження, що полегшує процес налагодження та тестування.

Інтеграція з іншими бібліотеками: JetPack Compose може бути легко інтегрований з іншими бібліотеками Jetpack, такими як ViewModel, LiveData, Navigation, Room, та ін.

Проте вона має і ряд недоліків такі як:

Молода технологія: JetPack Compose все ще є порівняно новою технологією, через що може бути менше ресурсів для навчання, документації та прикладів. Це може збільшити криву навчання для розробників, які вперше з ним працюють.

Підтримка старих версій Android: JetPack Compose має обмежену підтримку старих версій Android (API рівень 21 і вище), тому розробники можуть зіткнутися з проблемами сумісності зі старішими пристроями.

Міграція з існуючих проектів: Міграція існуючих проектів на JetPack Compose може бути складною задачею, оскільки вона вимагає переписання UI-коду та адаптації архітектури застосунку.

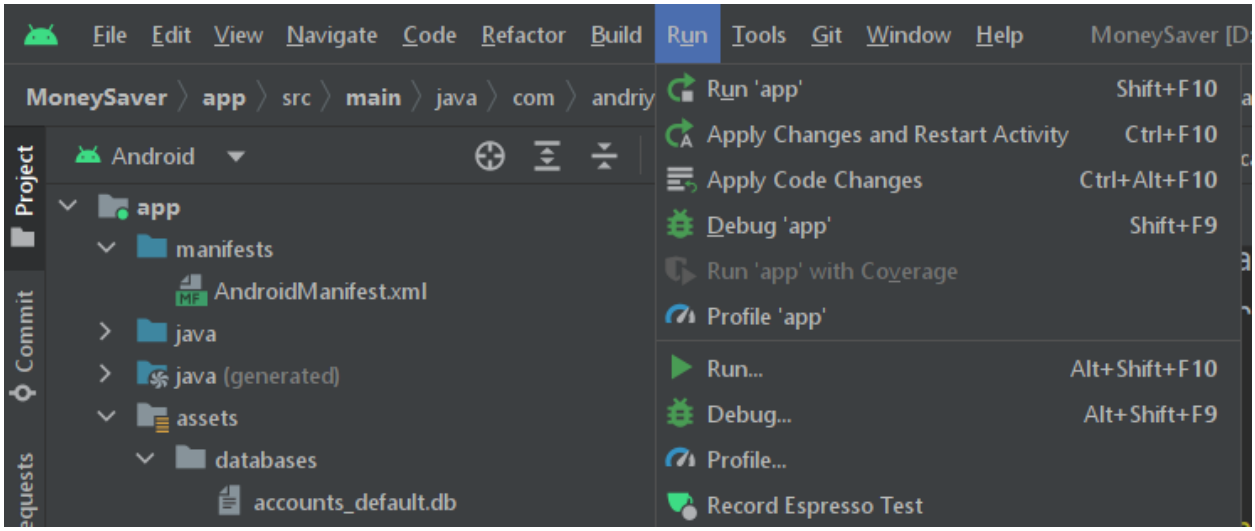
Висновок: JetPack Compose - це новітній фреймворк для створення UI в Android-додатках, який відрізняється від традиційного Java/XML підходу завдяки декларативному стилю, інтеграції з Kotlin, автоматичному оновленню UI, модульній структурі та спрощенню анімацій. Ці відмінності роблять JetPack Compose привабливішим варіантом для розробки сучасних

Android-додатків, дозволяючи створювати більш гнучкі, масштабовані та продуктивні проекти.

1.2 Огляд Android Studio IDE

Android Studio - це офіційне інтегроване середовище розробки (IDE) від Google для розробки застосунків на платформі Android. Android Studio забезпечує ряд інструментів, які полегшують процес розробки, налагодження, тестування та виведення на ринок застосунків для Android.

Рисунок 1. Інтерфейс меню Android Studio



Основні функції та можливості Android Studio :

Інтелектуальне редактування коду: Android Studio має потужний редактор коду, який підтримує синтаксичне виділення, автозаповнення, перехід до визначення та інші можливості для швидкої та ефективної розробки.

Інтеграція з Gradle: Android Studio інтегровано з системою збірки Gradle, яка дозволяє автоматизувати процес збірки, тестування та розгортання застосунків.

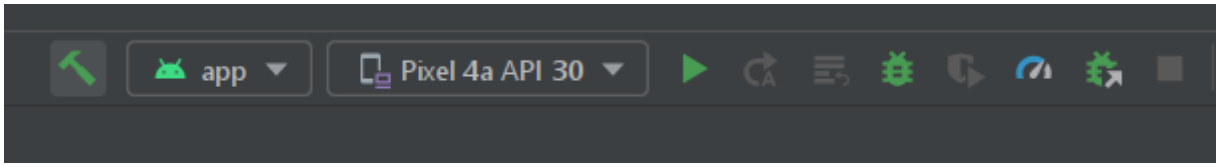
Підтримка різних мов програмування: Android Studio підтримує мови програмування Java, Kotlin та C++, що дає розробникам гнучкість у виборі мови для своїх проектів.

Гаряче перезавантаження (Instant Run та Apply Changes): Android Studio має функції, які дозволяють швидко перезапускати застосунок після внесення змін до коду без повного перезавантаження застосунку.

Профілювання та налагодження: Android Studio включає потужні інструменти для профілювання та налагодження застосунків, що допомагає розробникам виявити та вирішити проблеми з продуктивністю та якістю коду.

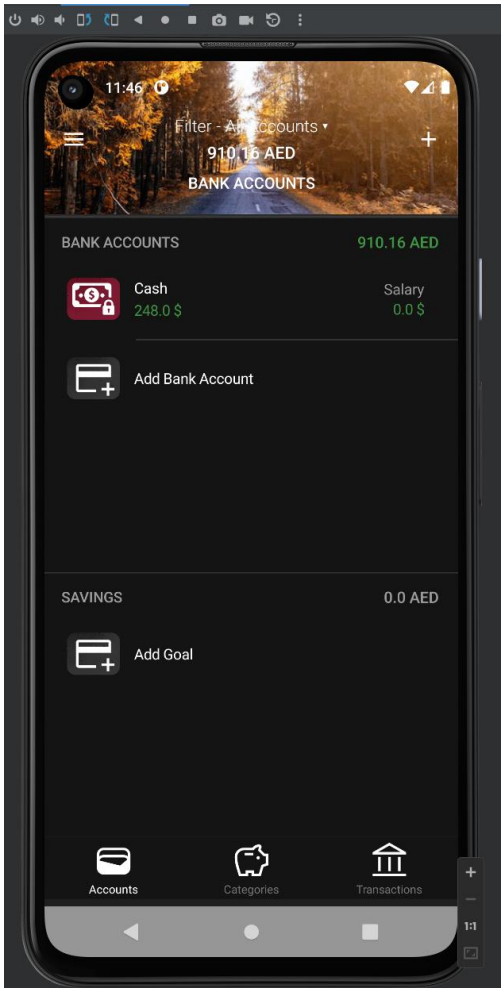
Інтеграція з системами контролю версій: Android Studio підтримує інтеграцію з системами контролю версій, такими як Git, SVN та ін., що спрощує роботу над проектами в команді.

Рисунок 2. Панель запуску емулятора в Android Studio



Також ця IDE дозволяє з легкістю створювати нові профілі емуляторів, як телефонів так і різних інших пристроїв, що базуються на ОС Android , при цьому можна обирати різні налаштування пристрою : модель , марку , версію SDK,

Рисунок 3. Вигляд емулятора в Android Studio



Extended Controls і емулятор в Android Studio дозволяють розробникам перевіряти роботу своїх Android-додатків на різних пристроях, конфігураціях та версіях Android без необхідності мати фізичний пристрій. Емулятор Android створює віртуальний пристрій (AVD), який імітує реальний смартфон або планшет з певними характеристиками.

Емулятор має наступні основні можливості:

Широкий вибір віртуальних пристроїв: Можна налаштувати віртуальний пристрій з різними роздільними здатностями, розмірами екрану, версіями Android, а також вибрати між смартфонами, планшетами, телевізорами, Wear OS та ін.

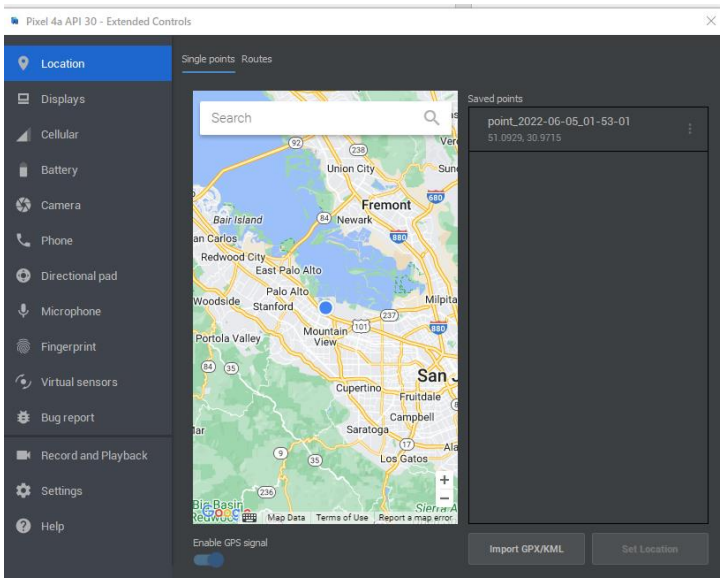
Імітація сенсорів та аксесуарів: Емулятор дозволяє імітувати роботу

сенсорів (акселерометр, гіроскоп, сенсор наближення та ін.) та аксесуарів (GPS, камера, мікрофон, батарея).

Extended Controls – це панель налаштувань емулятора, яка дозволяє керувати його різними параметрами та можливостями.

Основні опції Extended Controls включають:

Рисунок 4. Панель Extended Controls



Місцезнаходження: симулює різні координати GPS для тестування роботи застосунку з геолокаційними сервісами.

Батарея: імітує різний рівень заряду батареї та стан заряджання для тестування реакції додатку на ці параметри.

Камера: дає можливість використовувати камеру комп'ютера або зображення з файлу для симуляції роботи на віртуальному пристрої.

Телефон: симулює вхідні та вихідні дзвінки, а також надсилання та отримання SMS-повідомлень для перевірки реакції застосунку на події пов'язані з телефонною функціональністю.

Мережа: імітує різні стани підключення до мережі, типи мережі (Wi-Fi, мобільний інтернет) та швидкість передачі даних для тестування роботи додатку при різних умовах з'єднання.

Віртуальна реальність (VR): імітує роботу VR-сенсорів та аксесуарів для тестування реакції застосунку на події, пов'язані з віртуальною реальністю.

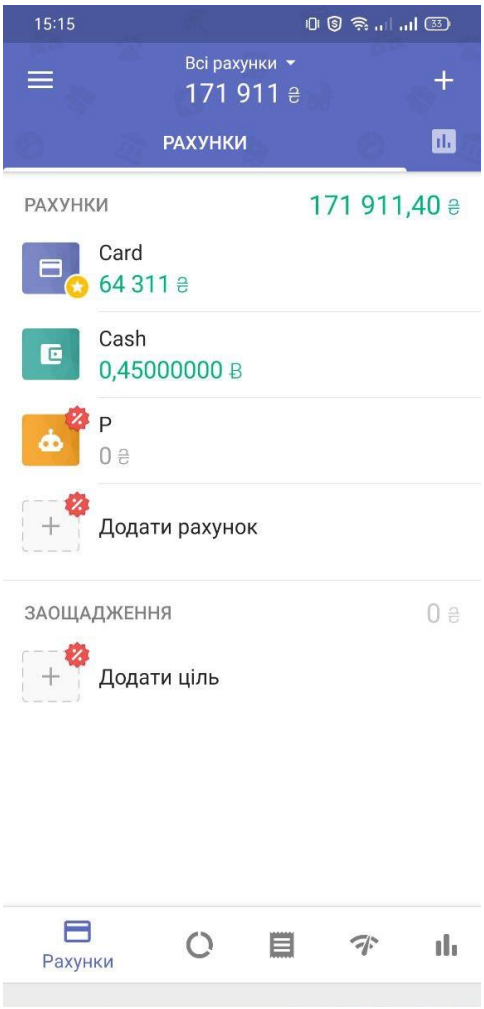
Аудіо: керує налаштуваннями аудіо для віртуального пристрою, включаючи гучність різних типів звуків та режим "Не турбувати".

Симуляція оновлень системи: тестує роботу додатку під час оновлення системи для виявлення можливих проблем, пов'язаних з цим процесом. Використання емулятора та Extended Controls в Android Studio допомагає розробникам тестувати свої застосунки в різних сценаріях і налаштуваннях, що дозволяє забезпечити більш стабільну та якісну роботу додатків на реальних пристроях.

Розділ 2 : Проектування та аналіз поставленої задачі

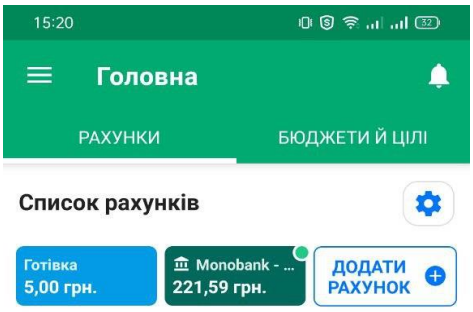
2.1 Схожі програми

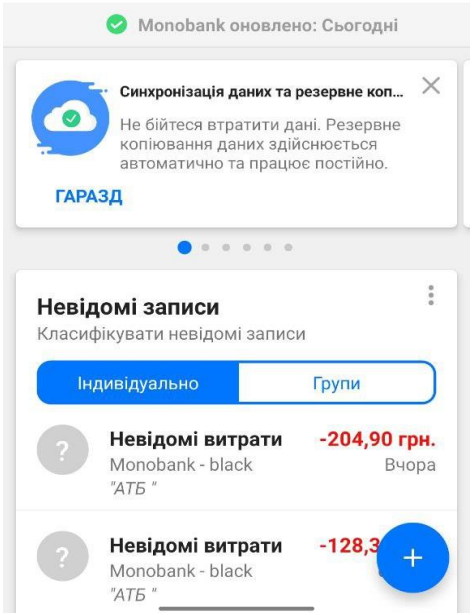
Рисунок 5. Знімок екрана з програми "1Money "



1Money - це безкоштовний додаток що дозволяє контролювати свої витрати , має можливість створювати бюджет , а також зручний графік , що показує кількість витрат в тій чи іншій категорії . Додаток був створений компанією PixelRush, має 5+ млн завантажень так близько 117тисяч відгуків в Google Play (2Q 2023).Додаток також має певну підписку , що дає додаткові можливості такі як зміну теми , захист по відбитку пальця та резервне копіювання.

Рисунок 6. Знімок екрана з програми "Wallet"





Wallet - це також безкоштовний додаток що дозволяє контролювати свої витрати, та має схожі функції , проте він має можливість синхронізовувати витрати з банками , таким чином користувач економить час , аби не вводити дані вручну .Також має можливість підписки що надає , необмежену кількість рахунків , банківських з'єднань та власний колір міток . Розроблений BudgetBakers, має 5+ млн завантажень так близько 250тисяч відгуків в Google Play (2Q 2023)

2.2 Функціональні вимоги

Спираючись на власний досвід та аналіз схожих по застосування додатків було прийнято ряд функціональний вимог такі як:

- Екран з інформацією про баланси рахунків .
- Екран з інформацією про витратив та надходження в тій чи іншій категорії .
- Екран з транзакціями .
- Можливість зміни мови .
- Можливість зміни основної валюти .
- Конвертація валюти за поточним курсом.
- Можливість захисту входу в додаток за допомогою 6 значного пароля або біометрії пальця.
- Можливість змінювати фон топ бару.
- Можливість додавати сповіщення в певний час доби.
- Можливість зміни теми (світла та темна)
- Додавання нових рахунків
- Кастомізація нових рахунків
- Додавання категорій
- Кастомізація категорій
- Можливість проводити транзакції між рахунками
- Калькулятор що дозволяє зручно порахувати певну суму в момент проведення транзакції
- Три види рахунків (Борговий , Заощаджувальний , Звичайний)
- Сортування інформації про категорії по даті
- Зручний графік зо показує значення категорій залежно від їх кольору та суми
- Загальна інформація про початковий баланс та кінцевий в певному проміжку часу
- Можливість редагування транзакцій
- Можливість сортування транзакцій по даті
- Можливість видалення та редагування транзакцій свайпом

Заплановані покращення:

- Можливість створення бюджету
- Можливість перегляду витрат на графіку в XY площині , де X – це час , Y – це витрати в певній категорії в той чи інший час.
- Можливість зберігати дані
- Можливість синхронізації даних з різних банків

Розділ 3 : Реалізація застосунку

3.1 Реалізація та основні бібліотеки

Головна ідея полягала у розробці додатку MoneySaver, використовуючи виключно Jetpack Compose/Kotlin із усіма запланованими функціональними можливостями. Було прийнято рішення використати MVVM архітектуру, де кожен екран має свій власний ViewModel, що дозволяє паралельно розробляти різні частини програми. Тимчасові значення для відображення на екрані зберігаються в State, тобто кожен екран має свій власний State і ViewModel.

Для зберігання категорій, транзакцій та рахунків використовується бібліотека Room. Налаштування, такі як час сповіщення, вибрана тема, основна валюта та мова, зберігаються в Shared Preferences.

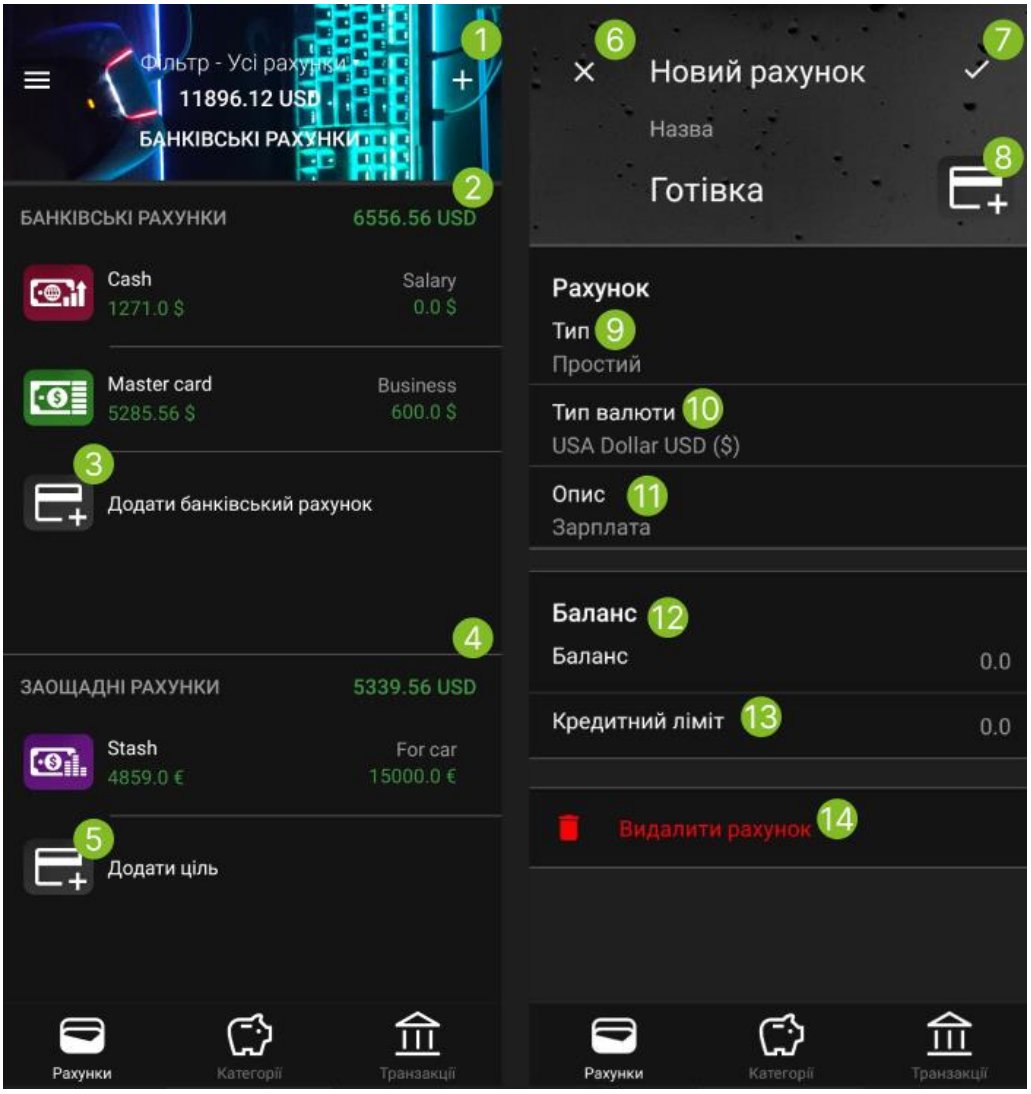
Залежності вирішено керувати за допомогою Dagger:Hilt, що дозволяє зручно впроваджувати Dependency Injection. Графік побудований з використанням бібліотеки ma.hu:compose-charts. Вибір кольору при створенні рахунку або категорії здійснюється за допомогою com.github.skydoves:colorpicker-compose. Для отримання поточного курсу валют використовується бібліотека Retrofit , яка є зручним інструментом для взаємодії з API. Вибір дати та часу реалізовано за допомогою стандартних компонентів, таких як DatePicker та TimePicker.

Для забезпечення захисту використовується біометричний захист, реалізований за допомогою androidx.biometric.

Узагальнюючи, розробка додатку MoneySaver з використанням Jetpack Compose/Kotlin та MVVM, разом з використанням відповідних бібліотек та компонентів, надає можливість створити функціональний та захищений додаток для управління фінансами.

3.2 Головний екран рахунків

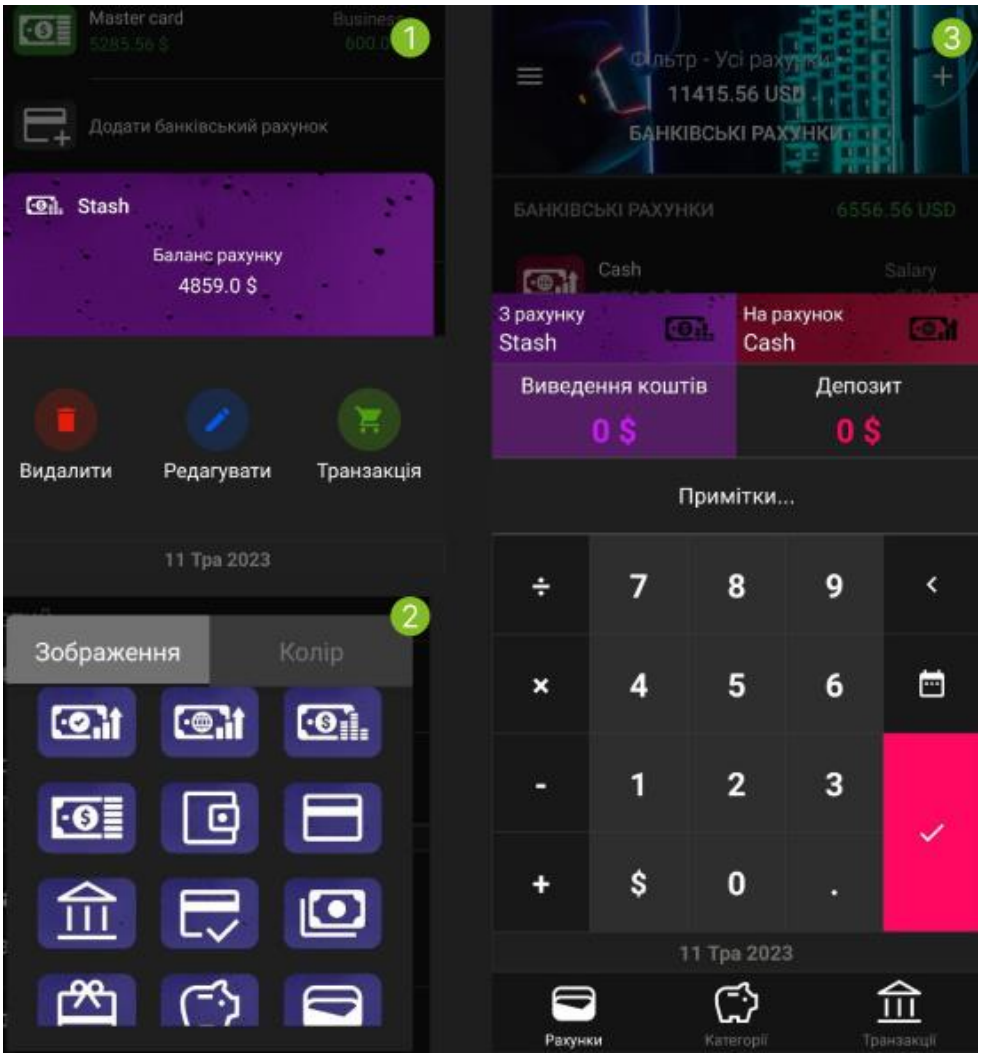
Рисунок 7. Знімок екрана з "MoneySaver" (Головний екран та екран додавання/редагування рахунку)



1. При натисканні на плюс відкривається діалогове вікно, яке дозволяє обрати тип рахунку, який потрібно створити.
2. Загальна сума всіх банківських рахунків (боргових та звичайних) в основній валюті - розраховується за допомогою даних про поточні курси валют.
3. Одразу обирає звичайний рахунок для створення .
4. Загальна сума всіх ощадний рахунків в основній валюті - розраховується за допомогою даних про поточні курси валют.

5. Одразу обирає ощадний рахунок для створення .
6. Скасувати додавання / редагування рахунку
7. Завершити додавання рахунку.
8. Змінити зображення рахунку
9. Тип рахунку / Якщо вікно відкрите для редагування рахунку , тоді це поле не можна змінювати
10. Встановити валюту рахунка
11. Вказати опис рахунка
12. Баланс рахунку
13. Кредитний ліміт / Борг / Ціль (залежно від типу рахунку)
14. Видалення рахунку

Рисунок 8. Знімок екрана з "MoneySaver" (Частини головного екрану)

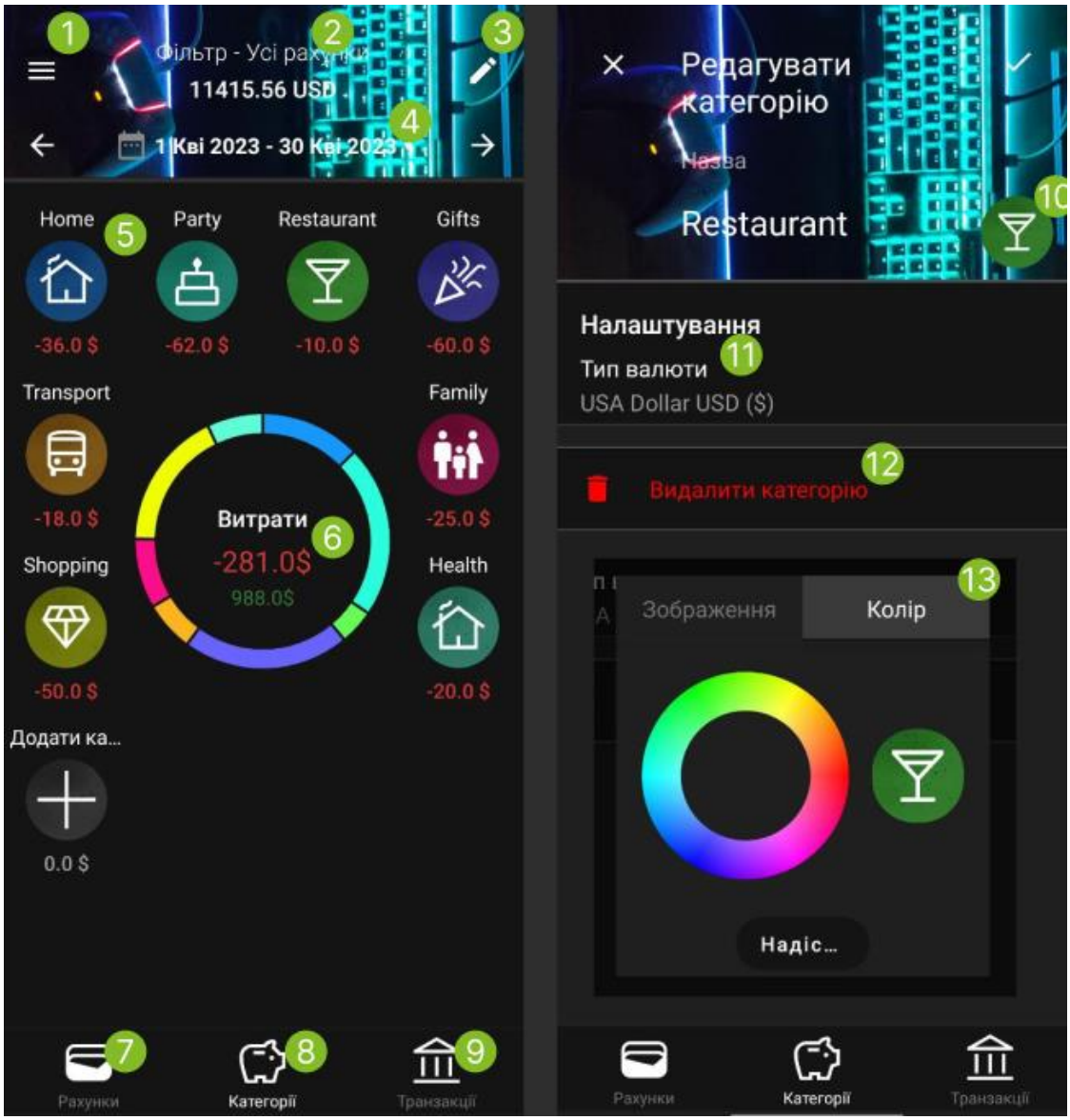


1. При натисканні на рахунок з'являється спливаюче віконце, де можна змінити рахунок, видалити його або провести транзакцію між рахунками.
2. Існує вікно, в якому можна змінити іконку рахунку.

3. Вікно транзакцій між рахунками дозволяє виконувати наступні дії: задавати примітку, змінювати рахунки, автоматично конвертувати валюту, якщо рахунки мають різні валюти, встановлювати дату транзакції. Також в цьому вікні є калькулятор, який дозволяє виконувати арифметичні операції з введеними коштами.

3.3 Екран категорій

Рисунок 9. Знімок екрана з "MoneySaver" (Екран категорій та екран додавання/редагування категорій)

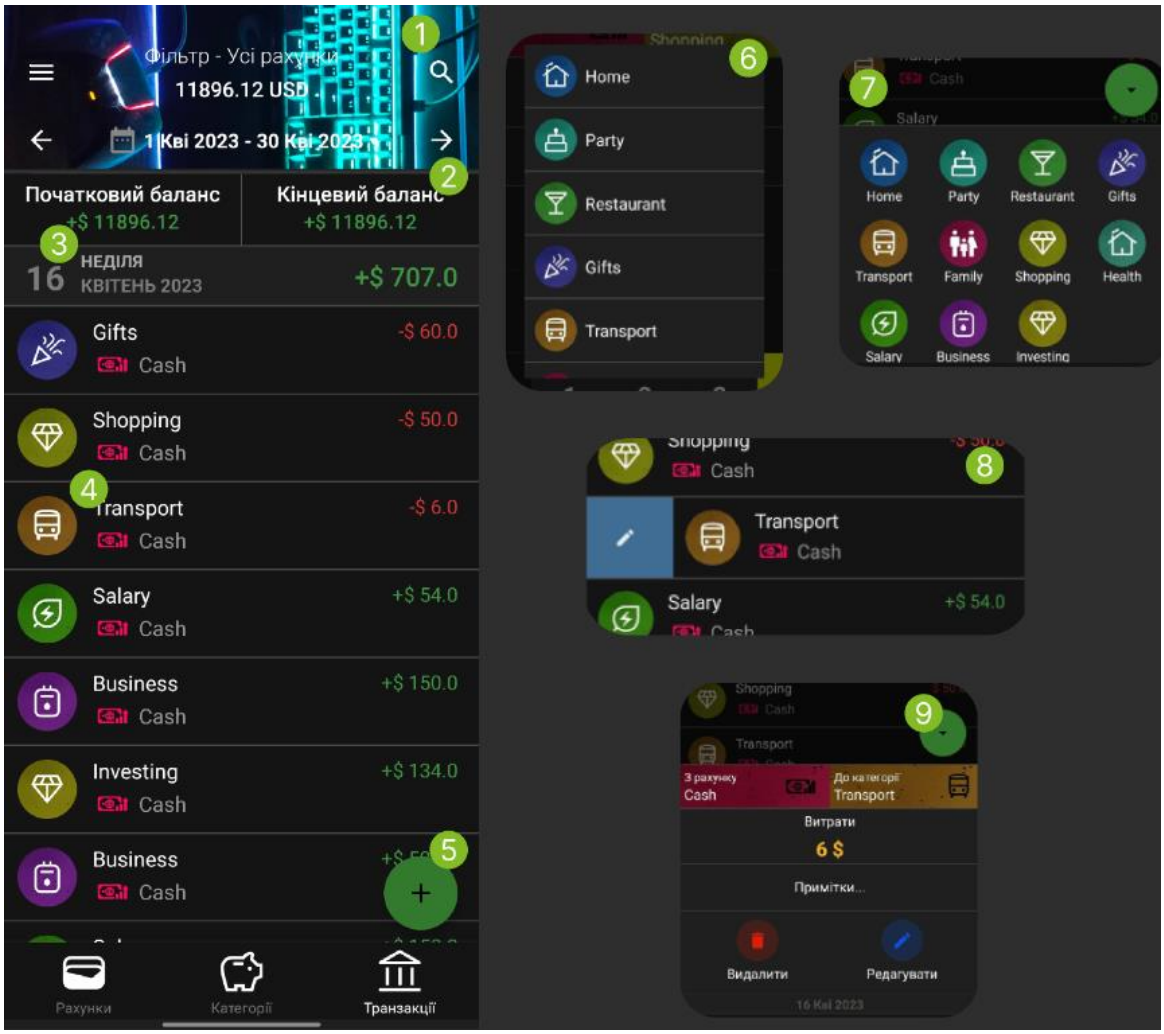


1. Кнопка, що викликає меню програми.
2. Загальна сума всіх рахунків з урахуванням боргів в основній валюті.
3. Кнопка, яка викликає екран редагування/додавання категорій.

4. Фільтр за датою, який дозволяє змінювати місяць під час натискання на стрілочки. При натисканні на конкретну дату викликається DatePicker, в якому можна обрати певний день або проміжок дат.
5. Категорія, що містить назву та кількість витрат в певній валюті. При натисканні на категорію "Додати категорію" відкривається вікно, де можна створити нову категорію.
6. Інтерактивний графік, що показує частку витрат у певних категоріях.
7. Навігація до основного екрану.
8. Навігація до екрану транзакцій.
9. Навігація до екрану категорій.
10. Задати зображення для категорії.
11. Вказати тип валюти.
12. Видалити категорію.
13. Обрати колір для зображення категорії.

3.4 Екран транзакцій

Рисунок 11 Знімок екрана з програми "MoneySaver" (Екран транзакцій та різні додаткові його частини)



1. Пошук транзакцій за допомогою приміток.
2. Розрахунок поточного та кінцевого балансу за певний період.
3. Відображення дати певного списку транзакцій.
4. Відображення транзакції з назвою, зображенням, описом, способом оплати та сумою.
5. Плаваюча кнопка (FloatButton), яка при натисканні відкриває 7-й пункт.
6. Список, що дозволяє легко змінити категорію під час проведення транзакції.
7. Можливість вибору категорії для додавання транзакції.
8. Можливість видалення шляхом змаху (swipe) вліво та редагування шляхом змаху вправо.
9. Відображення спливаючого вікна для редагування транзакцій.

3.5 Екран меню

Рисунок 10. Знімок екрана з програми "MoneySaver" (Меню програми та основні діалогові вікна підменю)

1. Лого додатку .
2. Зміна мови , додаток перекладений на 10 мов.
3. Зміна теми додатку (світла та темна).
4. Сповіщення які нагадують про те що треба додати транзакції.
5. Основна валюта додатку , дані 8 різних валют беруться за допомогою публічного апі.
6. Захист додатку 6 значним кодом , також якщо в телефоні налаштована біометрія , то в додатком можна буде зайти і за допомогою неї
7. Кастомізація топ бару

Висновки

Після закінчення розробки MoneySaver на базі Jetpack Compose/Kotlin, я з впевненістю можу сказати що освоїв розробку застосунків у Android Studio, використовуючи всі можливі її переваги . Також спираючись на минулий досвід розробки Android додатків за допомогою Java/Xml , можу зробити висновок , що Jetpack Compose це майбутнє Android розробки . ЇЇ основні переваги такі як : Декларативний підхід , Перевикористання коду, Інтеграція з Kotlin, Оновлення в реальному часі, Інтеграція з іншими бібліотеками в рази спрощують розробку застосунку , а з MVVM робить програму легко масштабованою.

Отже кінцевим результатом є Мобільний застосунок “MoneySaver” що призначений обліку фінансових витрат на базі Jetpack Compose - Kotlin для ОС Android. В ньому реалізовані весь функціонал що передбачався функціональними вимогами .

Список використаних джерел

1. Announcing Android's modern toolkit [Інтернет стаття] – Режим доступу до ресурсу: https://android-developers.googleblog.com/2021/07/jetpack-compose-announcement.html
2. Android Developers [Електронний ресурс] – Режим доступу до ресурсу: https://android-developers.googleblog.com
3. Run apps on the Android Emulator [Інтернет стаття] – Режим доступу до ресурсу: https://developer.android.com/studio/run/emulator
4. Android's Kotlin-first approach [Інтернет стаття] https://developer.android.com/kotlin/first
5. Getting started with Android Jetpack [Інтернет стаття] – Режим доступу до ресурсу: https://developer.android.com/jetpack/getting-started
6. Understanding Jetpack Compose [Інтернет стаття] – Режим доступу до ресурсу: https://medium.com/androiddevelopers/understanding-jetpack-compose-part-1-of-2-ca316fe39050 time-pickers-in-jetpack-compose-f641b1d72dd5
7. Dependency injection with Hilt [Інтернет стаття] – Режим доступу до ресурсу: https://developer.android.com/training/dependency-injection/hilt-android
8. Detailed Guide on Android Clean Architecture [Інтернет стаття] – Режим доступу до ресурсу: https://medium.com/android-dev-hacks/detailed-guide-on-android-clean-architecture-9eab262a9011
9. Збірка навчальних відео [Інтернет відео збірка] – Режим доступу до ресурсу: https://www.youtube.com/@PhilippLackner/videos