

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіoeлектроніки,
Факультет комп'ютерних наук і технологій
(повне найменування інституту, назва факультету)

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

магістр

(ступінь вищої освіти (освітній ступінь))

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ МОБІЛЬНОГО
ЗАСТОСУНКУ ДЛЯ УПРАВЛІННЯ ОСОБИСТИМИ ДОХОДАМИ ТА
ВИТРАТАМИ
RESEARCH AND SOFTWARE IMPLEMENTATION OF A MOBILE
APPLICATION FOR MANAGING PERSONAL INCOME AND EXPENSES

Виконав: студент(ка) 2 курсу, групи КНТ-219м
Спеціальності 122 Комп'ютерні науки

(код і найменування спеціальності)

Освітня програма (спеціалізація)
Системи штучного інтелекту

Кодочигов А.М

(прізвище та ініціали)

Керівник Корнієнко С.К

(прізвище та ініціали)

Рецензент Неласа Г.В

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування закладу вищої освіти)

Інститут, факультет ІРЕ ФКНТ
Кафедра програмних засобів
Ступінь вищої освіти магістр
Спеціальність 122 Комп'ютерні науки
(код і найменування)
Освітня програма (спеціалізація) Системи штучного інтелекту
(назва освітньої застосунка (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.

С.О. Субботін

“ ” 20 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Кодочигова Артура Михайловича

(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) Дослідження та програмна реалізація мобільного застосунку для управління особистими доходами та витратами. Research and Software Implementation of a Mobile Application for Managing Personal Income and Expenses

керівник проекту (роботи) Корнієнко Сергій Константинович, к.т.н., доцент .

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “03” листопада 2020 року №301

2. Строк подання студентом проекту (роботи) 10 грудня 2020 року

3. Вихідні дані до проекту (роботи) технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Огляд використовуваних систем і інструментів. 3. Проектування структури проекту. 4. Програмна реалізація застосунка. 5. Економіко-організаційна частина. 6. Охорона праці та безпека у надзвичайних ситуаціях. 7. Додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	Корнієнко С.К., доцент		
Економіко-організаційна частина	Пожуєва Т.О., професор		
Охорона праці та безпека у надзвичайних ситуаціях	Коробко О.В., ст. викладач		
Нормоконтролер	Андрєєв М.О., асистент		

7. Дата видачі завдання “4” вересня 2020 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області, вимоги та завдання.	2–3 тижні	Розділ 1
3	Аналіз та вибір моделей для вирішення поставленої задачі	4–5 тижні	Розділ 2
4	Розробка структури застосунка.	6 тиждень	Розділ 3
5	Розробка застосунка.	7–8 тижні	Розділ 4
6	Розробка економіко-організаційної частини	9 тиждень	Розділ 5
7	Розробка заходів з охорони праці та безпеки у надзвичайних ситуаціях.	10 тиждень	Розділ 6
8	Оформлення пояснювальної записки та документів до неї.	11 тиждень	Додатки
9	Нормоконтроль та рецензування.	12-13 тиждень	
10	Підготовка презентації та доповіді	14–15 тижні	
11	Захист роботи.	16 тиждень	

Студент(ка)

(підпис) Кодочигов А.М.
(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис) Корнієнко С.К.
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи магістра:
93 с., 41 рис., 5 табл., 3 дод., 11 джерел.

ОПЕРАЦІЙНА СИСТЕМА ANROID, MPANDROIDCHART,
ANDROID STUDIO, JAVA, SQLITE, MYSQL, GOOGLE PLAY.

Об'єкти дослідження – системи, інструменти та програмні засоби мобільного застосунку для управління особистими доходами та витратами.

Предмет дослідження – операційні системи та інструменти програмування для мобільного застосунку для управління особистими доходами та витратами.

Мета роботи – створення мобільного застосунку для управління особистими доходами та витратами.

Програмні засоби: Android, Android Studio, Java, Materail Design, MPIAndroidChart, PieChart, SQLite, MySQL, startActivityForResult, PutExtra, onActivityResult. CalendarView, RecyclerView, ListView, Google Play.

Результати. Обрана операційна система для базування додатку та інструменти програмування. Створена структура проекту, яка була програмно реалізована.

Висновки. Створено мобільний застосунок для управління особистими доходами та витратами, забезпечено його стабільну роботу, та можливість для подальшого розвитку.

Галузь використання – приватні особи та малий бізнес.

ABSTRACT

Explanatory note to the master's thesis: 93 pp., 41 figures, 5 tables, 3 appendix, 11 sources.

OPERATING SYSTEM ANDROID, MPAndroidChart, Android Studio, Java, SQLite, MySQL, Google Play.

Objects of research - systems, tools and software for mobile applications for personal income and expense management. The purpose of the work is to create a mobile application for managing personal income and expenses.

Subject of research - operating systems and programming tools for mobile application for personal income and expense management.

Materials, methods and technical means: Android, Android Studio, Java, Material Design, MPAndroidChart, PieChart, SQLite, MySQL, startActivityForResult, PutExtra, onActivityResult. CalendarView, RecyclerView, ListView, Google Play.

Results. The operating system for application application and programming tools is selected. The structure of the project was created, which was programmatically implemented.

Conclusions. A mobile application for personal income and expense management has been created, its stable operation and an opportunity for further development have been ensured.

Areas of use - individuals and small businesses.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	8
Вступ.....	9
1 Аналіз предметної області.....	11
2 Огляд використовуваних систем і інструментів.....	12
2.1 Операційна система Android.....	12
2.2 Інтегроване середовище розробки Android Studio.....	13
2.3 Мова програмування Java.....	14
2.4 Material Design.....	15
2.5 Бібліотека MPAndroidChart.....	16
3 Проектування структури проекту.....	18
3.1 Діаграма класів.....	18
3.2 Макет активностей додатку.....	19
3.2.1 Головне вікно. Доходи. Категорії доходів.....	19
3.2.2 Всі доходи. Діаграма доходів. Список доходів по категоріям.....	20
3.2.3 Витрати. Категорії витрат. Баланс.....	20
3.2.4 Всі витрати. Графік витрат. Перелік витрат за категоріями.....	21
4 Програмна реалізація застосунка.....	23
4.1 Структура бази даних.....	23
4.2 Програмна реалізація активностей.....	24
4.2.1 Головне вікно.....	24
4.2.2 Доходи. Категорії доходів.....	25
4.2.3 Витрати. Категорії витрат.....	29
4.2.4 Всі витрати. Діаграма витрат.....	33
4.2.5 Список витрат по категоріям.....	35
4.2.6 Всі доходи. Діаграма доходів. Список доходів по категоріям.....	37
4.2.7 Баланс.....	39
5 Економіко-організаційна частина.....	41
5.1 SWOT-аналіз та визначення пріоритетних груп стейкхолдерів.....	42

5.2 Визначення команди, та життєвого циклу проекту.....	43
5.3 Визначення витрат на розробку.....	45
5.3.1 Розрахунок основної та додаткової заробітної плати	45
5.3.2 Визначення витрат на матеріали	46
5.3.3 Витрати на обладнання.....	47
5.3.4 Накладні витрати.....	49
5.4 Оцінка економічної ефективності	50
6 Охорона праці та безпека у надзвичайних ситуаціях.....	51
6.1 Аналіз потенційних небезпек.....	51
6.2 Заходи з забезпечення безпеки	52
6.3 Заходи з виробничої санітарії та гігієни праці.....	55
6.4 Заходи безпеки у надзвичайних ситуаціях	56
6.4.1 Заходи з пожежної безпеки	56
6.4.2 Заходи з цивільного захисту	58
Висновки	62
Перелік джерел посилання	64
Додаток А Технічне завдання	65
Додаток Б Текст програми	69
Додаток В Слайди презентації.....	85

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

JVM	–Java Virtual Machine;
ОНА	– Open Handset Alliance;
УАН	– українська гривня;
БД	– база даних;
КПО	– коефіцієнт природної освітлен;
ОС	– операційна система;
ПБЕЕ	– правила безпечної експлуатації електроустановок;
ПБЕЕС	– правила безпечної експлуатації електроустановок споживачів;
ПЗ	– програмні засоби;
ПК	– персональний комп'ютер;
П-Па	– приміщення, в яких містяться тверді горючі речовини;
ПУЕ	– правила улаштування електроустановок;
СДОР	– сильнодіючі отруїні речовини;
ХНО	– хімічно небезпечні об'єкти.

ВСТУП

Кожна людина щодня робить великі або дрібні покупки в магазині, в інтернеті. На даний момент є безліч місць, де можна витратити свої гроші. Особливо в тих місцях, де була прикладена рука маркетолога, який знає, на що саме звертає увагу їх потенційний покупець, тим самим підвищуючи ймовірність покупки.

Для деяких людей вести контроль над своїми покупками є дійсно проблемою. У наше століття, щоб зробити покупки, досить пару кліків або прикласти до зчитувального апарату карту - і ти вже став власником певної речі.

Можливо, в даному випадку необхідна дисципліна і ментальні тренування для того, щоб придбати контроль над своїми фінансами. У наше століття при наявності великих мінусів - велика кількість кишить інформації, сторонніх зовнішніх імпульсів, також є велика кількість плюсів.

Наприклад, зараз, щоб розрахувати свій бюджет, визначити поточні витрати і доходи за поточний час, немає необхідності заводити товстелезні зошити для ведення власної бухгалтерії. Як і немає необхідності запом'ятовувати свої витрати і тягати в своїй кишені записну книжку. Зараз на даний момент величезна кількість функцій нам замінює смартфон.

Наприклад, написаний застосунок дозволяє дивитися доходи і витрати за певний час, дозволяє зробити висновок і проаналізувати свій грошовий потік, щоб в подальшому було можливо розпланувати свій бюджет.

Тим самим ми можемо допомогти користувачеві придбати певний контроль над своїми фінансами і бути більш усвідомленим в плані покупок. Якщо ти усвідомлюєш своє фінансове становище на даний момент - це є перший крок до того, щоб почати контролювати і змінювати на краще своє фінансове становище.

Метою випускної кваліфікаційної роботи є розробка застосунку для мобільних пристроїв під управлінням операційної системи Android, яке дозволяє вести персональний фінансовий облік.

Для досягнення поставленої мети були сформульовані наступні завдання:

- вивчення основ Android – Розробки і майданчиків, на яких можливо розробляти Android-додатки;
- формування вимог до функціонала і інтерфейсу застосунку;
- проектування структури проекту;
- програмна реалізація мобільного застосунку.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Кожна людина в своєму житті стикається з щоденними фінансовими питаннями. Полегшити рутинне заняття з ведення персональних грошових потоків покликані сучасні застосунки обліку доходів і витрат – зручні і практичні.

Вести облік фінансів регулярно – то, чого хочуть багато, але виходить лише у деяких. Причини – записи в різних місцях (сьогодні в зошиті, завтра в Ексель, післязавтра в телефоні) або відсутність такої звички в принципі.

Метою роботи є створення функціонального, інтуїтивно зрозумілого та оптимізованого застосунку для управління особистими доходами та витратами.

Якість розробляемого застосунку визначається тим, наскільки він відповідає вимогам, які були закладені на стадії проектування застосунку. Тому були сформульовані наступні вимоги до основного функціоналу застосунка:

- інтуїтивно зрозумілий і зручний інтерфейс;
- створення записів витрат і доходів з можливістю застосування різних атрибутів: дата, категорія, сума, валюта, коментар і ін .;
- редагування / видалення раніше доданих фінансових операцій;
- додавання / видалення / редагування категорій витрат і доходів;
- перегляд доходів і витрат з можливістю вибору часового проміжку часу;
- створення можливості вибору валюти для фінансових операцій;
- перегляд статистики витрат і доходів за допомогою кругових діаграм.

2 ОГЛЯД ВИКОРИСТОВУВАНИХ СИСТЕМ І ІНСТРУМЕНТІВ

2.1 Операційна система Android

Android – це операційна система, спроектована для роботи на смартфонах, планшетах та інших пристроях [1]. Вона заснована на ядрі Linux, введено всього кілька оновлених понять і використовується більшість засобів Linux [2] (віртуальна пам'ять, файлові системи, процеси, ідентифікатори користувачів, планування і тощо). Розроблено альянсом ОНА, який зараз займається підтримкою і подальшим розвитком платформи. Основною мовою програмування серед розробників Android вважається Java , але також є можливість розробляти застосунки та на Delphi , і на C і тощо. У понад 83% смартфонах, проданих в 2019 році, була встановлена операційна система (далі – ОС) Android .

Android відрізняється від інших операційних систем підходом до універсалізації, використання єдиних схем управління компонентами апаратної платформи і взаємодії з кінцевим користувачем. Маючи декілько пристроїв на Android , можна мати на них ідентичний набір програм, контакти, календарні. При цьому користувачеві не потрібно докладати ніяких зусиль.

Одне з головних переваг Android – це відкрита платформа. Вона доступна всім, і будь-який виробник мобільних пристроїв або розробник може використовувати Android в своїх цілях: для створення застосунків, пристроїв і навіть власної операційної системи. Це завжди буде її головною перевагою.

І останнє, але не менш важливе для користувача, перевага – на встановленій на пристроях ОС Android сьогодні можна вибрати телефон з дійсно гігантського списку пропозицій. Дана ОС є відкритою і безкоштовною, тому конкуренція на ринку виробників мобільних пристроїв йде щільно. На Android доступні як смартфони бюджетного рівня, так і продукти елітного класу.

2.2 Інтегроване середовище розробки Android Studio

Android Studio – це інтегроване середовище розробки для створення мобільних застосунків, створена компанією Google в 2013 році [3]. Вона заснована на програмному забезпеченні IntelliJ IDEA від компанії JetBrains, а також є офіційним засобом розробки Android застосунків. Доступна для Windows, OS X і Linux. Офіційні мови програмування для платформи Android: Java, C++ і Kotlin.

За кілька років свого існування у Android Studio було безліч оновлень, доробок, доповнень, і зараз вона є однією з самих кращих середовищ для розробки мобільних застосунків. Головною перевагою суспільством Android Studio вважається вбудований набір засобів розробки, котрий дозволяє створювати додатки для певного пакету програм, використовуючи компоненти, яких немає в інших версіях застосунка – SDK [4]. Також хорошим плюсом є зручний конструктор інтерфейсу.

Android Studio включає в себе велику кількість емуляторів різних видів пристроїв. Це дозволяє розробляти інтерфейс програм, розміщувати компоненти на екрані і переглядати результат на самих різних пристроях. Крім цього в Android Studio зручно і зрозуміло реалізований інтерфейс для роботи, меню структури проекту та пошук помилок. Абсолютно весь інтерфейс легко можна налаштувати, непотрібні вікна і панелі прибрати чи згорнути. Це дуже корисно при створенні об'ємних і високонавантажених застосунків і проектів.

Всі перераховані вище переваги роблять Android Studio найпопулярнішою середовищем розробки мобільних застосунків для операційної системи Android, а також все це допомагає новачкам і навіть досвідченим розробникам при роботі, значно прискорювати продуктивність.

2.3 Мова програмування Java

Java є найбільш поширеною мовою програмування для розробки застосунків під операційною системою Android [5]. Java строго типізована об'єктно-орієнтована мова програмування, розроблена компанією Sun Microsystems (в подальшому яка була придбана компанією Oracle). Дата офіційного випуску – 23 травня 1995 року.

Спочатку мова називалася Oak («Дуб»), розроблявся Джеймсом Гослінгом для програмування побутових електронних пристроїв. Потім його перейменували в Java і стали використовувати для того, щоб писати клієнтські додатки і серверне програмне забезпечення.

Головною перевагою цієї мови є те, що застосунка на ньому транслуються спочатку в байт-код, який не залежить від платформи і виконується віртуальною машиною JVM [6] – програмою, яка після обробки передає інструкції обладнанню як інтерпретатор. В цьому і проявляється його універсальність. Байт-код не залежить від операційної системи і устаткування, що дозволяє Java працювати як звичайна програма, так і безпосередньо в вікнах браузера. Ця здатність може бути застосована, коли код застосунка необхідно адаптувати до різних платформ – Windows , Mac OS, Linux і тощо. Для кожної операційної системи може бути своя віртуальна машина JVM , але кожна з них може виконувати той же код.

Гнучка система безпеки також вважається найважливішою особливістю Java , де виконання застосунка знаходиться під контролем віртуальною машиною. Будь-які операції, що перевищують постановлені повноваження застосунка, викликають негайне переривання роботи застосунка.

2.4 Material Design

Для того щоб програма була зручною і приємною для користувача, вона повинна слідувати принципам Material Design [7]. За допомогою специфікації кольорів Material Design були зроблені стилізовані теми застосунку, де кольори панелі застосунка, стану і контекстні панелі повинні поєднуватися з кольорами заголовків, текст на кнопках і елементами інтерфейсу (checkbox і текстові поля). Рекомендовано застосовувати для яркості значення – 500 для основного кольору і 700 для панелі стану з однієї кольорової палітри. Ще один інструмент для підбору кольорів – це системна палітра, де можна вибрати два кольори і подивитися, як буде виглядати застосунок, і нижче буде відображатися список квітів, які будуть гармонічно виглядати в цій кольоровій палітрі.

Також Material Design дає рекомендації щодо поліпшення розташування елементів екрану і поради за розміром тексту на складових екрану. У специфікації Material Design відображений найкращий досвід по дизайну застосунків. Специфікації представляють собою керівництво по вибору стилю, зображень, анімації і тощо. Для читабельності тексту в розділі по типографії були вказані поради щодо вибору розміру тексту. У керівництві Material Design рекомендовано застосування інтервалів в 8 dp . Вирівнювання компонентів екрану уздовж однієї лінії спрощує читання і роботу з додатком. Так, застосунок буде кастомизований за допомогою колірного дизайну, розстановки відступів і кордонів, а також грамотного розташування об'єктів на екрані.

Material Design по суті є комплексною, кроссплатформенною концепцією, розробленою Google для створення застосунків з візуалізацією і рухомими елементами. Це ціла філософія дизайну для спілкування розробника з користувачем. Дотримуючись вище перерахованим рекомендаціям Material Design , можна створювати додатки з унікальною графікою, красиві, зручні і інтуїтивно зрозумілі інтерфейси .

2.5 Бібліотека MPAndroidChart

MPAndroidChart – це бібліотека для побудови діаграм і графіків з відкритим вихідним кодом для операційної системи Android [8]. Графіки цієї бібліотеки дуже інтерактивні і легкі для створення.

До проекту була додатково підключена бібліотека MPAndroidChart . У файл gradle додана необхідна залежність "com.github.mikephil.charting:charts:3.1.0" . За допомогою цієї бібліотеки були побудовані кругові діаграми в додатку. MPAndroidChart використовувалася для відображення процентного співвідношення за категоріями доходу або витрати. Для цього в XML файл було додано наступне:

```
<com.github.mikephil.charting:charts.PieChart android:id="@+id/chart"
android:layout_width="match_parent" android:layout_height="match_parent">
```

Для вікон зі статистикою використовувалася діаграма PieChart :

```
private void setupPieChart(List<AllOutcome> allOutcomeList){ PieChart
chart = (PieChart)findViewById(R.id.chart); chart.setUsePercentValues(true);
ArrayList<PieEntry> pieEntries = new ArrayList<PieEntry>();
ArrayList<LegendEntry> legendEntries = new ArrayList<LegendEntry>(); String
labelName;
    for (int i = 0; i < allOutcomeList.size(); i++)
        {if(allOutcomeList.get(i).getSum_for_category() > 0.0) {
            labelName = getLabelName(allOutcomeList.get(i).getCategory_name(),
allOutcomeList.get(i).getSum_for_category());

            pieEntries.add(new
PieEntry(getPercent(allOutcomeList.get(i).getSum_for_category()), labelName,
i));

            legendEntries.add(new
LegendEntry(allOutcomeList.get(i).getCategory_name(),
Legend.LegendForm.SQUARE,
15f, 10f, null, ColorTemplate.COLORFUL_COLORS[i]));}}
```

```
PieDataSet dataSet = new PieDataSet(pieEntries, ""); PieData data = new
PieData(dataSet); data.setValueFormatter(new PercentFormatter());
chart.setData(data);
```

```
Description description = new Description(); description.setTextSize(15f);
description.setText(getDescription()); chart.setDescription(description);
```

```
Legend legend = chart.getLegend(); legend.setTextSize(15f);
legend.setFormSize(15f);
    legend.setVerticalAlignment(Legend.LegendVerticalAlignment.TOP);
    legend.setHorizontalAlignment(Legend.LegendHorizontalAlignment.RIGHT);
T);
    legend.setOrientation(Legend.LegendOrientation.VERTICAL);
    legend.setCustom(legendEntries);
```

```
chart.setDrawHoleEnabled(true); chart.setHoleRadius(15f);
chart.setTransparentCircleRadius(15f);
dataSet.setColors(ColorTemplate.COLORFUL_COLORS);
dataSet.setValueTextSize(20); chart.setDrawEntryLabels(true);
chart.setDrawSlicesUnderHole(true); chart.animateY(1000);
chart.invalidate();
```

3 ПРОЕКТУВАННЯ СТРУКТУРИ ПРОЕКТУ

3.1 Діаграма класів

Для створення застосунка в першу чергу була спроектована діаграма класів з метою опису структури проекту, де класи по своїй суті є вікнами застосунку, в яких будуть реалізовуватися описані функції, а виходять з них стрілки – залежності між ними. Кожна стрілка позначає можливість переходу з одного вікна в інше, якщо стрілка двостороння, то в такому випадку перехід здійснюється в обидва боки. Це посприяло тому, що в подальшому, було сформульовано точне уявлення того, що потрібно робити, як це реалізувати і які труднощі можуть з'явитися в подальшому (рис. 3.1).

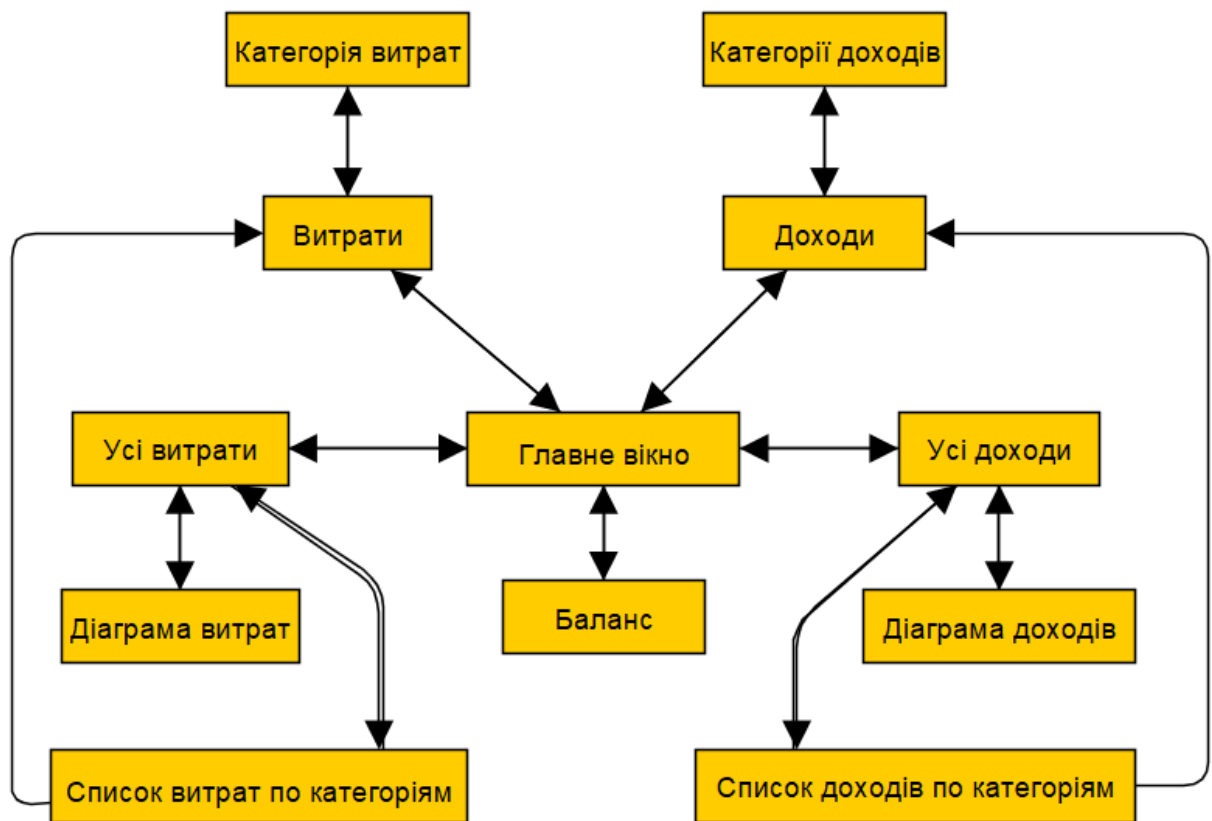


Рисунок 3.1 – Діаграма класів

3.2 Макет активностей додатку

На основі діаграми класів, а також дотримуючись вимог до функціоналу застосунку, був спроектований макет проекту, графічно описано те, як буде виглядати застосунок: його вікна і компоненти. На данній моделі було описано подальше розміщення елементів інтерфейсу застосунку: кнопки, списки, що випадають, поля введення, іконки і тощо.

3.2.1 Головне вікно. Доходи. Категорії доходів

Головне вікно відображає поточні доходи, витрати і баланс, також воно виконує функцію навігації по основним вікнам застосунку.

При переході в розділ «Доходи» відкривається вікно, в якому реалізується додавання інформації про доходи.

Натискаючи на «Категорію» при додаванні доходів, відбувається перехід у вікно «Категория доходів», де відображається список категорій доходу. Тут також можна створити нову категорію або редагувати вже існуючу категорію (рис. 3.2).

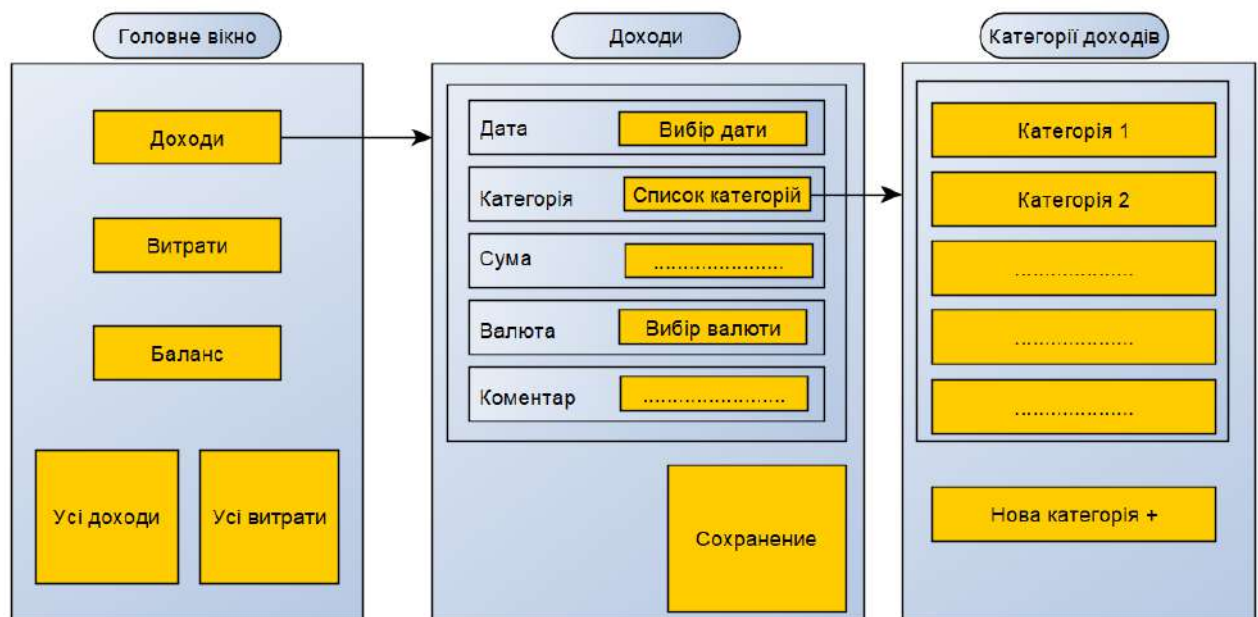


Рисунок 3.2 – Макети головного вікна, доходів і категорій доходів

3.2.2 Всі доходи. Діаграма доходів. Список доходів по категоріям

При натисканні на кнопку «Весь доход» у головному вікні відкриється одноповерхове вікно з інформацією про суму доходу за вказане період часу. Натисканням кнопки “Діаграма” відкриється нове вікно, у якому буде зображено секторну діаграму, яка показуватиме відсоток доходів від суми за категоріями.

Крім того, натиснувши на певну категорію в розділі "Весь доход", ви можете побачити список записів для цієї категорії. Можна повторно редагувати або видаляти записи (рис. 3.3).

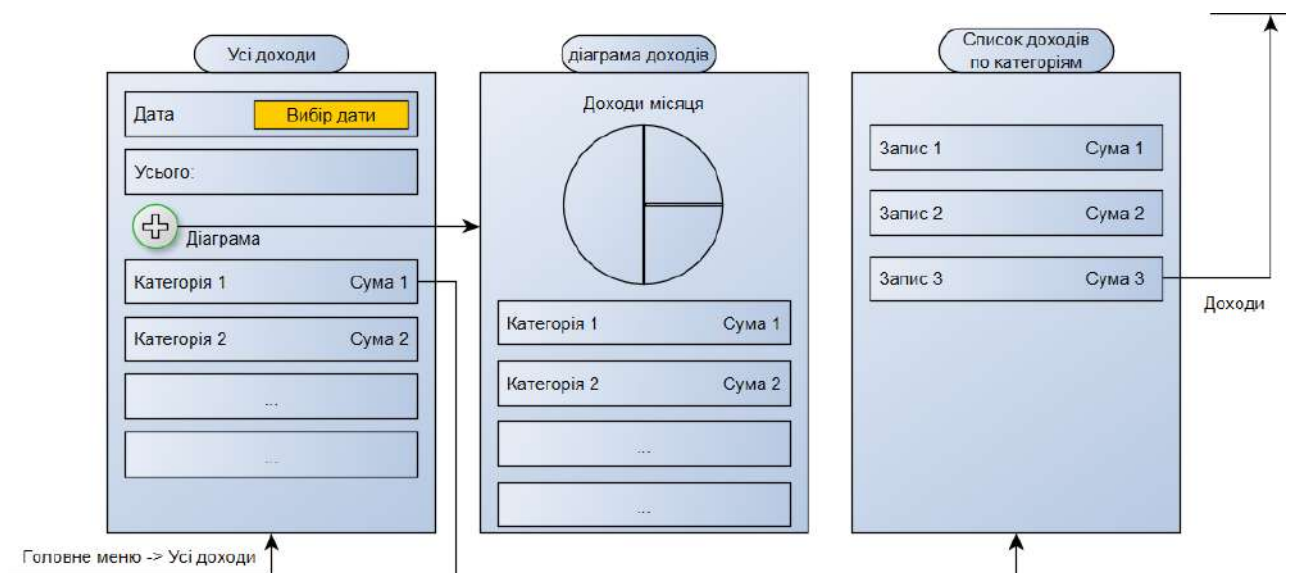


Рисунок 3.3 – Модель всіх доходів, діаграма доходів і список доходів за категоріями

3.2.3 Витрати. Категорії витрат. Баланс

Вибравши розділ "Витрати", відкривається активність з додаванням операцій по витратам. Натиснувши на "Категорию", відбувається перехід у вікно «Категории расходов», де відбивається список категорій, де потрібно вибрати конкретну категорію або створити самому, також є можливість відредагувати або видалити категорії

Коли ви перейдете до розділу «Баланс» у головному вікні, ви побачите щомісячну статистику витрат і доходів разом (рис. 3.4).

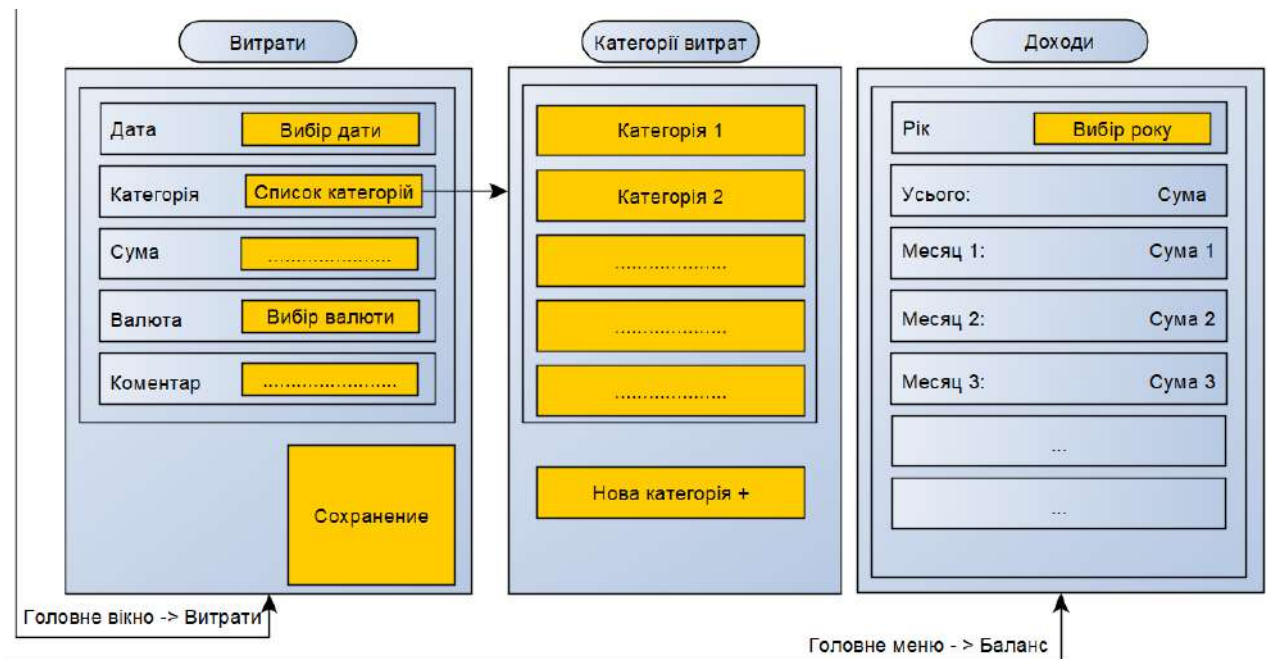


Рисунок 3.4 – Макет витрат, категорій витрат та баланса

3.2.4 Всі витрати. Графік витрат. Перелік витрат за категоріями

При переході в розділ «Всі витрати» з головного вікна, користувач отримує інформацію про витрати за категоріями за вказаний часовий проміжок. Натиснувши на кнопку «Діаграма», відкриється активність «Діаграма расходов», на якій буде представлена кругова діаграма, що представляє собою процентне співвідношення суми за категоріями витрат.

Якщо ви натиснете на категорію у вікні «Всі витрати», у новому вікні буде видно, що ви будете перераховувати витрати на цей тип категорії. Є можливість вносити зміни у записи. (рис. 3.5).

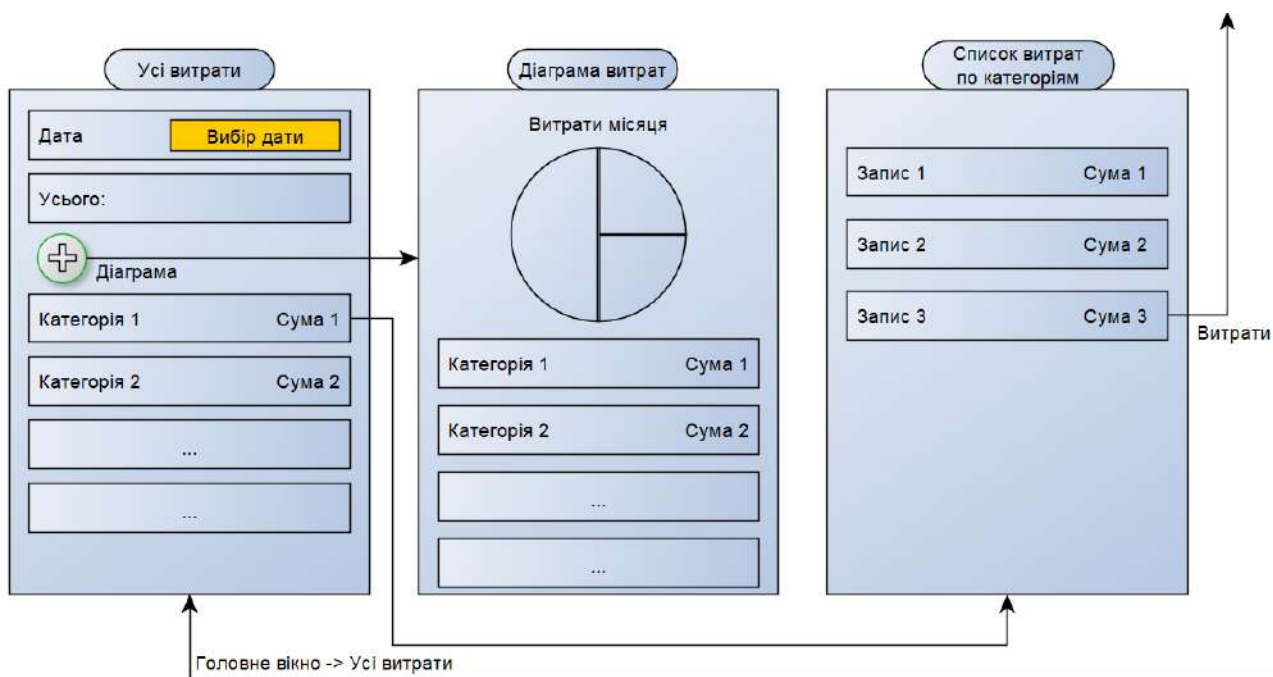


Рисунок 3.5 – Макет всіх витрат, діаграма витрат і список витрат за категоріями

4 ПРОГРАМНА РЕАЛІЗАЦІЯ ЗАСТОСУНКА

4.1 Структура бази даних

База даних – це організована структура, призначена для захоплення, змінення та обробки взаємопов'язаної інформації.

SQLite – це вбудована реляційна база даних з відкритим вихідним кодом. Тобто, він не використовує звичайну модель клієнтської серверної бази даних і не є окремим робочим процесом, наприклад, базою даних MySQL [9]. При такому підході БД SQLite (з усіма таблицями) – це звичайний текстовий файл, який можна розмістити в зручному для нас місці.

База даних SQLite не потребує окремого серверу для запуску. Движок SQLite вбудований безпосередньо в застосунок і вимагає тільки доступу до файлів.

Установка та налаштування нової бази дуже проста, і немає необхідності системним адміністраторам втручатися. Движок повністю вбудований в Android Studio і ніякого додаткового програмного забезпечення не потрібно.

SQLite чудово підходять для застосунків Android, коли основною масою запитів є запити на читання, також дуже добре для невеликих проектів. Оскільки він працює з тим самим запитом, що й бази даних серверного типу, це спрощує перехід до іншої бази даних без внесення кардинальних змін до логіки застосунку.

Для застосунка створено базу даних «Purse.db», яка має чотири таблиці: Income, Income_category, Outcome та Outcome_category.

У «Income» таблиці зберігаються відомості про фінансову операцію кожного користувача за доходами та містять наведені нижче поля.:

- id;
- date (дата);
- category (категорія);
- sum (сума доходу);
- currency (валюта, введенного доходу);

- comment (коментар до операції);
- sum (сума в гривнях).

«Income_category» – таблиця, в якій зберігаються відомості про категорії попереднього діапазону. Містить поля:

- id;
- name (названня категорії).

«Outcome» – таблиця, аналогічна «Income», але зберігає інформацію про фінансові операції, пов'язані з витратами користувачів. Містить наведені нижче поля:

- id;
- date (дата);
- category (категорія);
- sum (сума доходу);
- currency (валюта, введенного доходу);
- comment (коментар до операції);
- sum (сума в гривнях).

«Outcome_category» – таблиця, що зберігає дані категорій витрат.

Таблиця містить наступні поля:

- id;
- name (найменування категорії витрат).

4.2 Програмна реалізація активностей

4.2.1 Головне вікно

Головний екран (рис. 4.1) представляє собою кілька CardView с TextView и ImageButton. Ця активність відповідає за переходи між вікнами при натисканні на іконку із зображенням, реалізація відбувається за допомогою методів startActivityForResult, PutExtra та onActivityResult.

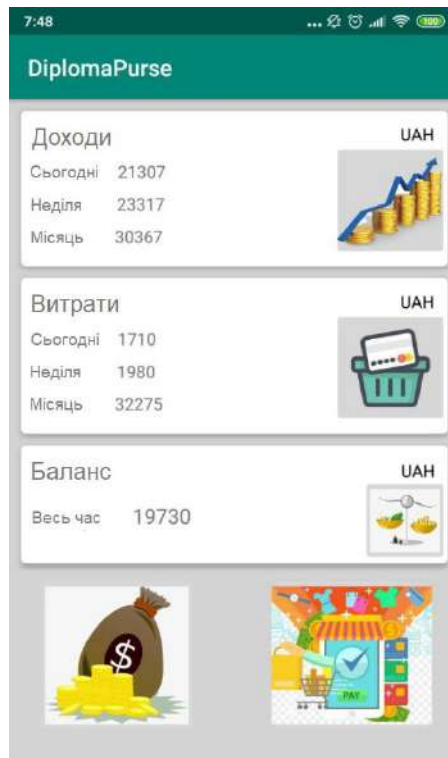


Рисунок 4.1 – Головне вікно

4.2.2 Доходи. Категорії доходів

Додавання такої фінансової операції як дохід, відбувається в активності «Доходи». Пропонується заповнити наступні поля: дата, категорія, сума, валюта, коментар (рис. 4.2).

Рисунок 4.2 – Процес додавання доходу

За замовчуванням дата – це дата, коли був здійснений перехід на цю активність. Зміна дати реалізовано за допомогою компонента `CalendarView` з розділу `Widgets`, який виводить на екран календар (рис. 4.3). Достатньо натиснути на кнопку із зображенням календаря.

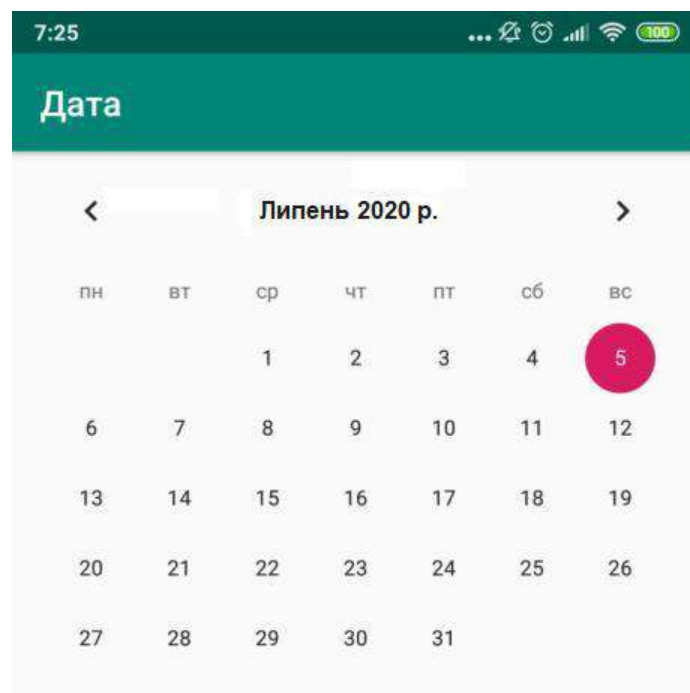


Рисунок 4.3 – Установка дати записи доходу

Для коректного введення суми використовується атрибут

android:inputType="numberDecimal", це здійснює обмеження на введення літерних значень.

Якщо не вибирати валюту, то за замовчуванням вона буде визначена як українська гривня (UAH). Для вибору валюти використовується випадаючий список, з переліком вже наявних валют (рис. 4.4).

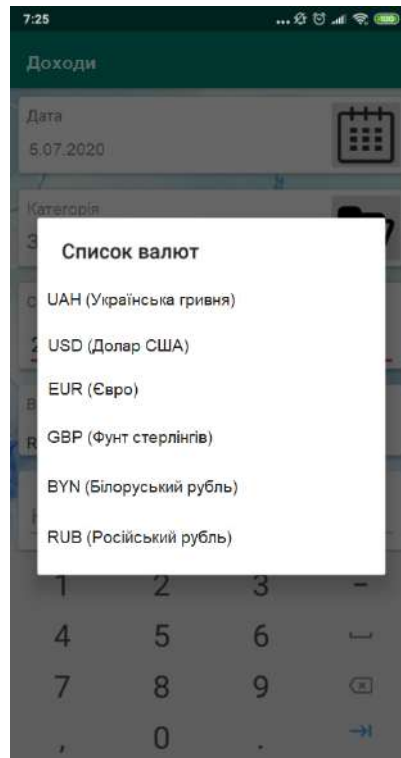


Рисунок 4.4 – Вибір валюти

Коментар є необов'язковим атрибутом при додаванні – за замовчуванням це поле пусте. При введенні опису коментаря кількість знаків не обмежена. Категорія є обов'язковим атрибутом для додавання доходу.

Натискаючи на іконку з папкою, відбувається перехід в активність Категорії доходів. Для зручності користувачів спочатку поля категорій порожні, щоб користувач сам додав категорії, які він вважає потрібними.

У цій активності здійснюється додавання в базу даних назв категорій, які будуть введені користувачем в поле введення. Потім при збереженні відображається введене значення на екрані, це реалізовано компонентом RecyclerView, який спрощує роботу зі списком (рис. 4.5).

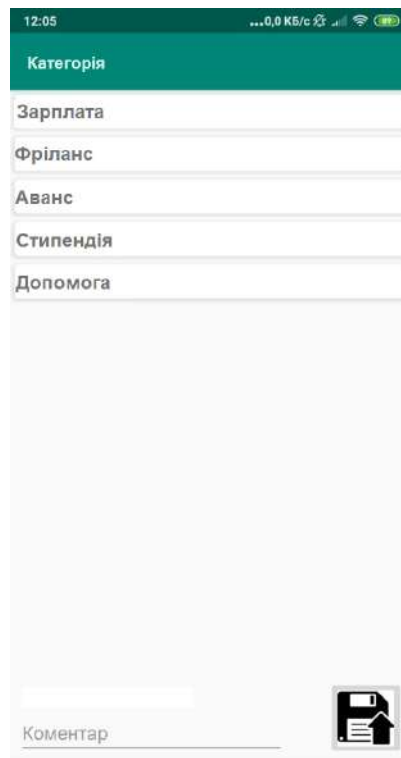


Рисунок 4.5 – Категорії доходів

Для вибору категорії необхідно натиснути на потрібну категорію, після чого відкриється діалогове вікно, в якому необхідно натиснути «Вибрати», реалізоване за допомогою підкласу `AlertDialog`. Можна здійснити редагування, видалення введених категорій також за допомогою `AlertDialog`, яке є розширенням класу `Dialog` (рис. 4.6).

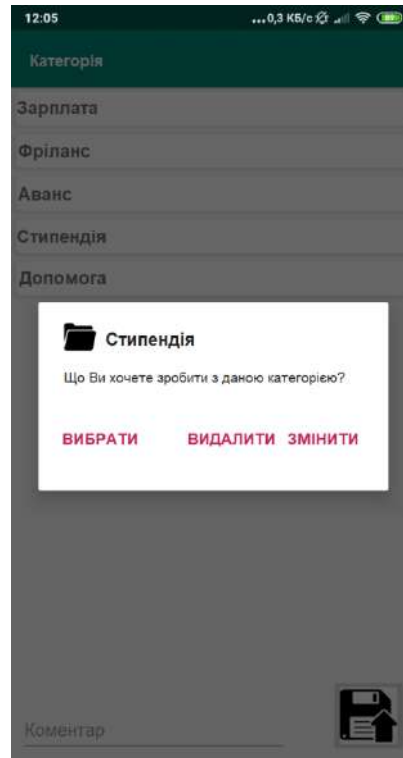


Рисунок 4.6 – Діалогове вікно при натисканні на категорію

Після вибору категорії вікно категорій закривається і відкривається знову вікно доходів. Потім, як завершився введення всіх полів, можна зберегти запис про дохід, натиснувши на кнопку «Зберегти», яка створює новий запис в базі даних записів доходів.

4.2.3 Витрати. Категорії витрат

При переході з головного вікна в розділ «Витрати», відкривається активність «Витрати», де відбувається введення записів про витрати і збереження їх в базу даних.

Додавання витрати передбачає широкий спектр можливих налаштувань. Для коректного введення обов'язковими атрибутами є категорія і сума, решта приймають значення за замовчуванням.

Дата спочатку проставляється поточною датою, для її зміни використовується компонент `CalendarView`, який виводить на екран календар, де необхідно вибрати дату запису витрати.

Здійснено обмеження введення тільки числових значень при додаванні суми, при цьому відкривається клавіатура тільки з числовими значеннями (рис. 4.7).

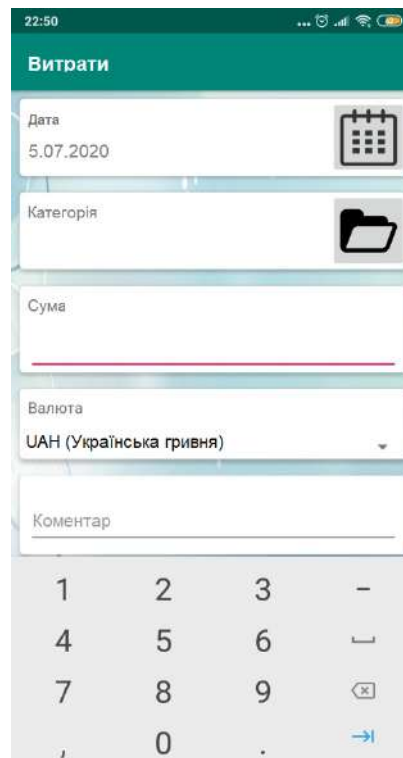
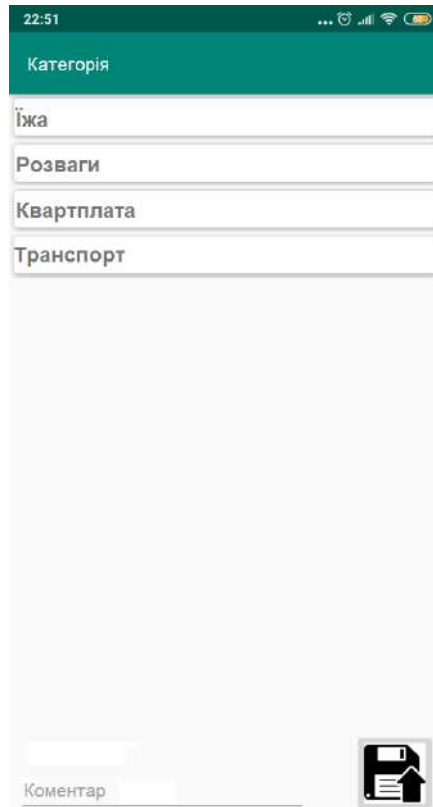


Рисунок 4.7 – Додавання витрати з обмеженням введення для суми

Так як категорія обов'язковий атрибут для додавання витрати, її необхідно заповнити. Для цього перейшовши в активність «Категорія витрат», потрібно вибрати з наданого списку категорій лише одну, за допомогою діалогового вікна. Відображення списку відбувається за допомогою компонента користуальницького інтерфейсу `RecyclerView` (рис. 4.8).



The screenshot shows a mobile application interface with a dark green header bar containing the title "Категорія". Below the header, there are four text input fields with the following labels: "Їжа", "Розваги", "Квартплата", and "Транспорт". At the bottom of the screen, there is a text input field labeled "Коментар" and a button with a document icon and an upward arrow.

Рисунок 4.8 – Категорії витрат

Головною функцією цієї активності є додавання в базу даних назв категорій витрати. До того ж можна змінювати і видаляти раніше введені категорії також за допомогою AlertDialog, який є розширенням класу Dialog (рис. 4.9).

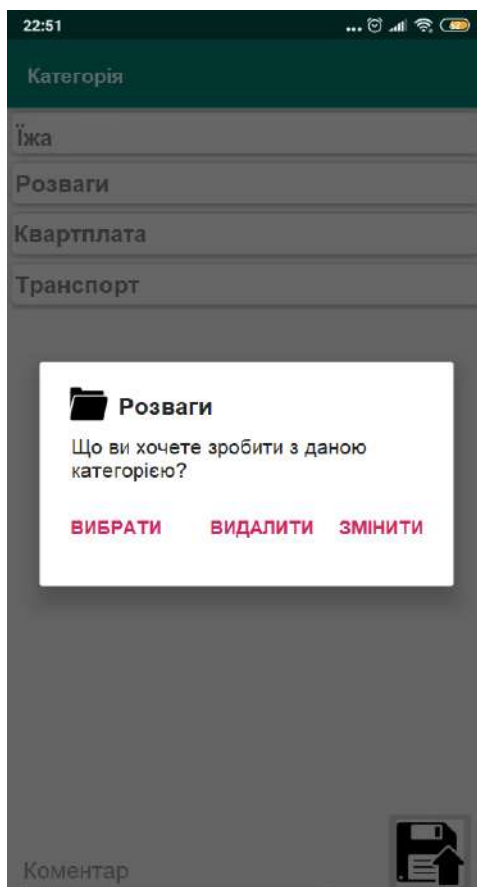


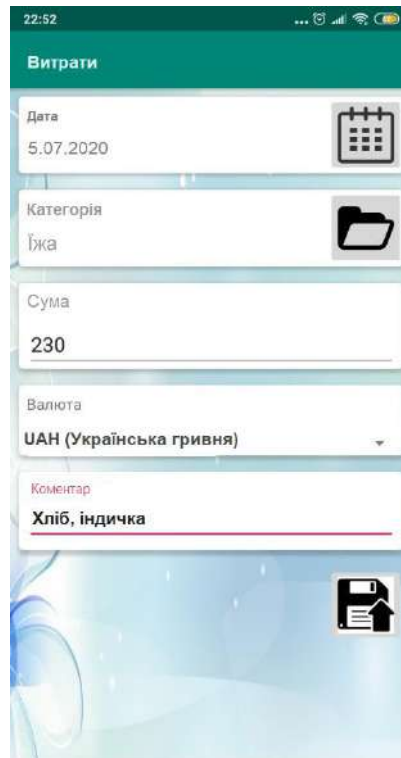
Рисунок 4.9 – Діалогове вікно в категоріях витрати

Після вибору категорії відбувається перехід назад у вікно доходи.

Валюта витрати обирається за замовчуванням гривнею, але її також можна змінити, натиснувши на неї, де відобразиться список, що випадає з уже наявними валютами для вибору.

Є можливість ввести коментар з необмеженою кількістю символів.

Після того як завершено введення значень, можна зберегти запис про витрату, натиснувши на кнопку з іконкою збереження, яка створює новий запис в базі даних записів витрат (рис. 4.10).



22:52

Витрати

Дата
5.07.2020

Категорія
Їжа

Сума
230

Валюта
UAN (Українська гривня)

Коментар
Хліб, індичка

Рисунок 4.10 – Витрата з введенними значеннями

4.2.4 Всі витрати. Діаграма витрат

В активності «Всі витрати» розташовується всі загальні витрати і перелік всіх категорій витрат з витраченої сумою за конкретний часовий проміжок (рис. 4.11). Ці значення вираховуються за допомогою запитів SQLite.



Рисунок 4.11 – Вікно всіх витрат

Часовий період дозволено визначити вручну. Для цього спочатку необхідно вибрати день, тиждень або місяць, а вже потім вибрати від якої дати рахувати вказаний період (рис. 4.12).

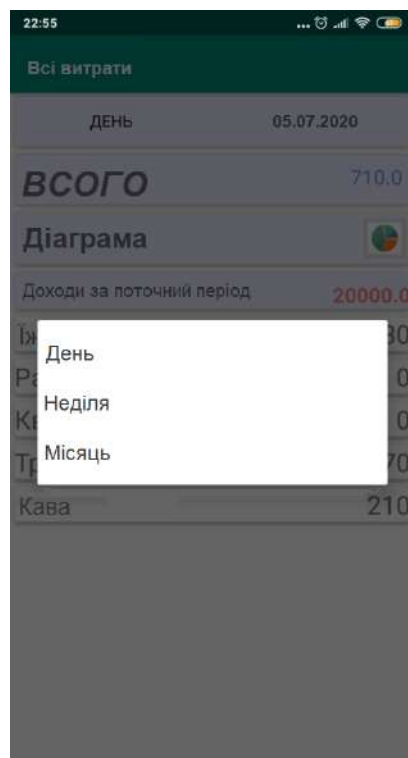


Рисунок 4.12 – Процес вказівки тимчасового періоду всіх витрат

Побачити статистику за категоріями можна натиснувши на іконку діаграми під записом «Всього». Тим самим відкриється нова активність «Діаграма всіх витрат». Створюється кругова діаграма, яка була реалізована завдяки бібліотеці MPAndroidChart. Діаграма відображає статистику, яка формується на основі інформації з бази даних за вказаний часовий період (рис. 4.13).

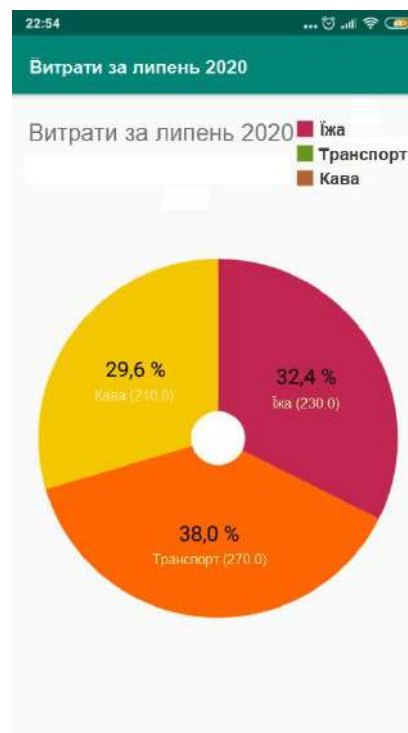
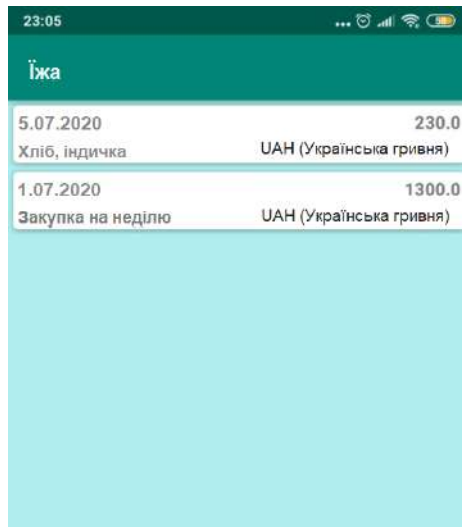


Рисунок 4.13 – Кругова діаграма всіх витрат за день

4.2.5 Список витрат по категоріям

При натисканні на певну категорію у вікні «Всі витрати» відкривається список витрат по цій категорії за допомогою компонента RecyclerView (рис. 4.14).



Їжа		
5.07.2020		230.0
Хліб, індичка	UAH (Українська гривня)	
1.07.2020		1300.0
Закупка на неділю	UAH (Українська гривня)	

Рисунок 4.14 – Список записів витрат по вкаханій категорії

Також є діалогове вікно для вибору редагування або видалення введених записів, реалізоване за допомогою AlertDialog (рис. 4.15). При редагуванні відбувається перехід у вікно Витрати.

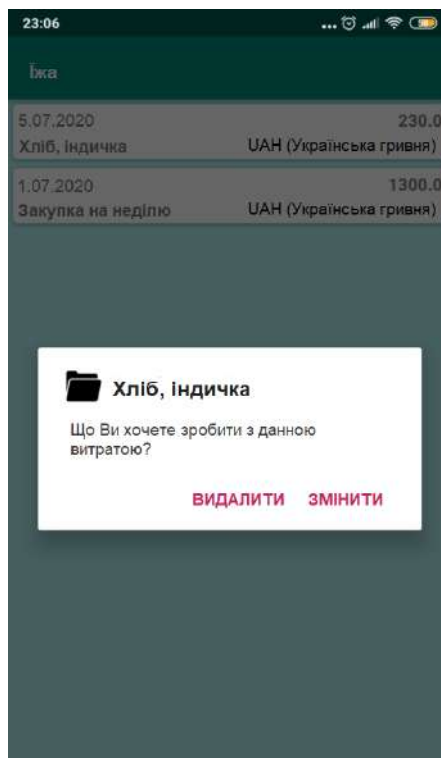


Рисунок 4.15 – Діалогове вікно в списку записів витрат

4.2.6 Всі доходи. Діаграма доходів. Список доходів по категоріям

Активність «Всі доходи» призначена для того, щоб стежити за своїми доходами. За допомогою запитів в цьому розділі відображається загальна сума доходів і список категорій доходів з витрачений сумою за конкретний часовий проміжок (рис. 4.16). Також часовий період можна визначити вручну: необхідно вибрати день, тиждень або місяць, а потім вибрати день, від якої залежить цей вказаний період.



Рисунок 4.16 – Всі доходи за тиждень

Вибравши часовий період, можна побачити статистику, натиснувши на іконку з діаграмою. Внаслідок чого відкриється нове вікно «Діаграма доходів», де буде зображуватися кругова діаграма, що показує процентне співвідношення суми доходів за категоріями (рис. 4.17).

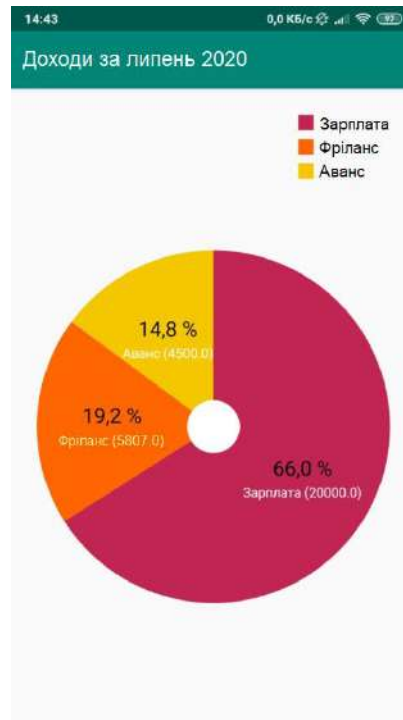
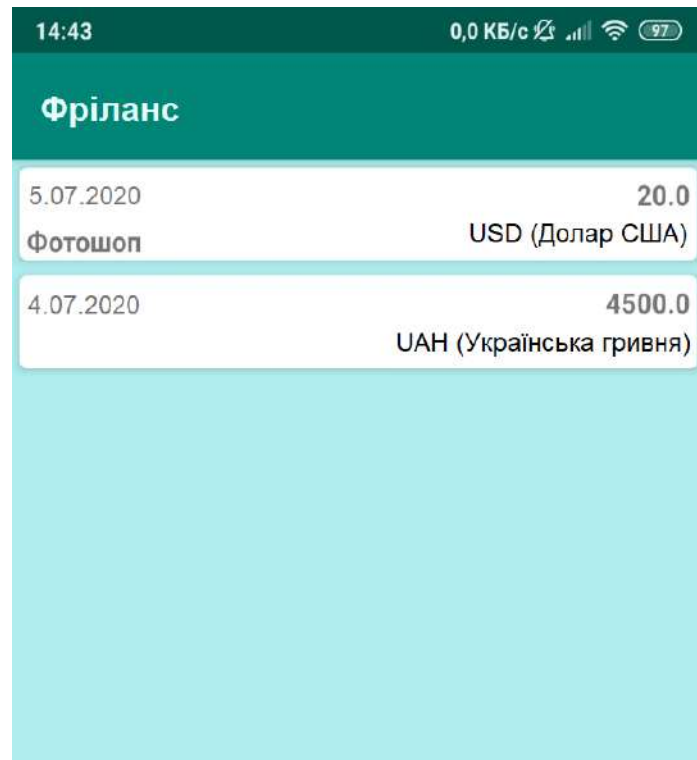


Рисунок 4.17 – Кругова діаграма всіх доходів за тиждень

Побачити список записів доходів потрібної категорії, можна звернувшись до необхідної категорії. У вікні будуть відображатися введені записи для цієї категорії за допомогою компонента RecyclerView (рис. 4.18). Також є можливість ці записи видалити або відредагувати за допомогою підкласу AlertDialog. При зміні відбудеться перехід у вікно доходи, де можна буде внести коригування в запис.

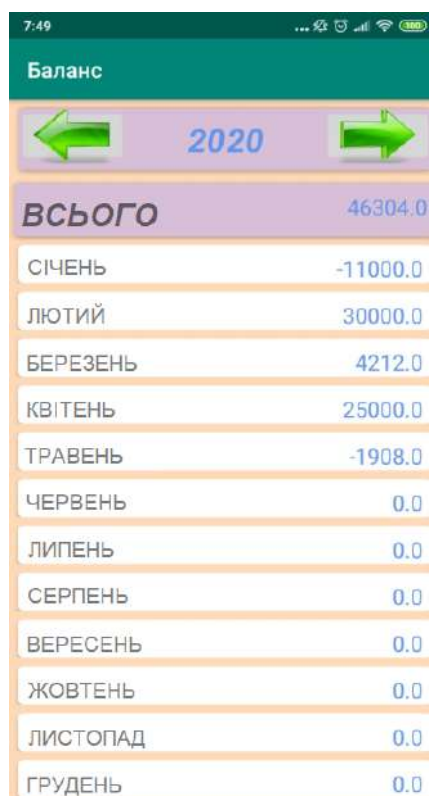


Фріланс	
5.07.2020	20.0
Фотошоп	USD (Долар США)
4.07.2020	4500.0
	UAH (Українська гривня)

Рисунок 4.18 – Список доходів по одній категорії за тиждень

4.2.7 Баланс

Активність відображає стан балансу кожного місяця протягом року за рахунок компонента ListView і запитів. Є можливість вибрати рік, за який треба подивитися статистику (рис. 4.19).



Баланс	
← 2020 →	
ВСЬОГО	46304.0
СІЧЕНЬ	-11000.0
ЛЮТИЙ	30000.0
БЕРЕЗЕНЬ	4212.0
КВІТЕНЬ	25000.0
ТРАВЕНЬ	-1908.0
ЧЕРВЕНЬ	0.0
ЛИПЕНЬ	0.0
СЕРПЕНЬ	0.0
ВЕРЕСЕНЬ	0.0
ЖОВТЕНЬ	0.0
ЛИСТОПАД	0.0
ГРУДЕНЬ	0.0

Рисунок 4.19 – Баланс

5 ЕКОНОМІКО-ОРГАНІЗАЦІЙНА ЧАСТИНА

Було приведено економічне обґрунтування програмної реалізації застосунку для управління особистими доходами та витратами, оцінка затрат на його створення, розрахунок економічної ефективності програмного продукту.

Продукт, що розробляється дозволить користувачам краще контролювати та економити власні кошти, а також поділяти усі свої витрати чи надходження на категорії.

Перспективи втілення ідеї на потенційному ринку представлені в таблиці 5.1

Таблиця 5.1 – Перспективи втілення ідеї на потенційному ринку

Показники стану ринку (найменування)	Характеристика
Головні конкуренти	1Money
	Monefy
	Money Manager Expense & Budget
Динаміка ринку (якісна оцінка)	спадає
Наявність обмежень для входу (вказати характер обмежень)	Обмежень немає

Основний напрям застосунку є малий бізнес або побутові фінансові питання, тому були сформульовані наступні вимоги до основного функціоналу застосунка:

- інтуїтивно зрозумілий і зручний інтерфейс;
- створення записів витрат і доходів з можливістю застосування різних атрибутів: дата, категорія, сума, валюта, коментар і ін .;
- редагування / видалення раніше доданих фінансових операцій;
- додавання / видалення / редагування категорій витрат і доходів;

- перегляд доходів і витрат з можливістю вибору часового проміжку часу;
- створення можливості вибору валюти для фінансових операцій;
- перегляд статистики витрат і доходів за допомогою кругових діаграм.

5.1 SWOT-аналіз та визначення пріоритетних груп стейкхолдерів

Був здійснений SWOT-аналіз, який представлений в таблиці 5.2.

Таблиця 5.2 – SWOT-аналіз

Сильні сторони:	Слабкі сторони:
- Продукція користується попитом ринком - Низькі витрати на реалізацію продукту - Швидка окупність проекту	- відсутність web-сайту - низькі знання на ринку
Можливості:	Загрози:
- Введення платного функціоналу - Можливість удосконалення застосунка	- Є декілька значних конкурентів - Поява нових конкурентів, більш розвинених в технологічному плані

Стейкхолдерами мого проекту є такі групи:

- ініціатор та розробник проекту;
- інвестори;
- споживачі.

Ініціатор та розробник проекту входить до сфери вповноважень та відповідальності, він має прямий вплив на весь процес розробки та реалізації проекту, та має найбільшу заінтересованість в проекті.

Інвестори це потенційний стейкхолдер, який входить до сфери прямого впливу, та які вкладають у розробку свій капітал з певною часткою ризику з метою отримання доходу, вони мають економічний інтерес.

Споживачі входять до сфери опосередкованого впливу, вони зацікавлені в стабільності роботи застосунка, як наслідок фінансової респектабельності розробника, а також вони зацікавлені в максимальному технологічній якості проекту, вони мають економічний та технологічний інтерес.

Виконавши аналіз усіх стейкхолдерів, ключовим були обрані споживачі, так як вони найбільш краще відповідають істотним параметрам, які були визначені на попередньому етапі.

5.2 Визначення команди, та життєвого циклу проекту

Команда складається з одної людини, яка є ініціатором та розробником проекту. На основі цього була створена схема життєвого циклу проекту (рис. 5.1).

Умовні позначки:

- t це тривалість віхи (етапу) життєвого циклу інноваційного проекту;
- T це тривалість життєвого циклу інноваційного проекту;
- D це кожна окрема віха проекту;
- t_1 це аналіз предметної області = 2 дні;
- t_2 це проектування структури застосунка = 7 днів;
- t_3 це розробка схеми функціонування застосунка = 7 днів;
- t_4 це створення програмного коду = 6 днів;
- t_5 це складання програмної документації = 5 днів;
- t_6 це маркетинг = 3 дні.

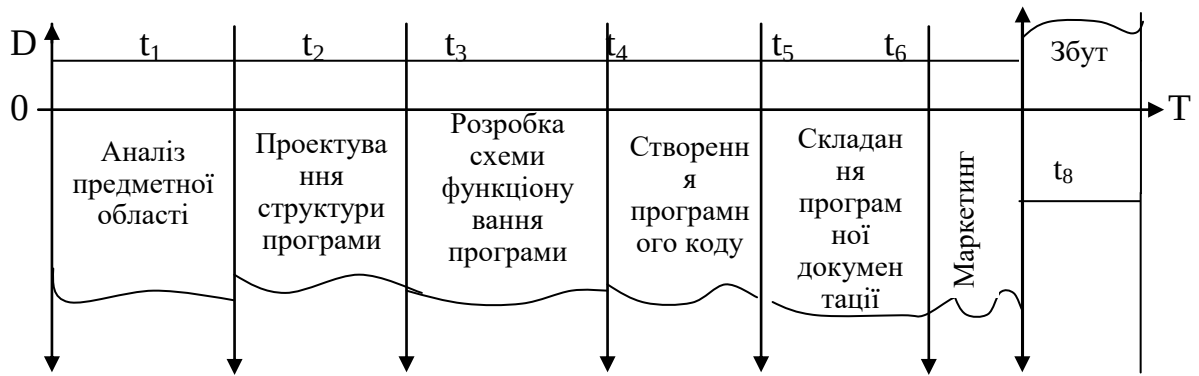


Рисунок 5.1 – Схема життєвого циклу проекту

Тривалість життєвого циклу проекту вираховується за формулою (5.1):

$$t_1 + t_2 + t_3 + t_4 + t_5 + t_6 = T. \quad (5.1)$$

Після підстановки всіх величин тривалості окремої віхи, отримуємо тривалість всього життєвого циклу:

$$2 + 7 + 7 + 6 + 5 + 3 = 30 \text{ днів.}$$

5.3 Визначення витрат на розробку

5.3.1 Розрахунок основної та додаткової заробітної плати

Проводиться розрахунок основної та додаткової заробітної плати, який наведено у таблиці 5.3. Вона заснована на місячному окладі працівників та тривалості, яка біла відпрацьована за планом.

Таблиця 5.3 – Склад, чисельність та фонд заробітної плати виробничих працівників

Категорії працівників	Наявна чисельність, осіб		Тарифна ставка за розрядом виконуваних робіт, грн / годину	Ефективний фонд робочого часу, годин	Тарифний заробіток, грн.	Преміальний відсоток до тарифного заробітку	Розмір премії, грн.	Річний фонд заробітної плати, грн.	ЄСВ, грн.
	за зміну	на добу							
1	2	3	4	5	6	7	8	9	10
Виробничі працівники, в тому числі основний працівник	1	1	31,25	160	5000	10	500	5500	1210
Разом виробничих працівників	1	1	x	x	x	x	500	5500	1210

Додаткова заробітна плата приймається рівною 10% від основної заробітної плати працівника і розраховується за формулою (5.2):

$$ЗП_{\text{дод}} = ЗП_{\text{осн}} \cdot 0,1. \quad (5.2)$$

Після підстановки величини основної заробітної плати в дану формулу, отримується:

$$ЗП_{\text{дод}} = 5000 \cdot 0,1 = 500 \text{ грн.}$$

Відрахування на єдиний соціальний внесок становлять 22% і беруться від основної та додаткової заробітної плати. Вони розраховуються за формулою (5.3):

$$\text{ВІД} = (\text{ЗП}_{\text{осн}} + \text{ЗП}_{\text{дод}}) \cdot 0,22, \quad (5.3)$$

де $\text{ЗП}_{\text{осн}}$ – основна заробітна плата (грн.);

$\text{ЗП}_{\text{дод}}$ – додаткова заробітна плата (грн.).

$$\text{ВІД} = (5000 + 500) \cdot 0,22 = 1210 \text{ грн.}$$

Таким чином отримано, що додаткові соціальні відрахування становлять 1210 грн.

5.3.2 Визначення витрат на матеріали

Витрати на матеріали розраховуються за формулою (5.4):

$$M = \sum_{i=1}^n (\text{Ц}_i N_i (1 + K_{\text{т.з.}}) - \text{Ц}_{io} N_{io}), \quad (5.4)$$

де M – витрати на матеріали, покупні напівфабрикати та комплектуючі (грн.);

$K_{\text{т.з.}}$ – коефіцієнт, що враховує транспортно-заготівельні витрати;

Ц_i – ціна i -го найменування чи комплектуючого (грн.);

N_i – потреба в i -му матеріалі, напівфабрикаті, комплектуючих;

Ц_{io} – ціна зворотних відходів i -го найменування матеріалу (грн.);

N_{io} – кількість зворотних відходів i -го найменування.

n – кількість найменувань матеріалів, напівфабрикатів та комплектуючих.

Додатковими матеріалами, необхідними для розробки, є папір для

оформлення документації, який коштує 100 грн.

5.3.3 Витрати на обладнання

У цю статтю входять витрати на придбання, транспортування, монтаж і налагодження нестандартного обладнання. Також необхідно розрахувати витрати на оплату машинного часу ПК (амортизаційні витрати на обслуговування та вартість спожитої електроенергії).

Розрахуємо витрати на обслуговування ПК. Амортизаційні відрахування розраховуються за формулою (5.5):

$$A = \Phi_B \frac{H_a}{100}, \quad (5.5)$$

де Φ_B – балансова вартість обчислювальної техніки (грн.);

H_a – норма амортизаційних відрахувань на повне відновлення обчислювальної техніки (%).

$$A = 12400 \cdot 0,25 = 3100 \text{ грн.}$$

Амортизаційні відрахування становлять 3100 грн.

Для розрахунку вартості експлуатації обладнання використаємо формулу (5.6):

$$C_\epsilon = N_H \cdot \Phi_{\text{еф}} \cdot K_{\text{зч}} \cdot K_{\text{зп}} \cdot C_e, \quad (5.6)$$

де N_H – номінальна потужність ПК, кВт;

$\Phi_{\text{еф}}$ – річний ефективний фонд часу роботи ЕОМ;

$K_{\text{зч}}$ – середній коефіцієнт завантаження за часом;

$K_{\text{зп}}$ – коефіцієнт завантаження за потужністю;

C_e – ціна одного кВт • год електроенергії, грн/(кВт • год.).

Річний фонд часу роботи ПК становить 900 годин. Середні коефіцієнти завантаження за часом та за потужністю становлять 0.6 та 0.9.

$$C_3 = 0,2 \cdot 900 \cdot 0,9 \cdot 0,6 \cdot 2,11 = 205,09 \text{ грн.}$$

Стаття «Поточний ремонт обладнання» приймається рівною 3% від балансової вартості обладнання і складає 372 грн.

Стаття «Інші витрати» приймається рівною п'яти відсоткам від суми всіх попередніх статей витрат на утримання та експлуатацію устаткування. Сума всіх попередніх статей дорівнює 3777,9 грн., 5% від суми становлять 188,89 грн.

Кошторис витрат на утримання та експлуатацію обладнання наведено у таблиці 5.4.

Таблиця 5.4 – Кошторис витрат на утримання та експлуатацію обладнання

Найменування статей витрат	Сума, грн.
Амортизація обладнання	3100
Експлуатація обладнання (окрім витрат на поточний ремонт)	205,09
Поточний ремонт обладнання	372
Інші витрати	188,89
Разом	3865,98

Витрати на оплату машинного часу ПК для написання й відлагодження застосунка вираховується за формулою (5.8):

$$C_{\text{мс}} = P_{\text{екс}} t_{\text{мч}}, \quad (5.8)$$

де $C_{\text{мс}}$ – витрати на сплату машинного часу (грн);

$P_{\text{екс}}$ – експлуатаційні витрати на один час машинного часу ПК;

$t_{\text{мч}}$ – машинний час ПК для написання та відлагодження програмного засобу.

$$P_{\text{екс}} = \frac{3865,98}{900} = 4,29 \text{ грн/машино-час.}$$

ПК експлуатується 30 днів в одну зміну, що становить у сумі 240 годин. Таким чином витрати на сплату машинного часу становлять:

$$C_{\text{мс}} = 4,29 \cdot 240 = 1029,6 \text{ грн.}$$

5.3.4 Накладні витрати

Накладними витратами називають витрати на загальне управління, загальногосподарські потреби (заробітна плата апарату управління, канцелярські витрати та інше), утримання та експлуатацію будівель. Накладні витрати включають у вартість розробки застосунка непрямым шляхом – у відсотках до основної заробітної плати розробників. В даному випадку накладні витрати становлять 40% до основної заробітної плати розробників, що складає 2000 грн.

Після чого розраховується рентабельність ПЗ, що становить 30% від суми попередніх витрат та ПДВ у розмірі 20% від загальної суми витрат з урахуванням рентабельності.

Результати визначення витрат на розробку застосунка приведені в таблиці 5.5.

Таблиця 5.5 – Вартість розробки застосунка

Стаття витрат	Сума (грн.)	Питома вага до підсумку (%)
Основна заробітна плата	5000	32,57
Додаткова заробітна плата	500	3,25
ЄСВ	1210	7,88

Вартість матеріалів	100	0,65
Витрати на обладнання	1029,6	6,71
Накладні витрати	2000	13,02
Разом	9839,6	64,1
Рентабельність	2951,88	19,23
Разом	12791,48	83,33
ПДВ	2558,296	16,67
Всього	15349,776	100

5.4 Оцінка економічної ефективності

Створення даного застосунку направлено на отримання додаткового прибутку, залучення нових клієнтів і збільшення обсягу послуг.

Витрати на розробку застосунка складають 15349,776 гривень. За рахунок купівлі розроблюваного продукту споживачами очікується продати 500 копій за мінімальною ціною 150 гривень за копію. З урахуванням ризику 50% заплановані надходження від продажів складуть 37500 гривень. Враховуючи, що надходження, з урахуванням ризику, перевищують ціну ПО (на 22150,22 гривні), в якій вже була врахована рентабельність (30%), можна зробити висновок про економічну доцільність розробки даного програмного забезпечення.

6 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА У НАДЗВИЧАЙНИХ СИТУАЦІЯХ

6.1 Аналіз потенційних небезпек

Основними потенційними небезпеками при проведенні роботи в офісі є:

- порушення роботи кістково-м'язового апарату через тривалі статичні навантаження при роботі з ПК;
- нервово-психічні перевантаження внаслідок постійного контакту з клієнтами, колегами по роботі, керівництвом при вирішенні робочих питань, які можуть носити конфліктний характер і призвести до емоційного дискомфорту, внутрішнього роздратування, емоційної нестабільності та захворювань нервової системи;
- незадовільні ергономічні характеристики робочого місця внаслідок нераціонального планування робочого місця, що може призвести до механічних травм, уражень електричним струмом та порушень кістково-м'язового апарату;
- негативний вплив недостатнього освітлення робочої зони на зір та продуктивність роботи працюючого, внаслідок несправності освітлювальних приладів або неправильного проектування освітлювальної системи;
- негативний вплив незадовільних параметрів повітряного середовища робочої зони на здоров'я працюючого, внаслідок неправильного проектування системи вентиляції або несправності її несправності;
- негативний вплив підвищеного рівня шуму на психоемоційний стан працюючого, який пов'язаний з використанням застарілої периферійної техніки, кондиціонерів, копіювальної техніки, освітлювальних приладів;
- небезпека загоряння у зв'язку із несправністю електричного обладнання, недотримання, або порушення правил протипожежної безпеки обслуговуючим персоналом, що може призвести до пожежі;
- неправильні дії персоналу у надзвичайних ситуаціях[10].
-

6.2 Заходи з забезпечення безпеки

Приміщення програмних розробок за «Правилами улаштування електроустановок» (ПУЕ) відноситься до категорії без підвищеної небезпеки ураження електричним струмом. Площа приміщення дорівнює 10,91 м².

Обладнання в цьому приміщенні є споживачем електроенергії, що живиться від змінного струму 220 В від мережі з заземленою нейтраллю, та відноситься до електроустановок до 1000В закритого виконання. За способом захисту людини від ураження електричним струмом відповідає згідно з ГОСТ 12.2.007.0-75* (2001) «ССБТ. Изделия электротехнические. Общие требования безопасности» І (стаціонарні комп'ютери,) та ІІ (освітлювальні прилади, кондиціонери, опалювальні пристрої, ноутбуки, сканери) класу захисту[10].

Згідно «Правилам улаштування електроустановок» (далі «ПУЕ») виконані такі групи заходів з електробезпеки.

Конструктивні заходи забезпечують захист від випадкового дотику до струмопровідних частин за допомогою їх ізоляції та захисних оболонок.

Згідно з ГОСТ 12.1.009-76 (1999) «ССБТ. Электробезопасность. Термины и определения» у приладах ІІ класу захисту використовується подвійна ізоляція - електрична ізоляція, що складається з робочої і додаткової ізоляції.

Так як згідно з НПАОП 40.1-1.32-01 «Правила устройства электроустановок. Электрооборудование специальных установок» кімнатні приміщення у більшості своїй відносяться до класу пожежебезпечної зони.

ІІ-Іа (приміщення, в яких містяться тверді горючі речовини), тому передбачений ступінь захисту ізоляції обладнання ІР44.

Схемно-конструктивні заходи призначені для забезпечення захисту від ураження електричним струмом при дотику до металевих оболонок, які можуть опинитися під напругою в результаті аварій[10].

Згідно з ГОСТ 12.1.030-81 (2001) «ССБТ. Электробезопасность.

Защитное заземление, зануление» у приміщенні де я працював, було влаштоване занулення.

Експлуатація електроустановок і електроустаткування проводиться відповідно до НПАОП 40.1-1.01-97 «Правила безпечної експлуатації електроустановок» (далі «ПБЕЕ») та НПАОП 40.1-1.21-98 «Правила безпечної експлуатації електроустановок споживачів» (далі «ПБЕЕС»).

Для запобігання статистичного навантаження при користуванні ПК робилися перерви в роботі 15 хв. через кожні дві години. Синдром зап'ястного каналу, або тунельний синдром зап'ястя, який може бути наслідком хронічної травми, трапляється у людей внаслідок тривалої роботи з мишею: постійні напруга і здавлювання приводить до мікротравм, здавлювання нерва прилеглими оточуючими тканинами, через що виникає набряк[10].

Щоб тунельний синдром мене не турбував, я дотримувався кількох правил організації робочого місця:

- висота клавіатури від підлоги – 65-75 см;
- наявність ергономічних та зручних миші і клавіатури;
- можливість регулювання нахил і висоту клавіатури (відстань від поверхні стола до середини клавіатури – не більше 30 мм, кут підйому клавіатури – від 2 град до 15 град);
- наявність у клавіатури підставки для рук;
- наявність килимка для миші з захистом від тунельного синдрому (спеціальний виступ забезпечує правильне положення кисті);
- наявність стільця або крісла з підлокітниками.

При роботі з мишкою і клавіатурою я дотримувався певних правила. Коли я набирав текст, рука була зігнута в лікті під прямим кутом (90 град.), а при роботі з мишкою, кисть була прямою і лежала на столі якнайдалі від краю. Час роботи з комп'ютером було обмежено до дійсно необхідного[10].

Щоб попередити тунельний синдром робилися спеціальні вправи для кистей якомога частіше. Ці вправи допомогли поліпшити кровообігу в м'язах

і розміняти їх. Комплекс вправ повторювався приблизно кожні 40 хвилин, тривалість однієї вправи – 1-3 хв.

Нервово напруження впливає на серцево-судинну систему, збільшуючи артеріальний тиск і частоту пульсу, а також на терморегуляцію організму та емоційний стан. Особливу роль у запобіганні втомі відігравали професійний відбір, організація робочого місця, правильне робоче положення, ритм роботи, раціоналізація трудового процесу, використання емоційних стимулів, впровадження раціональних режимів праці і відпочинку тощо. Боротьба зі втомою, в першу чергу, зводиться до покращення санітарно-гігієнічних умов виробничого середовища (ліквідація забруднення повітря, шуму, вібрації, нормалізація мікроклімату, раціональне освітлення тощо).

Для профілактики втоми були застосовані специфічні методи, до яких можна віднести засоби відновлення функціонального стану зорового та опорно-рухового апарату, зменшення гіподинамії, підсилення мозкового кровообігу, оптимізацію розумової діяльності[10].

Загальні ергономічні вимоги встановлено ДСТУ ISO 9241-1:2003.

«Ергономічні вимоги до роботи з відеотерміналами в приміщенні. Частина 1. Загальні положення». Організація робочого місця передбачає: правильне розміщення робочого місця у виробничому приміщенні; вибір ергономічно обґрунтованого робочого положення, виробничих меблів з урахуванням антропометричних характеристик людини; раціональне компонування обладнання на робочих місцях; врахування характеру та особливостей трудової діяльності.

«Правила охорони праці під час експлуатації електронно-обчислювальних машин» рекомендовані розміри столу для робочого місця з ВДТ становлять: висота – 725 мм, ширина – 600-1400 мм, глибина – 800-1000 мм. Мій робочий стіл має такі розміри: висота – 725 мм, ширина – 1200 мм, глибина – 800 мм.

6.3 Заходи з виробничої санітарії та гігієни праці

Згідно ДБН В.2.5-28:2018 «Природне і штучне освітлення» в приміщенні використовувалось природне та штучне освітлення. Природне освітлення здійснено через світлові прорізи, які забезпечували коефіцієнт природної освітленості (КПО) не нижче 1,5%. Для захисту від прямих сонячних променів, які створюють прямі та відбиті відблиски на поверхні екранів і клавіатури, використовувалися сонцезахисні пристрої, на вікнах встановлені жалюзі. Штучне освітлення в приміщенні здійснено системою загального рівномірного освітлення. Значення освітленості на поверхні робочого столу в зоні розміщення документів становить 300-500 лк[10].

Показники мікроклімату в приміщенні відповідають встановленим санітарно-гігієнічним вимогам ДСН 3.3.6-042-99 «Санітарні норми мікроклімату виробничих приміщень», ГОСТ 12.1.005-88 (1991).

«ССБТ. Общие санитарно-гигиенические требования к воздуху рабочей зоны» і ГН 2152-80 «Санітарно-гігієнічні норми допустимих рівнів іонізації повітря виробничих та громадських приміщень».

Були встановлені наступні оптимальні значення параметрів мікроклімату:

- у холодний період року: температура 22-24 °С; відносна вологість: 40- 60%; швидкість переміщення повітря: 0,1 м/с;
- у теплий період року: температура 23-25 °С; відносна вологість: 40- 60%; швидкість переміщення повітря: 0,2 м/с.

Оптимальні рівні позитивних (n+) і негативних (n-) іонів у повітрі приміщення з ПК нормовані згідно ГН 2152-80 «Санітарно-гігієнічні норми допустимих рівнів іонізації повітря виробничих та громадських приміщень» і становили: $n+=1500-30000$ (шт. на 1см^3); $n- = 3000-5000$ (шт. на 1см^3). Підтримка оптимального рівня легких позитивних і негативних аероіонів у повітрі на робочих місцях забезпечено за допомогою біполярних коронних

аероіонізаторів[10].

Дотримання вимог цих документів було досягнуто оснащенням приміщення пристроями кондиціонування і вентиляції, дезодорації повітря, опалювання згідно вимог ДБН В.2.5-67:2013 «Опалення, вентиляція та кондиціонування».

Рівні звукового тиску в октавних смугах частот, рівні звуку та еквівалентні рівні звуку на робочому місці у приміщенні нормуються згідно ДСанПіН 3.3.2.007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» та ДСН 3.3.6.037-99 «Санітарні норми виробничого шуму, ультразвуку та інфразвуку».

Зниження рівня шуму в приміщенні здійснено за допомогою:

- використання якомога сучасного обладнання;
- переведення жорсткого диска в режим сну (Standby), якщо комп'ютер не працює протягом визначеного часу;
- використання блоків живлення ПК з вентиляторами на гумових підвісках.

6.4 Заходи безпеки у надзвичайних ситуаціях

6.4.1 Заходи з пожежної безпеки

Закон України «Про пожежну безпеку» визначає загальні правові, економічні та соціальні основи забезпечення пожежної безпеки на території України, регулює відносини державних органів, юридичних і фізичних осіб у цій галузі незалежно від виду їх діяльності та форм власності[10].

Пожежна безпека – стан об'єкта, при якому з регламентованою ймовірністю виключається можливість виникнення та розвиток пожежі і впливу на людей її небезпечних факторів, а також забезпечується захист матеріальних цінностей.

Для забезпечення пожежної безпеки в установах проводять пожежну

профілактику, яка включає в себе комплекс організаційних і технічних заходів, спрямованих на забезпечення безпеки людей, на запобігання пожежі, обмеження її поширення, а також на створення умов для успішного гасіння пожежі.

Для ліквідації пожежі у початковій стадії її розвитку силами персоналу об'єктів застосовуються первинні засоби пожежогасіння. До них відносяться: вогнегасники, пожежний інвентар (покривала з негорючого теплоізоляційного полотна, ящики з піском, пожежні відра, совкові лопати, ломи, сокири тощо), системи автоматичного пожежогасіння.

Первинні засоби пожежогасіння, в залежності від категорії приміщень, можуть розташовуватись як окремо, так і в складі пожежних щитів[10].

Залежно від агрегатного стану й особливостей горіння різних горючих речовин і матеріалів пожежі за ДБН В.1.1.7-2002 «Пожежна безпека об'єктів будівництва» поділяються на відповідні класи.

В офісному приміщенні знаходиться дерев'яна мебель, електронна апаратура, бумажні носії інформації.

Клас пожежі у офісному приміщенні (згідно із ДБН В.1.1.7-2002 «Захист від пожежі. Пожежна безпека об'єктів будівництва») – пожежі твердих речовин, переважно органічного походження, горіння яких супроводжується тлінням (деревина, пластмаси, папір) – визначається як клас А. Категорія приміщення (згідно із НАПБ Б.03.002-2007 «Норми визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою») – визначається як категорії Д.

Визначення типу та розрахунок кількості первинних засобів пожежогасіння (згідно із ДБН В.1.1.7-2002 «Захист від пожежі. Пожежна безпека об'єктів будівництва») – для адміністративного приміщення площею 10,91 м² було встановлено один порошкових вогнегасниками типу ВП–5 (НАПБ Б.03.002-2007 «Норми визначення категорій приміщень, будинків та

зовнішніх установок за вибухопожежною та пожежною небезпекою». ДБН В.1.1.7-2002 «Пожежна безпека об'єктів будівництва»). Крім цього приміщення було обладнане автоматичними пожежними сповіщувачами, що реагують на підвищення температури, дим, полум'я. Наприклад, сповіщувачі моделей ДТЛ, ІТМ[10].

6.4.2 Заходи з цивільного захисту

Основні норми поведінки та дії при аваріях з викидом сильнодіючих отруйних речовин.

Відмінною особливістю аварії з викидом СДОР є те, що при високих концентраціях хімічних речовин ураження людей може відбуватися в короткі терміни. Тому вирішальне значення в цих умовах має оперативність (швидкість) виконання заходів щодо захисту населення.

Основними заходами захисту персоналу та населення при аваріях є:

- використання індивідуальних засобів захисту і притулків з режимом ізоляції;
- застосування антидотів та засобів обробки шкірних покривів;
- дотримання режимів поведінки (захисту) на зараженій території;
- евакуація людей із зони зараження, що виникла при аварії;
- санітарна обробка людей, дегазація одягу, території, споруд, транспорту, техніки і майна.

Наявність деяких СДОР в повітрі (таких, як хлор, аміак, сірководень і деяких інших) і їх концентрацію можна визначити за допомогою універсального переносного газоаналізатора типу УГ-2.

Персонал і населення, які працюють і проживають поблизу хімічно небезпечних об'єктів, повинні знати властивості, відмінні ознаки і потенційну небезпеку СДОР, використовуваних на даному об'єкті, способи індивідуального захисту від ураження СДОР, вміти діяти при виникненні аварії, надавати першу медичну допомогу ураженим.

Робітники і службовці, почувши сигнал оповіщення, негайно надягають засоби індивідуального захисту, перш за все ізолюючі і промислові протигази. Кожен на своєму робочому місці повинен зробити все можливе для зниження згубних наслідків аварії: забезпечити правильне відключення енергоджерел, зупинити агрегати, апарати, перекрити газові, парові і водяні комунікації відповідно до умов технологічного процесу і правилами техніки безпеки. Потім персонал укривається в підготовлених притулках або виходить із зони зараження. При оголошенні рішення про евакуацію робітники і службовці зобов'язані з'явитися на збірні евакуаційні пункти об'єкта.

Працівники, що входять в невоєнізовані формування, по сигналу про аварію прибувають на пункт збору формування і беруть участь в локалізації та ліквідації осередку хімічного ураження.

Населення, яке проживає поблизу ХНО, при аваріях з викидом СДОР, почувши сигнали оповіщення по радіо (телебаченню), рухомим громковорядним засобів і в інший спосіб, повинні надіти протигази, закрити вікна, відключити електронагрівальні прилади, газ (погасити вогонь в печах), одягнути дітей, взяти необхідне з теплого одягу і харчування (триденний запас псується продуктів), попередити сусідів, швидко, але без паніки вийти з житлового масиву у вказаному напрямку або в сторону, перпендикулярну напрямку вітру, бажано на піднесений, добре провітрюваних ділянку місцевості, на відстань не менше 1,5 км. від попереднього місця перебування, де перебуватиме до отримання подальших розпоряджень.

У разі відсутності протигаза необхідно зробити стрімкий вихід із зони зараження, затримавши дихання на кілька секунд. Для захисту органів дихання можна використовувати підручні вироби з тканин, змочені у воді, хутряні та ватні частини одягу.

При закриванні ними органів дихання знижується кількість вдихуваного газу, а отже і тяжкість ураження.

При русі на зараженій місцевості необхідно дотримуватися таких правил:

- рухатися швидко, але не бігти і не піднімати пилу;
- не притулятися до будівлям і не стосуватися оточуючих предметів;
- не наступати на які по дорозі краплі рідини або порошкоподібні розсипи невідомих речовин;
- не знімати засоби індивідуального захисту до розпорядження;
- при виявленні крапель СДОР на шкірі, одязі, взутті, засобах індивідуального захисту необхідно зняти їх тампоном з паперу, ганчір'я або носовою хусткою;
- по можливості надати необхідну допомогу постраждалим дітям, людям похилого віку, які не здатні рухатися самостійно.

Після виходу із зони зараження потрібно пройти санітарну обробку. Ті, хто отримав незначні поразки (кашель, нудота та інші), звертаються до медичних установ для визначення ступеня ураження і проведення профілактичних і лікувальних заходів.

Перша допомога ураженому хлором полягає в проведенні наступних дій:

- одягання на потерпілого промислового протигаза марки В або ГП-5, ГП-7;
- винесення постраждалого на ношах на територію без зараження і зняття протигаза;
- звільнення від стискує подих одягу;
- при відсутності дихання – штучне дихання, переважно "рот в рот";
- вдихання, для пом'якшення подразнення, аерозолі 0,5% розчину питної соди, а також кисню;
- промивання шкіри і слизових оболонок 2% содовим розчином;
- рясне пиття (тепла вода з содою, чай, кава);
- запобігання самостійного переміщення потерпілого, подальше транспортування тільки в лежачому положенні;

- в холодну пору – відігрівання і забезпечення повного спокою.

Перша допомога ураженому аміаком включає наступні дії:

- одягання на потерпілого промислового протигаза марки К або М, при дуже високих концентраціях – ізолюючого протигаза;
- винесення із зони зараження, зняття протигаза і зараженого одягу;
- при ослабленні або зупинці дихання – штучне дихання, переважно "рот в рот";
- вдихання теплих водяних парів і питво теплого молока;
- при попаданні аміаку в шлунок – виклик штучної блювоти;
- при попаданні аміаку в очі – промивання їх водою;
- при великих опіках – введення знеболюючих засобів і перев'язка;
- забезпечення повного спокою потерпілому, його зігрівання

ВИСНОВКИ

Таким чином, в ході проведеної роботи було розроблено мобільний застосунок під Android для управління особистими доходами і витратами, яке задовольняє цілі створення застосунку.

На фінальному етапі розробки проводилося тестування створеної системи, як єдиного цілого. Даний процес не виявив особливих порушень, так як проміжне тестування під час розробки, безсумнівно, допомогло запобігти їх.

Застосунок добре дає собі раду і виконує основне завдання: дозволяє вести свій бюджет і керувати ним. Застосунок дозволяє додавати всі свої фінансові операції, редагувати і видаляти їх. Можна переглядати статистику за категоріями доходу і витрати з висновком інформації, скільки по даній категорії витрачено. У додатку немає вікових обмежень.

Технічною новизною моєї роботи є те, що я використовував лише безкоштовні інструменти та бібліотеки програмування, які знаходяться у вільному доступі, та це ніяк негативно не вплинуло на результат моєї роботи. Чому я забов'язан особливостям обраної мною операційної системи. Операційна система Android вона є надзвичайно універсальна та відкрита до експериментів, тому дозволяє розробляти проекти на прикладі мого, та при цьому мій застосунок залишається зручним, швидким у використанні та інтуїтивно зрозумілим, завдяки чому він може бути конкурентно спроможним на ринку аналогічних застосунків.

Надалі планується покращувати і модернізувати цю програму за рахунок додавання нового функціоналу:

- поліпшення інтерфейсу;
- створення особистого кабінету;
- курс валют онлайн;
- віджети;
- захист паролем;
- синхронізація на декількох пристроях

Проте, розроблений застосунок в тому вигляді, в якому він є зараз, вже може бути завантажено на майданчик Google Play і поширюватися в вільному доступі.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Android [Електронний ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/Android>.
2. Linux [Електронний ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/Linux>.
3. Среда разработки Android Studio [Електронний ресурс]. – Режим доступа: <https://developer.android.com/?hl=RU/>.
4. Software Development Kit [Електронний ресурс]. – Режим доступа: <https://habr.com/ru/post/430940/>.
5. Учебник Java [Електронний ресурс]. – Режим доступа: <http://proglang.su/java>.
6. Java virtual machine. [Електронний ресурс]. – Режим доступа: <https://topjava.ru/blog/what-is-the-jvm>.
7. Дизайн приложений ОС Android Material Design [Електронний ресурс]. – Режим доступа: <https://developer.android.com/guide/topics/ui/look-and-feel/>.
8. Библиотека MPAndroidChart [Електронний ресурс]. – Режим доступа: <https://github.com/PhilJay/MPAndroidChart/>.
9. Работа с базами данных SQLite [Електронний ресурс]. – Режим доступа: <https://metanit.com/java/android/14.5.php/>.
10. Охорона праці та безпека у надзвичайних ситуаціях [Електронний ресурс]. – Режим доступа: https://zp.edu.ua/sites/default/files/konf/rozdil_dp_specialisty.pdf.
11. SQL as understood by SQLite [Electronic resource]. – Access mode: <https://www.sqlite.org/lang.html>.

ДОДАТОК А

Технічне завдання

Вступ

Назва “Дослідження та програмна реалізація мобільного застосунку для управління особистими доходами та витратами”.

Область застосування: приватні особи.

A.1 Призначення розробки

Мета роботи – створення мобільного застосунку для управління особистими доходами та витратами.

Призначення – створення мобільного застосунку для управління особистими доходами та витратами, для ведення обліку фінансових надходжень і витрат, для подальшої оптимізації бюджету.

Короткий опис – виходячи з вподобань і можливостей клієнта, ця робота дає йому можливість вести облік усіх фінансових операцій по категоріям, датам чи по типу операції.

A.2 Вимоги до застосунка

A.2.1 Вимоги до функціональних характеристик

Функції:

- створення записів витрат і доходів з можливістю застосування різних атрибутів: дата, категорія, сума, валюта, коментар і ін .;
- редагування / видалення раніше доданих фінансових операцій;
- додавання / видалення / редагування категорій витрат і доходів;
- перегляд доходів і витрат з можливістю вибору часового проміжку часу;
- створення можливості вибору валюти для фінансових операцій;
- перегляд статистики витрат і доходів за допомогою кругових діаграм.

Вхідні параметри – можливості та вподобання користувача: бюджет, бажані параметри та функції майбутнього застосунку.

Вихідні параметри – застосунок для управління особистими доходами та витратами.

A.2.2 Вимоги до надійності

Застосунок має мати обмежені права щодо використання функцій і даних, які зберігаються на пристрої користувача.

A.2.3 Умови до експлуатації

Застосунком може користуватись будьякий користувач, який має пристрій на базі Android від версії 6.0.1 Marshmallow .

A.2.4 Вимоги до складу і параметрів технічних засобів

Системні вимоги до пристрою на якому можна експлуатувати даний застосунок:

- операційна система Android 6.0.1 або більш нової версії;
- не менш 256 Мб оперативної пам'яті;
- 5-10 МБ вільного місця на пристрої;
- розширення екрану не менше 649x360 точок.

A.2.5 Вимоги до інформаційної і програмної сумісності

Для визначення коректного результату на виході та вході повинні бути вподобання і можливості, що чутко відповідають правилам обраної моделі.

A.3 Стадії й етапи розробки

Необхідно провести наступні етапи при створенні застосунку для управління особистими доходами та витратами.

1. Аналіз предметної області та визначення систем та інструментів розробки застосунку.
2. Формування вимог до функціонала і інтерфейсу застосунку.
3. Проектування структури проекту.
4. Вибір бібліотек та методів.
5. Програмна реалізація застосунку.

ДОДАТОК Б
Текст програми

MainActivity.java

```
package com.example.habik.diplomapurse; import ...
public class MainActivity extends AppCompatActivity implements
View.OnClickListener {
    TextView currencyBalance, currencyIncome, currencyOutcome,
    allTimeBalanceSum, incomeSumToday, incomeSumWeek, incomeSumMonth,
    outcomeSumToday, outcomeSumWeek, outcomeSumMonth;

    ImageButton incomeImgBtn, outcomeImgBtn, balanceImgBtn, allIncomeImgBtn,
    allOutcomeImgBtn;

    DBHelper dbHelper; SQLiteDatabase database;

    DateFormat df = new SimpleDateFormat("d.MM.yyyy");
    String mydate = df.format(Calendar.getInstance().getTime()); int dayofweek =
    Calendar.getInstance().get(Calendar.DAY_OF_WEEK);

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState); setContentView(R.layout.activity_main);

        dbHelper = new DBHelper(this);
        database = dbHelper.getWritableDatabase(); dbHelper.close();

        incomeSumToday = findViewById(R.id.income_sum_today); incomeSumWeek =
        findViewById(R.id.income_sum_week); incomeSumMonth =
        findViewById(R.id.income_sum_month);

        outcomeSumToday = findViewById(R.id.outcome_sum_today);
        outcomeSumWeek = findViewById(R.id.outcome_sum_week);
        outcomeSumMonth = findViewById(R.id.outcome_sum_month);

        allTimeBalanceSum = findViewById(R.id.all_time_balance_sum);

        currencyBalance = findViewById(R.id.currency_balance); currencyIncome =
        findViewById(R.id.currency_income); currencyOutcome =
        findViewById(R.id.currency_outcome);

        incomeImgBtn = findViewById(R.id.income_imgBtn);
```

```

incomeImgBtn.setOnClickListener(this); outcomeImgBtn =
findViewById(R.id.outcome_imgBtn);
outcomeImgBtn.setOnClickListener(this); balanceImgBtn =
findViewById(R.id.balance_imgBtn); balanceImgBtn.setOnClickListener(this);
allIncomeImgBtn = findViewById(R.id.allIncome_imgBtn);
allIncomeImgBtn.setOnClickListener(this); allOutcomeImgBtn =
findViewById(R.id.allOutcome_imgBtn);
allOutcomeImgBtn.setOnClickListener(this);
incomeSumToday.setText(String.valueOf(getTodayBalance(PurseContract.Income
Entry.TABLE_INCOME)));
incomeSumWeek.setText(String.valueOf(getWeekBalance(PurseContract.IncomeE
ntry.TABLE_INCOME)));
incomeSumMonth.setText(String.valueOf(getMonthBalance(PurseContract.Incom
eEntry.TABLE_INCOME)));
outcomeSumToday.setText(String.valueOf(getTodayBalance(PurseContract.Outco
meEntry.TABLE_OUTCOME)));
outcomeSumWeek.setText(String.valueOf(getWeekBalance(PurseContract.Outcom
eEntry.TABLE_OUTCOME)));
outcomeSumMonth.setText(String.valueOf(getMonthBalance(PurseContract.Outc
omeEntry.TABLE_OUTCOME)));
allTimeBalanceSum.setText(String.valueOf(getBalanceAllTime(PurseContract.Inc
omeEntry.TABLE_INCOME,
PurseContract.OutcomeEntry.TABLE_OUTCOME))); dbHelper.close();
}

```

```

@Override
protected void onResume() { super.onResume();
incomeSumToday.setText(String.valueOf(getTodayBalance(PurseContract.Income
Entry.TABLE_INCOME)));
incomeSumWeek.setText(String.valueOf(getWeekBalance(PurseContract.IncomeE
ntry.TABLE_INCOME)));
incomeSumMonth.setText(String.valueOf(getMonthBalance(PurseContract.Incom
eEntry.TABLE_INCOME)));
outcomeSumToday.setText(String.valueOf(getTodayBalance(PurseContract.Outco
meEntry.TABLE_OUTCOME)));
outcomeSumWeek.setText(String.valueOf(getWeekBalance(PurseContract.Outcom
eEntry.TABLE_OUTCOME)));
outcomeSumMonth.setText(String.valueOf(getMonthBalance(PurseContract.Outc
omeEntry.TABLE_OUTCOME)));
}

```

```

@Override
public void onClick(View view) { switch (view.getId()) {
case R.id.income_imgBtn: {
Intent intent = new Intent(MainActivity.this, IncomeActivity.class);

```

```

intent.putExtra("date_income", mydate); startActivity(intent);
break;
}

```

```

case R.id.outcome_imgBtn: {
Intent intent = new Intent(MainActivity.this, OutcomeActivity.class);
intent.putExtra("date_outcome", mydate); startActivity(intent);
break;
}

```

```

case R.id.balance_imgBtn: {
Intent intent = new Intent(MainActivity.this, BalanceActivity.class);
startActivity(intent); break;
}

```

```

case R.id.allIncome_imgBtn: {
Intent intent = new Intent(MainActivity.this, AllIncomeActivity.class);
startActivity(intent); break;
}

```

```

case R.id.allOutcome_imgBtn: {
Intent intent = new Intent(MainActivity.this, AllOutcomeActivity.class);
startActivity(intent); break;
}
}
}

```

```

int getTodayBalance(String table_name){ int sum = 0;
DateFormat df = new SimpleDateFormat("d.MM.yyyy"); String date_search =
df.format(Calendar.getInstance().getTime()); Cursor cursor;
database = dbHelper.getReadableDatabase(); cursor = database.query(table_name,
new
String[]{"sumInUAH"}, "date =?", new String[]{date_search}, null, null, null);
//cursor = database.rawQuery("SELECT sum_in_UAH AS SUM FROM
table_name WHERE date = date_search;", null);
if (cursor.moveToFirst()) {
sum = sum + cursor.getInt(0); while (cursor.moveToNext()) {
sum = sum + cursor.getInt(0);
}
} else
sum += 0; cursor.close(); database.close(); return sum;
}

```

```

int getWeekBalance(String table_name) { int sum = 0;

```

```

String date_search;
Calendar c = Calendar.getInstance();
DateFormat df = new SimpleDateFormat("d.MM.yyyy"); Cursor cursor;

database = dbHelper.getReadableDatabase();

for (int i = dayofweek == 1 ? 7 : dayofweek; i > 1; i--)
{
    date_search = df.format(c.getTime()); cursor = database.query(table_name, new
    String[]{PurseContract.IncomeEntry.INCOME_SUM_IN_UAH}, "date
    =?", new String[]{date_search}, null, null, null);
    //cursor = database.rawQuery("SELECT sum_in_UAH AS SUM FROM
    table_name WHERE date = date_search;", null);
    if (cursor.moveToFirst()) {
        sum = sum + cursor.getInt(0); while (cursor.moveToNext()) {
        sum = sum + cursor.getInt(0);
        }
    } else
    sum += 0; cursor.close();
    c.add(Calendar.DAY_OF_MONTH, -1);
}
database.close(); return sum;
}

private int getMonthBalance(String table_name) { int sum = 0;
DateFormat df = new SimpleDateFormat("MM.yyyy"); String date_search = "%"
+
df.format(Calendar.getInstance().getTime()); Cursor cursor;
database = dbHelper.getReadableDatabase(); cursor = database.query(table_name,
new
String[]{PurseContract.IncomeEntry.INCOME_SUM_IN_UAH}, "date LIKE ?",
new String[]{date_search}, null, null, null);
//cursor = database.rawQuery("SELECT sum_in_UAH AS SUM FROM
table_name WHERE date = date_search;", null);
if (cursor.moveToFirst()) {
    sum = sum + cursor.getInt(0);
    while (cursor.moveToNext()) {
        sum = sum + cursor.getInt(0);
    }
} else
sum += 0; cursor.close(); database.close(); return sum;
}

private int getBalanceAllTime(String table_first, String table_second)
{
    int balance = 0; Cursor cursor;

```

```

database = dbHelper.getReadableDatabase(); cursor =
database.query(table_first,new
String[]{PurseContract.IncomeEntry.INCOME_SUM_IN_UAH},null,null,
null,null,null);
if (cursor.moveToFirst()) {
balance = balance + cursor.getInt(0);
} else
balance += 0; cursor.close();
cursor = database.query(table_second,new
String[]{PurseContract.OutcomeEntry.OUTCOME_SUM_IN_UAH},null,nul
l,null,null,null);
if (cursor.moveToFirst()) {
balance = balance - cursor.getInt(0);
}
cursor.close(); database.close(); return balance;    }
}

```

PurseContract.java

```

package com.example.habik.diplomapurse; import android.provider.BaseColumns;
import java.util.HashMap;
import java.util.Map; public class PurseContract {
public static final      String TYPE_TEXT="TEXT"; public static final String
TYPE_INTEGER = "INTEGER"; public static final String TYPE_REAL="REAL";
public static          final class IncomeCategoriesEntry implements
BaseColumns {
public static final String TABLE_INCOME_CATEGORY = "categoriesIncome";
public static final String INCOME_CATEGORY_NAME = "name";
public static final String CREATE_INCOME_CATEGORY = "CREATE TABLE
IF NOT EXISTS " + TABLE_INCOME_CATEGORY + "(" + _ID + " " +
TYPE_INTEGER + " PRIMARY KEY AUTOINCREMENT, " +
INCOME_CATEGORY_NAME + " "
+ TYPE_TEXT + ")";
public static final String DROP_INCOME_CATEGORY = "DROP TABLE IF
EXISTS " + INCOME_CATEGORY_NAME;
}
public static final class OutcomeCategoriesEntry imple ments BaseColumns
{
public static final String TABLE_OUTCOME_CATEGORY =
"categoriesOutcome";
public static final String OUTCOME_CATEGORY_NAME =
"name";

public static final String TYPE_TEXT="TEXT"; public static final String
TYPE_INTEGER = "INTEGER"; public static final String

```

```

CREATE_OUTCOME_CATEGORY= "CREATE TABLE IF NOT EXISTS "+
TABLE_OUTCOME_CATEGORY + "("+_ID + " " + TYPE_INTEGER + "
PRIMARY KEY AUTOINCREMENT, " + OUTCOME_CATEGORY_NAME + "
" + TYPE_TEXT + ")";
public static final String DROP_OUTCOME_CATEGORY = "DROP TABLE IF
EXISTS " + OUTCOME_CATEGORY_NAME;
}
public static final class OutcomeEntry implements BaseColumns
{

public static final String TABLE_OUTCOME = "Outcome"; public static    final
String OUTCOME_DATE = "date"; public static    final String
OUTCOME_CATEGORY = "category";
public static final String OUTCOME_SUM = "sum";
public static final String OUTCOME_CURRENCY = "currency";
public static final String OUTCOME_COMMENT = "comment"; public static final
String OUTCOME_SUM_IN_UAH = "sumInUAH";
public static final String CREATE_OUTCOME = "CREATE TABLE IF NOT
EXISTS "+ TABLE_OUTCOME + "("+_ID + " " + TYPE_INTEGER + "
PRIMARY KEY AUTOINCREMENT, " + OUTCOME_DATE + " "
+ TYPE_TEXT + ", " + OUTCOME_CATEGORY + " " + TYPE_TEXT + ", " +
OUTCOME_SUM + " " + TYPE_INTEGER + ", " + OUTCOME_CURRENCY + "
" + TYPE_TEXT + ", " + OUTCOME_COMMENT + " " + TYPE_TEXT + ", " +
OUTCOME_SUM_IN_UAH + " " + TYPE_REAL + ")";
public static final String DROP_OUTCOME = "DROP TABLE IF EXISTS " +
TABLE_OUTCOME;
}
public static final class IncomeEntry implements BaseColumns
{
public static final String TABLE_INCOME = "Income"; public static    final String
INCOME_DATE = "date"; public static final String INCOME_CATEGORY =
"category";

public static final String INCOME_SUM = "sum";
public static final String INCOME_CURRENCY = "currency";
public static final String INCOME_COMMENT = "comment";
public static final String INCOME_SUM_IN_UAH = "sumInUAH";
public static final String CREATE_INCOME = "CREATE TABLE IF NOT
EXISTS "+ TABLE_INCOME + "("+_ID + " " + TYPE_INTEGER + " PRIMARY
KEY AUTOINCREMENT, " + INCOME_DATE + " " + TYPE_TEXT + ", " +
INCOME_CATEGORY + " " + TYPE_TEXT + ", " + INCOME_SUM + " " +
TYPE_INTEGER + ", " + INCOME_CURRENCY + " " + TYPE_TEXT + ", " +
INCOME_COMMENT + " " + TYPE_TEXT + ", " + INCOME_SUM_IN_UAH + "
" + TYPE_REAL + ")";

```

```

public static final String DROP_INCOME = "DROP TABLE IF EXISTS " +
TABLE_INCOME;
}
}

```

IncomeActivity.java

```

package com.example.habik.diplomapurse.Income; import ...
public class IncomeActivity extends AppCompatActivity implements
View.OnClickListener {
final int REQUEST_CODE_DATA = 0; final int
REQUEST_CODE_CATEGORY = 1;
TextView incomeDate,categoryText;
ImageButton dateImgBtn, categoryImgBtn, saveImgBtn; EditText incomeSum,
incomeComment;
TextInputLayout textInputLayout;

Map<String,Double> currencyDict = new HashMap<String, Double>();
private String[] currencies = {"UAH (українська гривня)", "USD (Долар
США)", "EUR (Євро)", "GBP (Фунт стерлінгів)", "BYN (Белоруський рубль)",
"RUB (Російський рубль)"};
String currency; double sumInUAH = 0.0; int index = 0;
boolean isEdited = false;

DBHelper dbHelper; SQLiteDatabase database;

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState); setContentView(R.layout.activity_income);
incomeDate = findViewById(R.id.date_income); dateImgBtn =
findViewById(R.id.date_imgBtn); dateImgBtn.setOnClickListener(this);
categoryText = findViewById(R.id.category);
categoryImgBtn = findViewById(R.id.category_imgBtn);
categoryImgBtn.setOnClickListener(this);
saveImgBtn = findViewById(R.id.btn_save);
saveImgBtn.setOnClickListener(this); incomeSum = findViewById(R.id.sum);
incomeComment = findViewById(R.id.comment); Intent intent = getIntent();
String date = intent.getStringExtra("date_income"); incomeDate.setText(date);
textInputLayout = findViewById(R.id.text_input_layout); incomeComment =
(EditText) textInputLayout.findViewById(R.id.comment);
textInputLayout.setHint(getString(R.string.comment));
ArrayAdapter<String> currencyAdapter = new ArrayAdapter<String>(this,
android.R.layout.simple_spinner_item, currencies);

```

```
currencyAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
```

```
currencyDict.put("UAH (Українська гривня)",1.0); currencyDict.put("USD (Долар США)",28.37); currencyDict.put("EUR (Євро)",33.70);
currencyDict.put("GBP (Фунт стерлінгів)",37.86); currencyDict.put("RUB (Російський рубль)",0.38); currencyDict.put("BYN (Белоруський рубль)",11.10); Spinner spinnerCurrency = (Spinner)
findViewById(R.id.currencies_spinner);
spinnerCurrency.setAdapter(currencyAdapter);
spinnerCurrency.setPrompt("Список валют");
spinnerCurrency.setOnItemClickListener(onItemSelectedListener()); dbHelper
= new DBHelper(this);
false);
isEdited = getIntent().getBooleanExtra("isEdited",
if(isEdited)
{
index = getIntent().getIntExtra("id", 0);
incomeDate.setText(getIntent().getStringExtra("date"));
incomeSum.setText(String.valueOf(getIntent().getDoubleExtra("sum",
0.0)));
incomeComment.setText(getIntent().getStringExtra("comment"));
categoryText.setText(getIntent().getStringExtra("category"));
spinnerCurrency.setSelection(currencyAdapter.getPosition(getIntent().getStrin
gExtra("currency")));
}
}
```

```
AdapterView.OnItemClickListener onItemSelectedListener()
{
return new AdapterView.OnItemClickListener() { @Override
public void onItemSelected(AdapterView<?> parent,
View view, int position, long id) {
currency = currencies[position]; if(currency!= null &&
!incomeSum.getText().toString().equals("")) {
sumInUAH = currencyDict.get(currency) *
Double.parseDouble(incomeSum.getText().toString());
}
}
@Override
public void onNothingSelected(AdapterView<?> parent)
{
}
};
```

```

}
public void onClick(View view) {
    ContentValues contentValues = new ContentValues(); if(sumInUAH == 0 &&
    currency!= null && !incomeSum.getText().toString().equals("")) {
        sumInUAH = currencyDict.get(currency) *
        Double.parseDouble(incomeSum.getText().toString());
    }
    database = dbHelper.getWritableDatabase(); switch (view.getId()) {
    case R.id.date_imgBtn: {
        Intent intent_data = new Intent(IncomeActivity.this,CalendarActivity.class);
        startActivityForResult(intent_data,REQUEST_CODE_DATA);
        break;
    }
    case R.id.category_imgBtn: {
        Intent intent_category = new
        Intent(IncomeActivity.this,CategoryIncomeActivity.class);
        startActivityForResult(intent_category,REQUEST_CODE_CATEGORY);
        break;
    }
    case R.id.btn_save:
        {contentValues.put(PurseContract.IncomeEntry.INCOME_DATE,
        incomeDate.getText().toString());
        contentValues.put(PurseContract.IncomeEntry.INCOME_CATEGORY,
        categoryText.getText().toString());
        contentValues.put(PurseContract.IncomeEntry.INCOME_SUM,
        Double.parseDouble(incomeSum.getText().toString()));
        contentValues.put(PurseContract.IncomeEntry.INCOME_CURRENCY, currency);
        contentValues.put(PurseContract.IncomeEntry.INCOME_COMMENT,
        incomeComment.getText().toString());
        contentValues.put(PurseContract.IncomeEntry.INCOME_SUM_IN_UAH,
        sumInUAH);
        if(!isEdited) { database.insert(PurseContract.IncomeEntry.TABLE_INCOME, null,
        contentValues);
        }
        else {
            database.update(PurseContract.IncomeEntry.TABLE_INCOME, contentValues,
            PurseContract.IncomeEntry._ID + "= ?", new String[]{String.valueOf(index)});
        }
        sumInUAH = 0;
        Intent intent = new Intent(IncomeActivity.this, MainActivity.class);
        startActivity(intent); break;
    }
    }
    dbHelper.close();
}

```

```

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data); if(resultCode
    ==RESULT_OK) {
        switch (requestCode){
            case REQUEST_CODE_DATA:{
                String calendarDate = data.getStringExtra("calendarDate");
                incomeDate.setText(calendarDate); break;
            }
            case REQUEST_CODE_CATEGORY:{
                String categoryName = data.getStringExtra("category");
                categoryText.setText(categoryName); break;
            }
        }
    }
}

```

CategoryIncomeActivity.java

```

package com.example.habik.diplomapurse.Income; import ...
public class CategoryIncomeActivity extends AppCompatActivity implements
View.OnClickListener{
    List<Category> categoriesInc = new ArrayList<>(); ImageButton
    imgBtnAddCategory;
    EditText addCategoryET; String catName;
    String oldCatName; RecyclerView recyclerView; DataAdapterIncome adapter;
    private DBHelper dbHelper; AlertDialog.Builder builder; Context context;
    int position;
    boolean isEdited = false; SQLiteDatabase database; @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_category_income); imgBtnAddCategory =
        (ImageButton) findViewById(R.id.btn_save);
        imgBtnAddCategory.setOnClickListener(this); addCategoryET = (EditText)
        findViewById(R.id.category_et);
        recyclerView = (RecyclerView) findViewById(R.id.category_income_list);
        context = CategoryIncomeActivity.this; dbHelper = new DBHelper(this);
        database = dbHelper.getWritableDatabase();
        adapter = new DataAdapterIncome(this, categoriesInc, new
        DataAdapterIncome.OnItemClickListener() {
            @Override
            public void onItemClick(final Category item) { builder = new
            AlertDialog.Builder(context);
            builder.setTitle(item.getNameCategory().toString()).setMessage("Що Ви хочете
            зробити з даною
            категорією?").setIcon(R.drawable.category2).setCancelable(true);
            builder.setNegativeButton("Видалити", new DialogInterface.OnClickListener() {

```

```

@Override
public void onClick(DialogInterface dialog, int which) {
    if(item.getNameCategory().toString().equalsIgnoreCase(""))
    {
        return;
    }
    database.delete(PurseContract.IncomeCategoriesEntry.TABLE_INCOME_CATEGORY,
PurseContract.IncomeCategoriesEntry.INCOME_CATEGORY_NAME + "
= ?",new String[]{item.getNameCategory().toString()});
    categoriesInc.remove(item); adapter.notifyDataSetChanged();
    dbHelper.close();
}
});
builder.setPositiveButton("Змінити", new DialogInterface.OnClickListener() {

```

```

@Override
public void onClick(DialogInterface dialog,
int which) {

    dialog.cancel(); oldCatName =
    item.getNameCategory().toString();
    position = adapter.getItemCount();
    addCategoryET.setText(item.getNameCategory().toString());
    isEdited = true;
}
});
builder.setNeutralButton("Вибрати", new DialogInterface.OnClickListener() {
@Override
public void onClick(DialogInterface dialog,
int which) {

```

```

Intent intent_category = new Intent(); intent_category.putExtra("category",
item.getNameCategory().toString());
    setResult(RESULT_OK, intent_category); finish();
}
});
AlertDialog alertDialog = builder.create(); alertDialog.show();
}
});
recyclerView.setAdapter(adapter); Cursor cursor =
    database.query(PurseContract.IncomeCategoriesEntry.TABLE_INCOME_CATEGORY, null, null, null, null, null, null);
    if (cursor.moveToFirst()) { int nameIndex =

```

```

cursor.getColumnIndex(PurseContract.IncomeCategoriesEntry.INCOME_CATEGORY_NAME);
Category category = new Category(cursor.getString(nameIndex));
categoriesInc.add(category); while (cursor.moveToNext()) {
nameIndex =
cursor.getColumnIndex(PurseContract.IncomeCategoriesEntry.INCOME_CATEGORY_NAME);
category = new Category(cursor.getString(nameIndex));
categoriesInc.add(category);
}
adapter.notifyDataSetChanged();
}
cursor.close();
}

```

```

@Override
public void onClick(View v) {
catName = addCategoryET.getText().toString(); ContentValues contentValues =
new ContentValues(); database = dbHelper.getWritableDatabase();
switch (v.getId()) {
case R.id.btn_save: {
if (isEdited == true) {
if (catName.equalsIgnoreCase("")) { break;
}
contentValues.put(PurseContract.IncomeCategoriesEntry.INCOME_CATEGORY_NAME, catName);
database.update(PurseContract.IncomeCategoriesEntry.TABLE_INCOME_CATEGORY, contentValues,
PurseContract.IncomeCategoriesEntry.INCOME_CATEGORY_NAME + " = ?",
new String[]{oldCatName});
addCategoryET.setText(""); categoriesInc.set(position - 1, new
Category(catName)); adapter.notifyDataSetChanged(); isEdited = false;
}
else{
contentValues.put(PurseContract.IncomeCategoriesEntry.INCOME_CATEGORY_NAME, catName);
database.insert(PurseContract.IncomeCategoriesEntry.TABLE_INCOME_CATEGORY, null, contentValues);
categoriesInc.add(new Category(catName)); addCategoryET.setText("");
adapter.notifyDataSetChanged();
}
break;
}
}
dbHelper.close();
}

```

```
}
}
```

AllOutcomeInCategoryActivity.java

```
package com.example.habik.diplomapurse.AllOutcome; import ...
public class AllOutcomeInCategoryActivity extends AppCompatActivity {
    SQLiteDatabase database; DBHelper dbHelper; AlertDialog.Builder builder;
    List<AllOutcomeInCategory> allOutcomeInCategoryList = new ArrayList<>();
    Context context; RecyclerView recyclerView;
    DataAdapterOutcomeItem adapter; String category;
    String currentDate; DateTypeEnum dateType; @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_all_outcome_in_category);
        category = getIntent().getStringExtra("category_outcome");
        currentDate = getIntent().getStringExtra("date_outcome");
        dateType = (DateTypeEnum)
            getIntent().getSerializableExtra("dateType_outcome");
        setTitle(category);
        recyclerView = findViewById(R.id.outcome_recycler); context =
            AllOutcomeInCategoryActivity.this; dbHelper = new DBHelper(this);
        adapter = new DataAdapterOutcomeItem(this, allOutcomeInCategoryList, new
            DataAdapterOutcomeItem.OnItemClickListener() {
            @Override
            public void onItemClick(final AllOutcomeInCategory
            item)
            {
                builder = new AlertDialog.Builder(context);
                builder.setTitle(item.getComment().toString()).setMessage("Що ви хочете
                зробити з даною
                витратою?").setIcon(R.drawable.category2).setCancelable(true);
                builder.setNegativeButton("Видалити", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog,
                int which) {
                    if(PurseContract.OutcomeEntry._ID.equalsIgnoreCase(""))
                    {
                        return;
                    }
                    database = dbHelper.getWritableDatabase();
                    database.delete(PurseContract.OutcomeEntry.TABLE_OUTCOME,PurseContract.OutcomeEntry._ID + " = ?", new String[]{String.valueOf(item.getId())});
                    allOutcomeInCategoryList.remove(item); database.close();
                    adapter.notifyDataSetChanged();
                }
            }
        });
    }
}
```

```

    }
    });
    builder.setPositiveButton("Змінити", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog,
            int which) {
            Intent intent = new
            Intent(AllOutcomeInCategoryActivity.this, OutcomeActivity.class);
            intent.putExtra("id", item.getId()); intent.putExtra("date", item.getDate());
            intent.putExtra("currency", item.getCurrency());
            intent.putExtra("sum", item.getSum());
            intent.putExtra("comment", item.getComment());
            intent.putExtra("category", category); intent.putExtra("isEdited", true);
            startActivityForResult(intent, 205);
        }
    });
}
}); AlertDialog alertDialog = builder.create(); alertDialog.show();
getAllOutcomesByCategory(); recyclerView.setAdapter(adapter);    }
private void getAllOutcomesByCategory()
{
    database = dbHelper.getReadableDatabase(); Cursor cursor;
    String[] dates = getConvertedDate(currentDate, dateType).split("-");
    for (String date : dates) { cursor =
    database.query(PurseContract.OutcomeEntry.TABLE_OUTCOME,
    new String[]{PurseContract.OutcomeEntry._ID,
    PurseContract.OutcomeEntry.OUTCOME_DATE,
    PurseContract.OutcomeEntry.OUTCOME_CURRENCY,
    PurseContract.OutcomeEntry.OUTCOME_COMMENT,
    PurseContract.OutcomeEntry.OUTCOME_SUM},
    "category=? AND date=?", new String[]{category, date}, null, null, null
    );
    if (cursor.moveToFirst()) {
        AllOutcomeInCategory allOutcomeInCategory = new
        AllOutcomeInCategory(cursor.getInt(0), cursor.getString(1), cursor.getString(2),
        cursor.getString(3), cursor.getDouble(4));
        allOutcomeInCategoryList.add(allOutcomeInCategory);
        while (cursor.moveToNext()) { allOutcomeInCategory = new
        AllOutcomeInCategory(cursor.getInt(0), cursor.getString(1), cursor.getString(2),
        cursor.getString(3), cursor.getDouble(4));
        allOutcomeInCategoryList.add(allOutcomeInCategory);
        }
    }
    cursor.close();
}
}

```

```

database.close();
}
private String getConvertedDate(String date, DateTypeEnum dateType) {
if(date.equals("")) return "";
String convertedDate = ""; switch (dateType) {
case DAY: {
DateFormat df = new SimpleDateFormat("d.MM.yyyy");
Calendar c = Calendar.getInstance(); try {
c.setTime(df.parse(date));
} catch (ParseException e) { return "";
}
convertedDate = df.format(c.getTime()); return convertedDate;
}
case WEEK: {
DateFormat df = new SimpleDateFormat("d.MM.yyyy");
Calendar c = Calendar.getInstance(); try {
c.setTime(df.parse(date));
} catch (ParseException e) { return "";1; i--}
{"-"};
}
int dayOfWeek = c.get(Calendar.DAY_OF_WEEK); for(int i = dayOfWeek == 1 ?
7 : dayOfWeek; i >
convertedDate += (df.format(c.getTime()) + c.add(Calendar.DAY_OF_MONTH, -
1);
}
if(dayOfWeek == 1)
{
convertedDate += (df.format(c.getTime()) +
}
return convertedDate.replaceAll("[-]+$", "");
}
case MONTH: {
DateFormat df = new SimpleDateFormat("MM.yyyy"); Calendar c =
Calendar.getInstance();
try {
c.setTime(new SimpleDateFormat("dd.MM.yyyy").parse(date));
} catch (ParseException e) { return "";
}
convertedDate = "%" + df.format(c.getTime()); return convertedDate;
}
}
return convertedDate;
}
}

```

ДОДАТОК В
Слайди презентації

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Дипломна робота на тему:

**Дослідження та програмна реалізація
мобільного застосунку для управління
особистими доходами та витратами**

Виконав: ст. гр. КНТ-219м

А.М.Кодочигов

Керівник: к.т.н., доц.

С.К.Корнієнко

2020

Активация Windows
Чтобы активировать Windows, перейдите в раздел
"Параметры".

1

Рисунок В.1 – Слайд №1

- Мета: створення мобільного застосунку для управління особистими доходами та витратами.
- Об'єкт дослідження – операційні системи та інструменти програмування для створення застосунку для управління доходами та витратами.

Активация Windows
Чтобы активировать Windows, перейдите в раздел
"Параметры".

2

Рисунок В.2 – Слайд №2

Вимоги до функціоналу застосунку

- інтуїтивно зрозумілий і зручний інтерфейс;
- створення записів витрат і доходів з можливістю застосування різних атрибутів: дата, категорія, сума, валюта, коментар і ін. ;
- редагування / видалення раніше доданих фінансових операцій;
- додавання / видалення / редагування категорій витрат і доходів;
- перегляд доходів і витрат з можливістю вибору часового проміжку часу;
- створення можливості вибору валюти для фінансових операцій;
- перегляд статистики витрат і доходів за допомогою кругових діаграм.

Активация Windows
Чтобы активировать Windows, перейдите в раздел
"Параметры".

3

Рисунок В.3 – Слайд №3

Вибір операційної системи

Android - це операційна система, спроектована для роботи на смартфонах, планшетах та інших пристроях. Вона заснована на ядрі Linux, введено всього кілька оновлених понять і використовується більшість засобів Linux.

- У понад 70% смартфонах, в 2020 році, встановлена операційна система Android .
- Android - це відкрита платформа. Вона доступна всім, і будь-який виробник мобільних пристроїв або розробник може використовувати Android в своїх цілях: для створення застосунків, пристроїв і навіть власної операційної системи.
- ОС Android сьогодні можна вибрати телефон з гігантського списку пропозицій. Дана ОС є відкритою і безкоштовною, тому конкуренція на ринку виробників мобільних пристроїв йде щільно.

Активация Windows
Чтобы активировать Windows, перейдите в раздел
"Параметры".

4

Рисунок В.4 – Слайд №4

Середовище розробки

Android Studio - це інтегроване середовище розробки для створення мобільних застосунків, створена компанією Google, є офіційним засобом розробки Android застосунків.

- Офіційні мови програмування для платформи Android : [Java](#) , [C ++](#) і Kotlin .
- Головною перевагою суспільством Android Studio вважається вбудований набір засобів розробки, котрий дозволяє створювати додатки для певного пакету програм, використовуючи компоненти, яких немає в інших версіях застосунка. Також хорошим плюсом є зручний конструктор інтерфейсу.
- Android Studio включає в себе велику кількість емуляторів різних видів пристроїв. Це дозволяє розробляти інтерфейс програм, розміщувати компоненти на екрані і переглядати результат на самих різних пристроях. Крім цього в Android Studio зручно і зрозуміло реалізований інтерфейс для роботи, меню структури проекту та пошук помилок. Абсолютно весь інтерфейс легко можна налаштувати, непотрібні вікна і панелі прибрати чи згорнути. Це дуже корисно при створенні об'ємних і високонавантажених застосунків і проектів.

Активация Windows
Чтобы активировать Windows, перейдите в раздел
"Параметры".

5

Рисунок В.5 – Слайд №5

Вибір мови програмування

Java - є найбільш поширеною мовою програмування для розробки застосунків під операційною системою Android. Java строго типізована об'єктно-орієнтована мова програмування. Це одна з найпопулярніших мов програмування, яким користується приблизно 9 мільйонів розробників. ОС Android працює на 60% смартфонів у світі, це означає, що додаток на Java має найбільший потенціал, в плані охоплення користувачів.

Java
Працює на всіх платформах, універсальність.
Вихідний код бібліотек відкритий.
Потужна підтримка спільноти для Java і Android.
Легкість у вивченні.
Гнучка система безпеки.

Активация Windows
Чтобы активировать Windows, перейдите в раздел
"Параметры".

6

Рисунок В.6 – Слайд №6

Material Design

Material Design - є комплексною, кроссплатформенною концепцією, розробленою Google для створення застосунків з візуалізацією і рухомими елементами.

- простота сприйняття ;
- уніфікований інтерфейс на всіх платформах;
- логічна поведінка об'єктів;
- адаптивність.

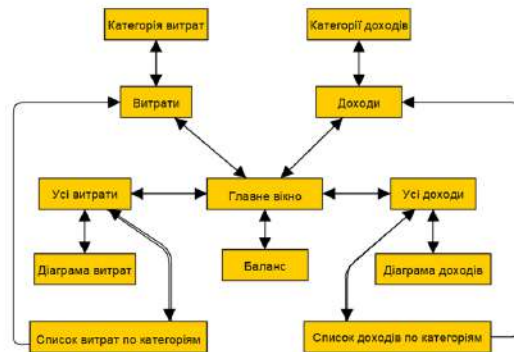
Активация Windows
Чтобы активировать Windows, перейдите в раздел "Параметры".

7

Рисунок В.7 – Слайд №7

Структура проекту

Для створення застосунка в першу чергу була спроектована діаграма класів з метою опису структури проекту, де класи по своїй суті є вікнами застосунку, в яких будуть реалізовуватися описані функції, а виходять з них стрілки - залежності між ними. Кожна стрілка позначає можливість переходу з одного вікна в інше, якщо стрілка двостороння, то в такому випадку перехід здійснюється в обидва боки. Це посприяло тому, що в подальшому, було сформульовано точне уявлення того, що потрібно робити, як це реалізувати і які труднощі можуть з'явитися в подальшому (рис 3.1).



Активация Windows
Чтобы активировать Windows, перейдите в раздел "Параметры".

8

Рисунок В.8 – Слайд №8

Макет активностей додатку

Головне вікно. Доходи. Категорії доходів.

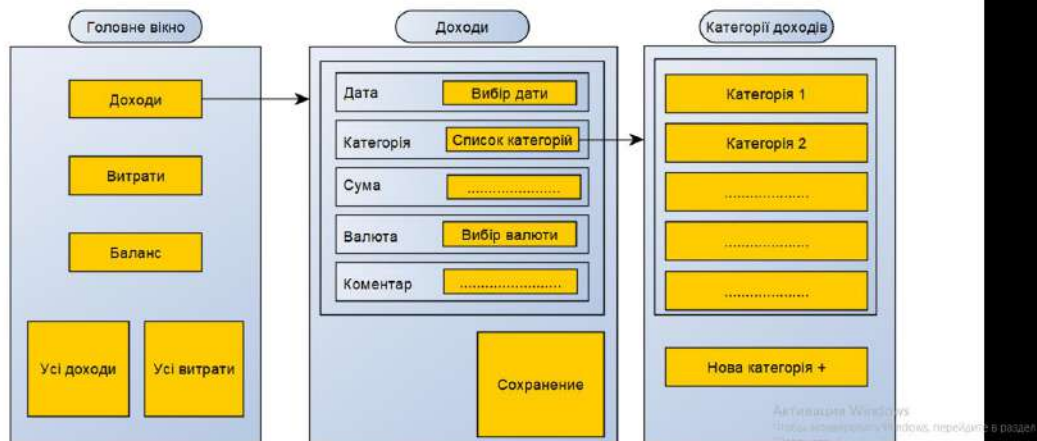


Рисунок В.9 – Слайд №9

Всі доходи. Діаграма доходів.
Список доходів по категоріям.

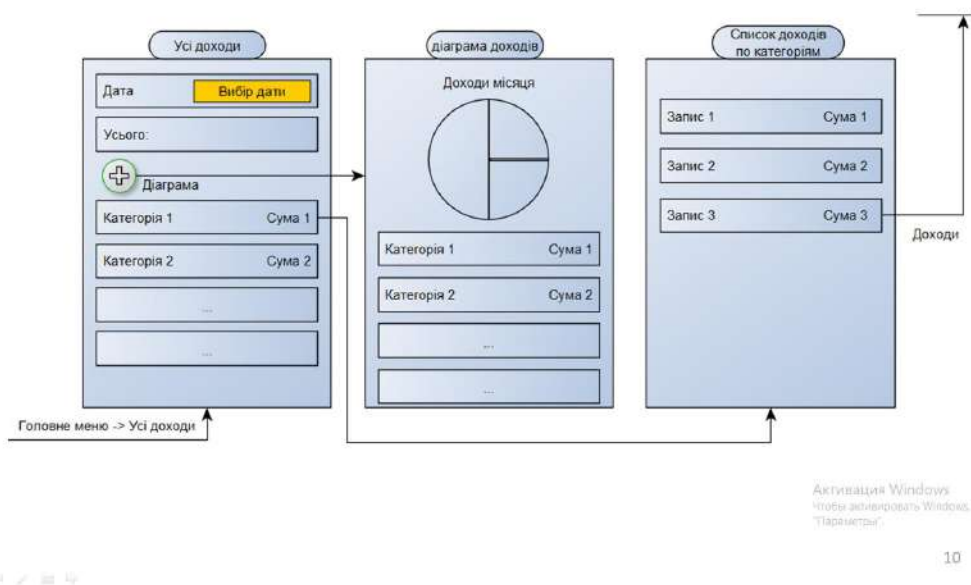
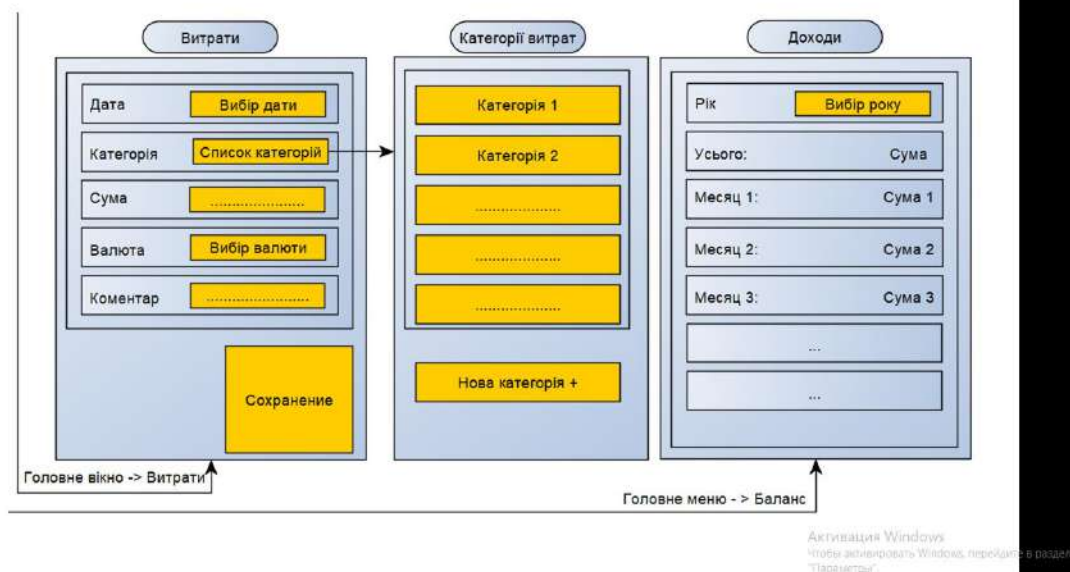


Рисунок В.10 – Слайд №10

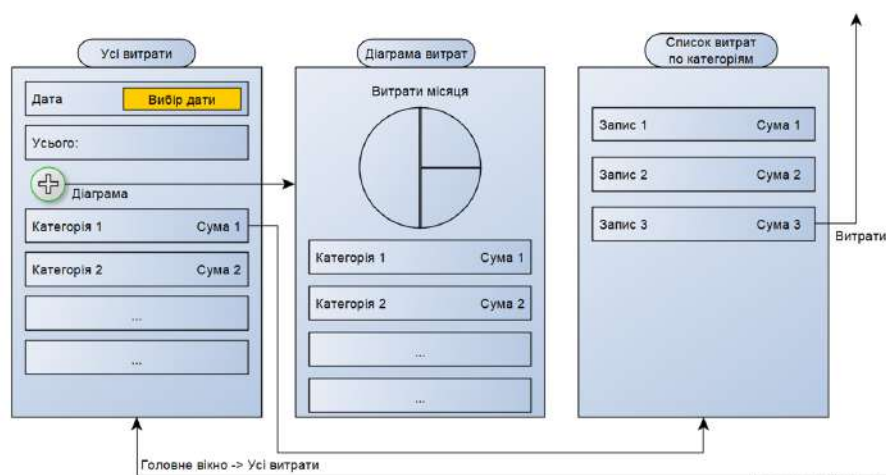
Витрати. Категорії витрат. Баланс.



11

Рисунок В.11 – Слайд №11

Всі витрати. Графік витрат. Перелік витрат за категоріями



12

Рисунок В.12 – Слайд №12

Програмна реалізація застосунка

- **SQLite** — полегшена реляційна система керування базами даних. Втілена у вигляді бібліотеки.

Income	Income_category	Outcome	Outcome_category
id	id	id	id
date (дата)	name (назва категорії)	date (дата)	name (найменування категорії витрат)
category (категорія)		category (категорія)	
sum (сума доходу)		sum (сума доходу)	
currency (валюта, введеного доходу)		currency (валюта, введеного доходу)	
comment (коментар до операції)		comment (коментар до операції)	
sum (сума в гривнях)		sum (сума в гривнях)	

Рисунок В.13 – Слайд №13

Результати роботи

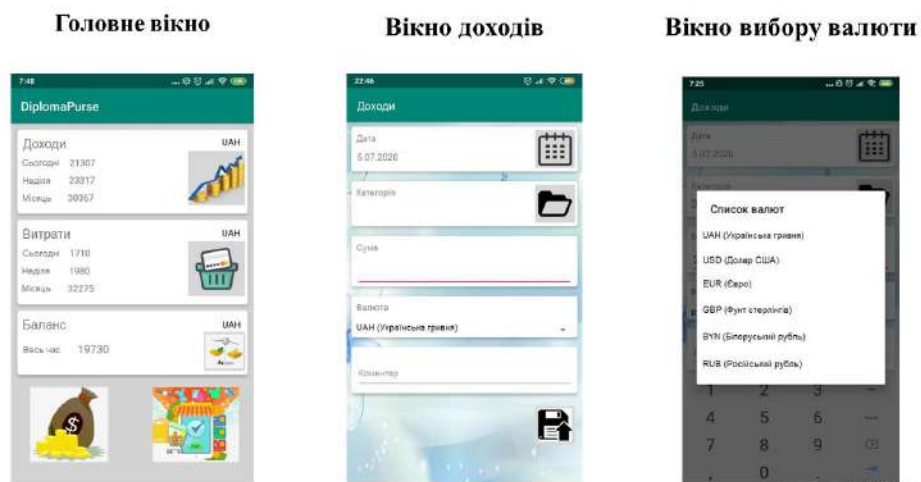
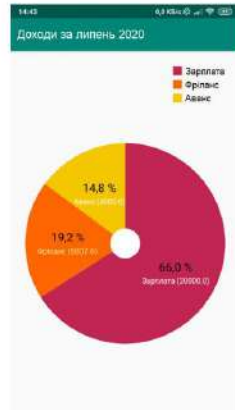


Рисунок В.14 – Слайд №14

Діаграма



Всього доходів за неділю

Всі доходи	
неділя	05.07.2020
Всього	30307.0
Діаграма	
Витрати за поточний період	
Зарплата	20000.0
Фріланс	5807.0
Аванс	4500.0
Стипендія	0.0

Всього доходів по місяцям

Баланс	
← 2020 →	
Всього	46304.0
січень	-11000.0
лютий	30000.0
березень	4212.0
квітень	25000.0
травень	-1908.0
червень	0.0
липень	0.0
серпень	0.0
вересень	0.0
жовтень	0.0
листопад	0.0
грудень	0.0

Активация Windows
Чтобы активировать Windows, перейдите в раздел
"Параметры".

15

Рисунок В.15 – Слайд №15

ВИСНОВКИ

- У моїй роботі були розглянуті та обрані операційна система та інструменти програмування для створення застосунку для управління доходами та витратами.
- Було обрано Android, як єдину, та головну операційну систему для реалізації застосунку. Так як вона є відкритою для розробників а також найпопулярнішою операційною системою на смартфонах.
- У даній роботі було обрано середовище розробки, а саме Android Studio, та мову програмування Java, так як вона є універсальною, гнучкою а також її вихідний код бібліотек є відкритий.
- Були створені макет активностей додатку, база даних, та після чого був написан текст програми.
- Зроблені організаційно-економічні розрахунки та запезпечені всі заходи с охорони праці.
- Технічною новизною моєї роботи є те, що я використовував лише безкоштовні інструменти та бібліотеки програмування, які знаходяться у вільному доступі, та це ніяк негативно не вплинуло на результат моєї роботи. Чому я забов'язан особливостям обраної мною операційно системи. Мій застосунок залишається зручним, швидким у використанні та інтуїтивно зрозумілим, завдяки чому він може бути конкурентно спроможним на ринку аналогічних застосунків.

16

Рисунок В.16 – Слайд №16