МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД «УНІВЕРСИТЕТ «КРОК»

Фаховий коледж Університету «КРОК»

ДИПЛОМНА РОБОТА

за темою

«Розробка системи інтернет продажів, що працює на мобільній операційній системі»

Студент 4 курсу групи КН-20к

Сирота Петро Сергійович
(прізвище, ім`я та по-батькові студента)

Керівник дипломної роботи

Кандидат фізико-математичних наук
(посада керівника)

Кириченко Віктор Вікторович
(прізвище, ім'я та по-батькові керівника)

До захисту
(резолюція «До захисту»)

(підпис студента)

10.06.24
(дата)

(підпис викладача)

2024 рік

СКОРОЧЕННЯ

IOS(I Operating System) - Відсилає до найменувань фірмових мобільних пристроїв Apple (iPhone, iPod та iPad) плюс скорочення OS (Operating System, операційна система).

FP(Функціональне програмування) - парадигма програмування, в якій процес обчислення трактується як обчислення значень функцій у математичному розумінні останніх.

API(application programming interface) - Програмний інтерфейс, тобто опис способів взаємодії однієї комп'ютерної програми коїться з іншими. Зазвичай входить до опису будь-якого інтернет-протоколу, програмного каркасу або стандарту викликів функцій операційної системи. Часто реалізується окремою програмною бібліотекою чи сервісом операційної системи. Використовується програмістами під час написання різноманітних додатків.

Wi-fi(Wireless Fidelity) - Торгова марка Wi-Fi Alliance та загальновживана назва для стандарту IEEE 802.11 передавання цифрових потоків даних по радіоканалах.

GPS(Global Positioning System) - Сукупність радіоелектронних засобів, що дозволяє визначати положення та швидкість руху об'єкта на поверхні Землі або в атмосфері.

КПК(кишеньковий персональний комп'ютер) - Кишеньковий комп'ютер, кишеньковий персональний комп'ютер, (КПК), (англ. personal digital assistant), а також: ручний комп'ютер, надолонний комп'ютер, надолонник, палмтоп) — збірна назва класу портативних електронних обчислювальних пристроїв, спочатку запропонованих до використання як електронні органайзери.

NFC(Near-Field Communication) - технологія бездротового зв'язку малого радіуса дії «за один дотик». Забезпечує обмін даними між пристроями, насамперед

смартфонами та безконтактними платіжними терміналами на відстані близько 10 см.

OS(operating system) - Операційна система, скорочено ОС (англ. operating system, OS) - це базовий комплекс програм, що виконує керування апаратною складовою комп'ютера або віртуальної машини; забезпечує керування обчислювальним процесом і організовує взаємодію з користувачем.

SDK(software Development Kit) - набір із засобів розробки, утиліт і документації, який дає програмістам змогу створювати прикладні програми за визначеною технологією або для певної платформи.

ПК(Персональний комп'ютер) - багатофункціональний комп'ютер, з розмірами, можливостями і ціною, що уможливлює його індивідуальне використання.

FM-радіо(FM-частотна модуляція) - технологія радіотрансляції, винайдена Едвіном Говардом Армстронгом, який використав частотну модуляцію (FM), щоб забезпечити високу точність відтворення звуку по радіо. Термін FM є абревіатурою терміну «частотна модуляція».

PDF (Portable Document Format) - формат файлу, створений і підтримуваний компанією Adobe Systems, для представлення двовимірних документів у незалежному від пристрою виведення та роздільної здатності вигляді.

ASCII (Американський стандартний код для інформаційного обміну) - в обчислювальній техніці — система кодів, у якій числа від 0 до 127 включно поставлені у відповідність літерам, цифрам і символам пунктуації.

ZIP (Lempel Ziv Welch) - формат стиснення та архівації даних. Файл цього формату зазвичай має розширення .zip і зберігає у стиснутому або не стиснутому вигляді один або декілька файлів. Використовує LZW-стиснення, яке не вносить спотворень і втрат.

AES (Advanced Encryption Standard) - також відомий під назвою Rijndael, симетричний алгоритм блочного шифрування (розмір блока 128 біт, ключ 128/192/256 біт). Фіналіст конкурсу AES, прийнятий як американський стандарт шифрування урядом США.

Web(Всесвітнє павутиння) - найбільше всесвітнє багатомовне сховище інформації в електронному вигляді: десятки мільйонів пов'язаних між собою документів, що розташовані на комп'ютерах, розміщених на всій земній кулі.

ГБ(Гігабайт) - кратна одиниця вимірювання кількості інформації, що дорівнює 1 073 741 824 стандартним (8-бітним) байтам або 1024 мегабайтам.

HTML5(HyperText Markup Language 5) - наступна версія мови HTML. До складу робочої групи з HTML5 увійшли AOL, Apple, Google, IBM, Microsoft, Mozilla, Nokia, Opera та кілька сотень інших виробників.

MWC (Mobile World Congress) - зібрання світової мобільної індустрії, до складу якого входять міжнародна виставка мобільних технологій та конференція видатних CEO, які представляють операторів мобільного зв'язку, виробників обладнання, постачальників технологій, власників та постачальників контенту зі всього світу.

MVVM(Model-View-ViewModel) - шаблон проєктування, що застосовується під час проєктування архітектури застосунків (додатків).

ADT Control - забезпечує плавний перехід від домашнього або робочого, до мобільного розумного захисту та автоматизації.

XML(Розширювана мова розмітки) - запропонований консорціумом World Wide Web Consortium стандарт побудови мов розмітки ієрархічно структурованих даних для обміну між різними застосунками, зокрема, через Інтернет.

IntelliJ IDEA — комерційне інтегроване середовище розробки для різних мов програмування (Java, Python, Scala, PHP та ін.) від компанії JetBrains.

ЗМІСТ

ВСТУП

Кожен раз коли ви йдете в магазин, чи впевнені ви що запам'ятали все що треба купити? Інколи стаються ситуації в яких ви набираєте більше непотрібних речей, продуктів, коли вийшли за молоком, а набрали ще йогуртів, шоколаду та усього іншого.

Тому список покупок необхідний як для людей у яких це стається не так рідко, так і людям у яких це не разова ситуація, бо ніхто не застрахований від помилок тому просте рішення таке як список зможе полегшити купівлю продуктів в магазинах онлайн та офлайн.

Розуміння того що ти хочеш придбати допоможе організувати своє життя хоча б у походах до магазину що позитивно вплине на вашому гаманці та часу буде витрачено менше який можна витратити на хороші книги чи ігри, також коли це увійде у звичку ви здивуєтесь як багато грошей можна заощаджувати таким простим способом.

Список покупок можна скласти навіть на паперовому листку, але зараз майже всі люди використовують смартфони в повсякденному житті мало хто хоча б кудись ходить без телефону, та понад 70% людей користуються смартфонами й ця цифра все росте і навряд буде скидати оберти, тому я зосередився на розробці саме додатку оскільки це зручніше у повсякденному використанні.

РОЗДІЛ 1 ОГЛЯДОВИЙ АНАЛІЗ ДОДАТКІВ «СПИСОК ПОКУПОК» НА ANDROID

1.1 Наведення прикладів додатків

1)Listonic

Одним з популярних додатків для складання списку є Listonic, Крім списку, в нього також можна додавати ціни та кількість продуктів, що дозволяє сформувати приблизну загальну вартість. Після реєстрації в системі з'явиться функція синхронізації, завдяки якій отримати доступ до списків можна з інших підключених пристроїв, і на веб-сайті також. Є голосові функції можливість додавання позицій голосовими командами, а під час введення в інтерфейсі відображаються найбільш популярні варіанти, що економить час при складанні списку. (рис.1.1)

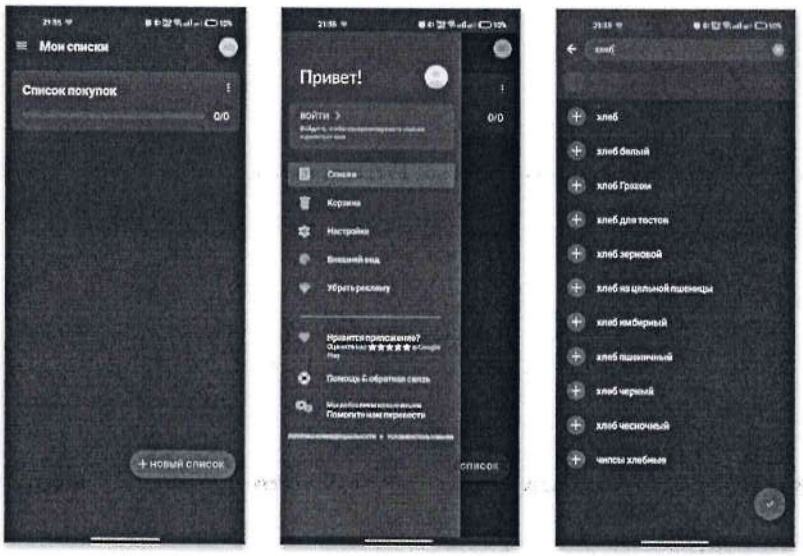

Рис 1.1 Інтерфейс Listonic

За допомогою Listonic можна організувати сімейні покупки, оскільки передбачається функція обміну даними. Також є функція сортування продуктів, що спрощує пошук товарів під час походу в магазині. Для зручності можна створювати безліч списків: наприклад, запланувати покупки на вихідні або на

свята та інші дати. Інтерфейс нажаль російською мовою, а його оформлення можливо налаштувати: вибрати тему, розмір тексту і стиль шрифт.

2)SoftList

Простий та зручний список покупок SoftList окрім звичайного функціонала має в своєму розпорядженні додаткові функції такі як: підтримка декількох списків, можливість ділитися створеними списками, зберігання історії покупок і багато іншого. У додатку також реалізована функція розпізнавання товарів за допомогою сканування штрих кодів, а зображення позицій можна самостійно налаштувати. Функція формування звітів, дозволяє відстежувати ціни що може допомогти в деяких ситуаціях.(рис.1.2)

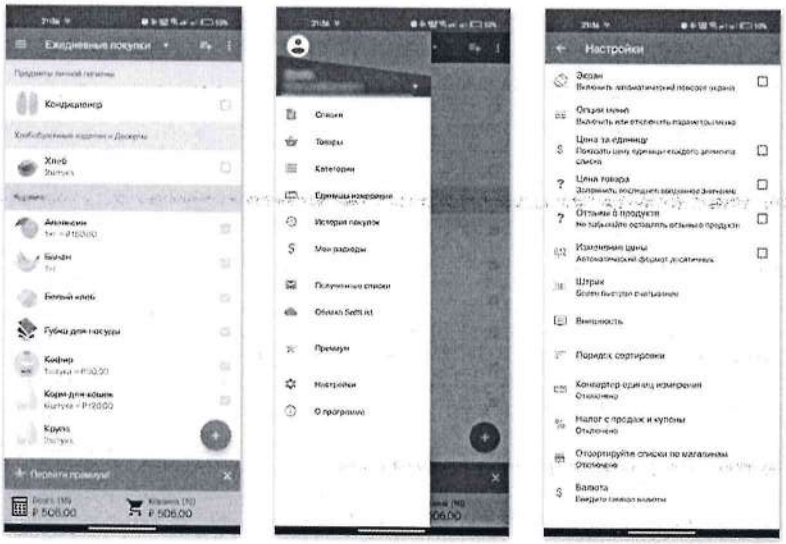

Рис 1.2 інтерфейс SoftList

Інтерфейс SoftList нажаль також російською мовою, У ньому всі позиції автоматично розподіляються за категоріями. Кожну з них можна редагувати або видалити, також є можливість створювати нові категорії і шаблони товарів. Сформованими даними є можливість поділитися, і відразу відправити на друк. У розділі з настройками включаються додаткові можливості: конвертер одиниць вимірювання, податок з продажів і купони.

3)Bring

Ще один додаток для складання списків покупок-Це Bring! Однією з особливостей програми є вбудована колекція рецептів, яка регулярно поповнюється. Кожен з рецептів можна зберігати собі і вибрати зі сторінки опису відсутні інгредієнти. Імпортувати рецепти можна з інтернету, використовуючи функцію **«поділитися»**. У додатку також є формування відразу декількох списків, припустимо продуктів або ліків. Намальованих значків продуктів у інтерфейсі налічується понад 350.(рис.1.3)

Рис 1.3 інтерфейс Bring

Додаток нагадує про те, що треба сходити в магазин, також в ньому реалізована система профілів і синхронізація з хмарним сховищем. У Bring! є безліч налаштувань від персоналізації, включаючи темний і світлий режим, до настроювання зовнішнього вигляду самого списку. Крім того, при плануванні покупки того чи іншого товару додаток порекомендує відповідний продукт: наприклад, якщо потрібно купити картоплі, додаток запропонує придбати і солі або спецій.

4)Купи батон!

Додаток Купи батон! відрізняється простим і приємним інтерфейсом в мінімалістичному виконанні що не відволікає. При введенні назви продукту відобразиться перелік варіантів, позначених різними кольорами в залежності від типу. Користувач може або вибрати шаблон, або створити нову позицію, визначивши для нього категорію та колір. Доступно створення відразу декількох списків. Після створення облікового запису доступ буде наданий і на інші пристрої.(рис.1.4)

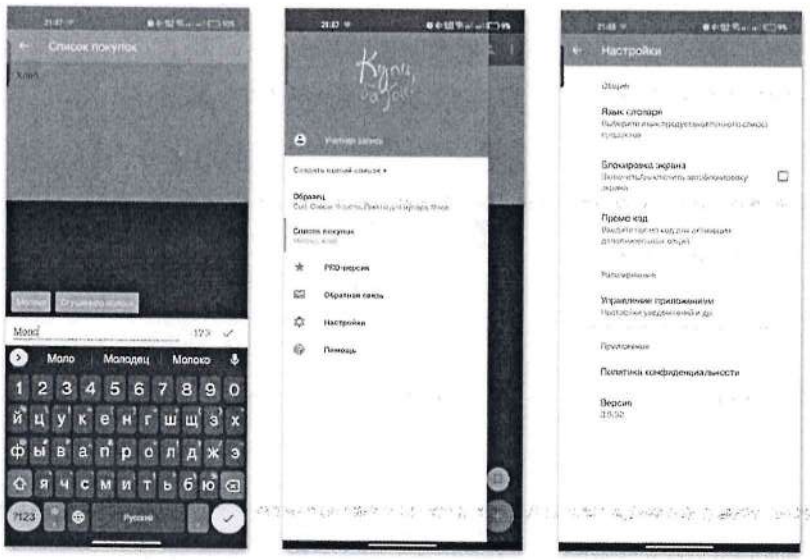

Рис 1.4 інтерфейс Купи батон!

Натискаючи на три горизонтальні смужки у верхньому кутку зліва, можна відкрити головне меню, яке відображає списки покупок із кількома варіантами, які можна знайти в ньому. Крім того, це дозволяє перейти в розділ налаштувань і показати інструкції по використанню. Хоча основні функції програми закінчуються на цьому рівні, його головна перевага полягає в зручному інтерфейсі та простому управлінні.

5)Ваш список покупок

Програма Твій список покупок дозволяє швидко створювати списки, змінювати їх, ділитися, клонувати та виконувати інші функції. Воно постійно працює офлайн і має вбудовану велику базу товарів, яка включає широкий асортимент продуктів. Усі категорії представлені за допомогою кольорів і зображень, що полегшує швидке орієнтування в списку покупок під час відвідування магазину. Введіть назву продукту, і нижче будуть показані подібні варіанти. Просто напишіть цифру після пробілу, щоб вказати кількість товару. Голосове введення є можливим.(рис.1.5)

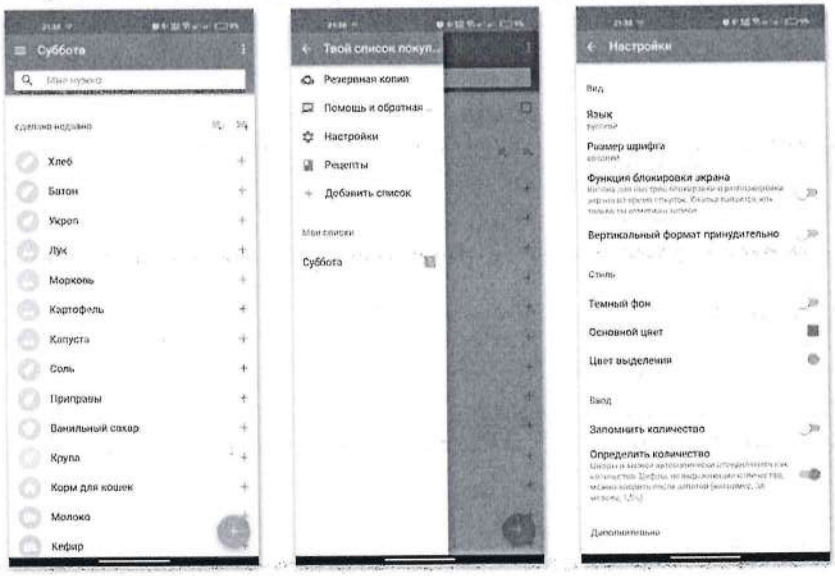

Рис 1.5 інтерфейс Ваш список покупок

Для швидкого створення списків можна використовувати історію, оскільки товари, які найчастіше купуються, зберігаються в окремому розділі. Крім того, є можливість створити книгу рецептів, вводячи рецепти вручну за допомогою вбудованого редактора або використовуючи посилання з Інтернету. Користувач може змінити категорії продуктів, які вже є, і додати свої. Видалення, копіювання та очищення списків можна зробити одним натисканням. Стиль і колір елементів інтерфейсу, світлий і темний фон, розмір шрифту — це деякі з налаштувань, доступних для налаштування. У додатку всі можливості доступні безкоштовно, і в ньому немає реклами.

1.2 Плюси використання додатків «список покупок» у повсякденності

Щоб уникнути перевантаження мозку. Організм має обмежені ресурси. У кожного з нас є розумові та фізичні сили. Ці сили витрачаються під час ухвалення рішень. Увечері складніше приймати рішення, ніж протягом дня. Зі збільшенням навантаження наш мозок гріється, а процесор смартфона розряджає акумулятор.

Емоційні, а не розумні рішення приймаються, коли ресурси мозку вичерпуються. На нього починають діяти всі види реклами, що призводить до того, що ми купуємо не те, що нам потрібно, а те, що маркетологи повинні продати нам.

Вирішити, що купити, важче після важкого робочого дня: звичайний зелений горошок або новий у красивій банці. Це є наслідком відсутності розумової енергії.

Щоб збільшити уважність. Крім того, наш час обмежений. Ми можемо зосередитися на обмеженій кількості об'єктів. Вчені стверджують, що від 3 до 7; однак багато залежить від людини та об'єктів. Простіше тримати у фокусі п'ять двоцифрових чисел, ніж п'ять чотирицифрових.

Якщо ви заздалегідь не склали список побажань на папері, ваша увага буде зосереджена на списку, який у вас в голові. Щоб не забути нічого, його потрібно тримати в центрі уваги. Ви можете включити десять пунктів у список. Події, які постійно з'являються протягом робочого дня, наприклад, начальник накричав, колеги пліткують, співробітники підводять, займають значну частину уваги. Вечеря, яку потрібно приготувати, також потребує уваги. Крім того, що наближається день народження сина: що подарувати, де провести? додатково до нещодавньої розмови зі старим знайомим. Звичайно, магазин займає більшу частину уваги через акції, оголошення гучного зв'язку, працівників, які переміщують товари з полиці на полицю та тих, хто не помічає нічого.

Згадайте, скільки разів ви зустрічалися з іншими клієнтами в магазині. Це результат неправильної уваги.

Порахуйте, що у фокусі більше п'ятнадцяти різних розмірів об'єктів. Не дивно, що деякі з найменших можуть залишитися поза увагою.

Приберіть з вашої уваги покупки та інші деталі.

Щоб зменшити витрати. Економія - це результат вільної уваги та невитрачених знань.

Ви помічаєте вигідніші пропозиції, якщо вам не потрібно стежити за списком покупок.

Ви можете витратити трохи часу, щоб дізнатися про те, що не було в списку, якщо ви не змогли прийняти десяток рішень щодо того, що купити.

У магазині може бути акція на шампунь, яким ви користуєтеся. Ви можете не помітити її, якщо немає списку покупок. Крім того, якщо ви це помітите, ви довго прийматимете рішення щодо того, чи вигідно це купувати, і скільки штук купувати.

Коли та як починати складати?

Вперед! У нашому холодильнику висить магнітний блокнот, у який ми записуємо все, що нам потрібно купити під час наступної поїздки в магазин. Як тільки ми відкриваємо останню пачку олії та записуємо її в список цукру, ми також заносимо останню пачку олії з холодильника до блокноту.

1.3 Технології для розробки додатків на андроїд

На сьогодні одна з найбільш поширених і актуальних проблем в IT-галузі — розробка мобільних додатків. Згідно з різними джерелами, користувачі мобільних додатків використовують від п'ятдесяти до сімдесяти відсотків свого цифрового часу.

На сьогодні Android (близько 70%) і iOS (близько 28%) є найбільш поширеними мобільними операційними системами.

Кожна операційна система використовує своє середовище програмування та мову програмування для розробки додатків. Kotlin або Java є мовами розробки програм для операційної системи Android.

Розробка на місцевому рівні має низку переваг. Серед них є можливість створювати програми з будь-якими необмеженими функціями (окрім обмежень самої платформи, під яку ведеться розробка), а також максимальна швидкість роботи користувача.

Однак такий метод має деякі недоліки. Основний — необхідність розробки та підтримки як мінімум двох додатків, щоб задовольнити потреби більшості користувачів. Другий недолік, який випливає з першого, полягає в тому, що замовники повинні залучати кілька команд або розробників, щоб створити мобільні програми для різних операційних систем.

Для вирішення цих проблем інді-розробники з різних компаній створили інструменти, які дозволяють розробляти програми на Android і iOS без необхідності використання обох платформ. Так, Google представив Flutter у 2014 році, а Facebook випустила React Native у 2015 році. Це кросплатформові фреймворки, які дозволяють розробникам використовувати одну кодову базу для різних операційних систем.

Flutter використовує мову програмування Dart для створення настільних, мобільних і веб-додатків, які працюють під Windows, macOS і Linux. Він є комплексом засобів розробки та фреймворку з відкритим вихідним кодом.

React Native — це кросплатформовий фреймворк з відкритим вихідним кодом, який дозволяє створювати програми для мобільних і настільних пристроїв, використовуючи JavaScript і TypeScript.

Це очевидна перевага розробки кросплатформових додатків: розробка та підтримка програмного забезпечення стає простішою завдяки використанню єдиної кодової бази. Однак є значні недоліки. З них найважливішим є те, що тепер обмеження на розробку накладаються як на платформу, так і на фреймворк.

Іншим недоліком є те, що кросплатформові програми часто працюють повільніше, ніж оригінальні програми. Це особливо стосується фреймворку React Native.

Загалом, кожен розробник або команда вибирає власну технологію розробки відповідно до потреб замовника щодо майбутнього додатку.

1.4 Мова програмування Kotlin

Kotlin — це сучасна мова програмування з багатьма цілями, яка зародилася в JetBrains і швидко набула популярності серед розробників. Хоча ця мова розроблена для створення програм на Java, вона також працює з Android-розробкою, веб-розробкою та іншими цілями. Kotlin привабливий для широкого спектру завдань і проектів завдяки своїй виразності, безпеці типів і простоті використання.

Існує чудова сумісність Kotlin з мовою програмування Java. Це означає, що код Kotlin можна інтегрувати в існуючі Java-проекти без проблем. Схожий синтаксис і те, що Kotlin компілюється в байт-код, який може виконуватися віртуальною машиною Java, пояснюють цю сумісність. Це дозволяє розробникам поступово впроваджувати нові елементи в кодову базу, не переробляючи всю систему з нуля, що полегшує перехід на Kotlin.

Статична типізація Kotlin гарантує, що типи змінних не змінюються під час роботи програми, оскільки вони визначаються на етапі компіляції. Оскільки багато помилок і проблем можна виявити на ранніх стадіях розроблення, до того як програма буде запущена, це значно підвищує надійність коду. Ця функція Kotlin допомагає запобігти багатьом поширеним помилкам, таким як неправильне звернення до об'єктів (NullPointerException).

Функціональне програмування (FP) є важливою частиною функціонального програмування Kotlin. Він пропонує розробникам широкий спектр інструментів і конструкцій, включаючи замикання, функції вищого порядку та лямбда-вирази. Функціональне програмування спрощує код і робить

його більш виразним. Воно дозволяє створювати гнучкий і модульний код, дозволяючи використовувати функції як аргументи та значення, що повертаються.

Безпека

Система безпеки Kotlin є великою перевагою. Як було зазначено раніше, Kotlin пропонує інструменти для запобігання помилкам, допущеним під час процесу компіляції. Наприклад, змінні не можуть зберігати значення «нуль» без явної вказівки в Kotlin, оскільки там не існує поняття «нульоване за замовчуванням». Це дозволяє запобігти великій кількості помилок, пов'язаних із «нулевим показником», які часто є джерелом серйозних проблем в інших мовах.

Конструкції мови Kotlin пропонують розробникам широкий вибір конструкцій, щоб зробити роботу з даними та логікою програми простішою та ефективнішою. Це включає оператори присвоювання, умовні оператори (if, when), цикли (for, while) і багато інших інструментів. Якщо поєднати ці конструкції, код Kotlin стає більш простим для розуміння та підтримки.

Сфера використання Kotlin

Розробка додатків Android

Розробка мобільних додатків для платформи Android є однією з основних сфер застосування Kotlin. Kotlin стрімко набирає популярності серед Android-розробників після його оголошення офіційною мовою програмування для Android у 2017 році. Головні риси та переваги Kotlin:

- Офіційна підтримка Google: Google офіційно підтримує Kotlin в Android Studio, надаючи плагіни та інструменти, необхідні для успішного розробки програм під Android.

- Покращена безпека: Завдяки своїй системі типів і безпеки на етапі компіляції Kotlin допомагає уникнути багатьох поширених помилок, таких як NullPointerException.

•Скорочення обсягу коду: Kotlin порівняно з Java написаний коротший і виразніший код, що полегшує розробку та обслуговування додатків Android.

•Функціональне програмування: Kotlin ідеально підходить для розробки сучасних Android-додатків завдяки своїм функціональним можливостям.

Серверна розробка

Kotlin також є частиною серверної розробки. Можна використовувати Kotlin для створення мікросервісів, API, веб-сервісів і інших компонентів серверних додатків. Він добре працює з фреймворками та бібліотеками, такими як Spring і Ktor, що робить його придатним для розробки серверних застосунків, які мають високу продуктивність і надійність.

Додатки для комп'ютера

Kotlin не часто використовується для розробки desktop-додатків, але він надає можливості для створення програм, які працюють на різних платформах. Підпроєкт мови Kotlin, Kotlin/Native, дозволяє створювати програми для робочого столу, які працюють на різних операційних системах, таких як Windows, macOS і Linux. Це може бути корисно для розробки програм, які мають невелику базу клієнтів.

Веб-розробка також використовує Kotlin. За допомогою фреймворку Ktor можна створювати веб-додатки, які використовують Kotlin. Використовуючи переваги Kotlin, такі як функціональне програмування та безпека, це дозволяє розробникам створювати сучасні та високопродуктивні веб-додатки.

Для загальних завдань програмування, таких як створення скриптів, утиліт і інструментів, також можна використовувати загальні завдання програмування Kotlin. Його чудовий вибір для розробки різноманітних додатків і сценаріїв через його простий синтаксис і широкий спектр функціональних конструкцій.

Kotlin є мовою програмування з широким спектром застосування, що робить його універсальною мовою програмування, яка підходить для створення мобільних, серверних і інших типів додатків

Переваги програмування Kotlin

Kotlin має безліч переваг, які роблять цю мову привабливою для розробників:

•Збільшення продуктивності: тому що Kotlin дозволяє писати код компактніше та виразніше, процес розробки прискорюється. Kotlin також сприяє збільшенню продуктивності додатків завдяки статичній типізації та функціональним можливостям.

•Покращена безпека: Kotlin пропонує безліч інструментів для захисту коду. Це дозволяє уникати багатьох помилок на етапі компіляції, що скорочує час і витрати ресурсів на налагодження.

•Чистий і читабельний код: Завдяки своєму синтаксису та функціональним конструкціям Kotlin сприяє написанню чистішого та читабельнішого коду. Це полегшує підтримку та супровід проєкту.

•Екосистема: Kotlin полегшує розробку та розширення функціональності ваших додатків завдяки багатій екосистемі інструментів і бібліотек.

Недолік мови Kotlin

Хоча Kotlin має багато переваг, він також має деякі недоліки:

•Страх перед змінами: Перехід з Java на Kotlin може хвилювати деяких розробників. Це пов'язано з тим, що перенавчання та адаптація до нового синтаксису завжди супроводжують зміну мови програмування.

•Обмежене використання в деяких сферах: Kotlin може бути менш поширеним у порівнянні з іншими мовами в деяких сферах розробки, таких як системне програмування або розробка ігор.

Перспективи розробки на Kotlin

Постійний розвиток: активний розвиток самої мови є важливим компонентом перспектив розробки Kotlin. Kotlin продовжує отримувати розвиток і підтримку від компанії JetBrains, яка його створила. Розробники

постійно отримують нові версії Kotlin, які включають покращення та нові функції, що робить мову потужнішою та зручнішою для використання.

Розширення екосистеми: екосистема Kotlin продовжує розвиватися. З кожним роком виходять нові бібліотеки, фреймворки та інструменти, спеціально розроблені для Kotlin. Це полегшує розробку програм і розширює функціональність мови. Kotlin вже широко використовується в таких сферах, як мобільна (Android) розробка, веб-розробка, бекенд-розробка та навіть розробка ігор.

Зростання популярності: Kotlin стає все більш популярним серед розробників. Багато компаній перейшли на Kotlin або починають його використовувати в нових проектах. Це означає, що зростає попит на розробників Kotlin. Як роботодавці, так і компанії високо оцінюють здатність Kotlin покращувати продуктивність і універсальність.

Розвиток Android-платформи: важливо відзначити, що Kotlin став офіційною мовою розробки Android. Для розробників Google надає інструменти та ресурси для підтримки Kotlin в Android Studio. Це означає, що Kotlin продовжуватиме домінувати в галузі мобільної розробки на Android, і його популярність зростатиме разом із популярністю платформи Android.

Зручність для розробників: Kotlin гарантує, що розробники будуть відчувати себе комфортно та щасливо працювати. Він простий у читанні та має чистий синтаксис, що полегшує супровід коду. Розробники високо цінують безліч можливостей Kotlin для написання ефективнішого і виразнішого коду.

Підтримка спільноти: спільнота розробників Kotlin активна та гостинна. Багато експертів готові допомогти новачкам і поділитися своїм досвідом. Це сприяє професійному зростанню та навчанню.

Kotlin — потужна та гнучка мова програмування, яка надає розробникам безліч інструментів, щоб створювати надійні та ефективні програми. Її можна використовувати для розробки різноманітних програм, включаючи серверні та

мобільні. Незважаючи на деякі недоліки, Kotlin все ще є однією з найкращих мов програмування, і вивчення його може відкрити багато можливостей у сфері інформаційних технологій.

1.5 Загальна характеристика мобільних платформ:

У сучасному світі існує широкий спектр мобільних пристроїв. «Мобільний» — це те, що означає «транспортабельний», «переносний». Традиційно це мобільні телефони, планшети, електронні книги, нетбуки, навігатори, розумні годинники та плеєри, які знаходяться в нашій свідомості. В останні роки це також включає телевізори, кінотеатри та окуляри віртуальної реальності.

Мобільна операційна система,— це система, яка працює на смартфонах, планшетах, КПК та інших мобільних пристроїв. Мобільні операційні системи поєднують функції операційної системи ПК із функціями мобільних і кишенькових пристроїв, такими як Bluetooth, Wi-Fi, GPS-навігація, сенсорний екран, стільниковий зв'язок, відеокамера, розпізнавання мови, диктофон, музичний плеєр, NFC (ближній зв'язок) і інфрачервоне дистанційне управління.

Хоча ноутбуки також можна віднести до мобільних пристроїв, операційні системи, під управлінням яких вони працюють, зазвичай не вважаються мобільными. Це пов'язано з тим, що ці операційні системи розроблялися для стаціонарних настільних комп'ютерів, які традиційно не мали будь-яких спеціальних «мобільних» функцій, а також не потребували їх. Деякі нові операційні системи є гібридами того й іншого, що робить це помітно.

Смартфони та інші пристрої мобільного зв'язку мають дві операційні системи.

Дворівнева пропрієтарна операційна система реального часу, яка обслуговує радіоустаткування, доповнює основну програмну платформу взаємодії з користувачем. Дослідження показали, що шкідливі базові станции,

здатні контролювати мобільний пристрій, можуть атакувати такі низькорівневі операційні системи.

1.5.1 Перелік найбільш поширених і застарілих мобільних операційних систем:

Android, iOS, CyanogenMod, Cyanogen OS, Fire OS, Flyme OS, Firefox OS, Sailfish OS, Tizen, Ubuntu Touch та інші.

Програмні платформи, які вже не підтримуються в даний час, включають Windows Phone, BlackBerry OS, Symbian, Palm OS, webOS, Maemo, MeeGo та LiMo.

Розглянемо кілька з них більш детально.

1) Android — це операційна система та платформа на базі Linux, розроблена Google для мобільних телефонів і планшетів. Схвалено Open Handset Alliance (OHA).

Незважаючи на те, що Android базується на ядрі Linux, він досить далеко від спільноти Linux і інфраструктури Linux. Реалізація Dalvik — віртуальної машини Java — є основним компонентом цієї операційної системи, і на ній базується все програмне забезпечення та застосування.

Планшети та смартфони Samsung, NTTC, Huawei, SONY, LG і Lenovo є одними з найвідоміших на вітчизняному ринку.

У 2008 році на смартфоні HTC дебютувала перша версія операційної системи, і з тих пір вона постійно оновлюється.

Можливість встановлювати Android на пристрої різних виробників є основною причиною його популярності. Такі пристрої найбільш поширені в країнах, що розвиваються, таких як Азія, Індія, Африка та Європа.

Переваги операційної системи Android:

- Унікальна різноманітність програм і ігор на Android;
- швидка інтеграція з сервісами Google;

•повна незалежність від апаратної начинки мобільного пристрою;

•Android є екосистемою з відкритим кодом;

•багатозадачність, що дозволяє одночасно працювати з кількома програмами;

•просте встановлення програм із різних ресурсів;

•широкі можливості індивідуалізації;

•відсутність обмежень щодо вибору оператора мобільного пристрою;

•і підтримка флеш-програм;

Нова професія — розробник додатків Android — стрімко набирає обертів. Java, C# і Python вважаються найкращими мовами програмування для Android.

Недоліки операційної системи Android:

•Пристрої під керуванням операційною системою Android доводиться заряджати та підзаряджати досить часто;

•є деякі проблеми з сумісністю нових версій операційної системи з застарілими пристроями або з пристроями, випущеними маловідомими китайськими виробниками;

•користувачі, які цінують загальну швидкість роботи та налаштування, можуть не бути задоволені системою в цілому та її налаштуваннями.

2) IOC

На відміну від Windows Phone (Microsoft) і Android (Google), розроблена відомою компанією Apple виключно для своїх смартфонів, планшетів і інших пристроїв. Оперативна система була вперше представлена разом із телефоном iPhone у 2007 році. Спочатку була розроблена для iPhone і IPod Touch, а потім для iPad і Apple TV.

У зв'язку з тим, що ядро iOS майже ідентичне ядру операційної системи Apple MacOS (раніше відомої як OS X), його основні компоненти майже ідентичні.

Переваги iOS включають:

- Зручність роботи, приголомшливий дизайн,
- Акцент на надійності та якості операційної системи;
- Оптимізація збільшує тривалість роботи пристроїв при повному завантаженні, а також відсутність сповільнень і «глюків» у програмах і на робочому екрані пристроїв;
- Великий вибір додатків;
- Рідко зустрічається реклама в програмах, на відміну від Google Play;
- Оновлення доступні для всіх відразу після випуску
- Передбачена спеціальна панель меню, яка може виконувати багато завдань одночасно.

Недоліки iOS:

- Система є закритою та досить консервативної; за кількома винятками все програмне забезпечення є платним. Завдяки цьому цей варіант доступний для всіх пристроїв Apple, від смартфонів iPhone до планшетів iPad;
- Пряме копіювання файлів або передача з'явилася лише нещодавно, що означає, що переміщення файлів можливо лише за допомогою iTunes;
- В iOS все пов'язано з Інтернетом, тому користувачі не можуть використовувати багато функцій, якщо вони не підключені до Інтернету; і
- Працює лише на пристроях компанії Apple, що знач Найпоширенішими у багатих країнах є iPad, iPhone, iPod, Macbook, iMac і iWatch. Ці пристрої доступні в Сполучених Штатах, Європі та інших країнах.

Як операційна система iOS була представлена на Macworld Conference & Expo 9 січня 2007 року, а в червні того ж року була випущена. За словами Apple, вона просто заявила, що «iPhone використовує OS X». Спочатку сторонні програми не були підтримані. За словами Джобса 7, розробники можуть створювати веб-додатки, які «будуть вести себе, як рідні програми на iPhone». 17 жовтня 2007 року Apple повідомила, що їхній власний SDK знаходиться в процесі розробки та планує надати його «розробникам у лютому». 6 березня 2008 року Apple випустила першу бета-версію операційної системи та нову назву — iPhone OS. У результаті продажів мобільних пристроїв Apple SDK викликав інтерес. Під час курортного сезону 2007 року Apple також продала більше одного мільйона iPhone. У червні 2010 року Apple змінила назву свого iPhone OS на iOS.

3) Windows Phone

Операційна система для мобільних пристроїв, розроблена корпорацією Майкрософт у 2010 році. На стартовому екрані її легко визначити за своєрідними «живими» кольоровими плиточками.

Переваги Windows Phone:

•Windows є найпоширенішою операційною системою для ПК у світі, і мобільні пристрої, що працюють під Windows, легко синхронізуються з комп'ютером, ноутбуком або планшетом;

•достатня кількість програм, встановлених за замовчуванням;

•плавність інтерфейсу, висока швидкість запуску програм, перемикання між відкритими вікнами та інші функції;

•прості та зрозумілі налаштування; • підтримка великої кількості;

Проблеми з операційною системою Windows включають:

•мало додатків або вони не працюють належним чином;

•підтримка недоступна для деяких користувачів старих смартфонів;

Майже всі мобільні пристрої Nokia нового покоління, які замінили операційну систему Symbian, мають Windows Phone. Поширені мобільні телефони Nokia в Європі, Індії та Бразилії.

Інтерфейс операційної системи складається з шести панелей, витягнутих по горизонталі, які можна прокручувати ліворуч і праворуч на екрані мобільного пристрою.

Панель «Люди», також відома як «Люди», об'єднує фотографії, записи та коментарі в соціальних мережах, а також всю інформацію, пов'язану з людиною, надаючи централізований доступ до таких мереж, як Windows Live і Facebook.

Панель «Картинки» об'єднує фотографії та відеозаписи користувача, які зберігаються в пам'яті пристрою, в Інтернеті та в пам'яті комп'ютера, а також дозволяє отримати доступ до фотографій і відеозаписів друзів.

На панелі «Ігри» можна отримати доступ до мобільних ігор, досягнень, аватарів, які використовуються в Xbox Live, і профілів інших гравців.

«Музика+Відео» (Music+Video): об'єднує мультимедійний контент, що зберігається на комп'ютері користувача, музичні онлайн-сервіси та вбудоване FM-радіо. Крім того, він дозволяє користувачам отримати доступ до сервісу для обміну музикою Zune Social.

За допомогою панелі Marketplace можна завантажувати застосунки та ігри, а Office надає доступ до SharePoint, OneNote і Office Mobile. Користувач може відкрити, створити та редагувати документи.

4)Операційна система BlackBerry

Blackberry використовується лише на пристроях, розроблених канадською компанією, яка, відповідно, є найбільшою. Blackberry також стає все більш популярним у США, Австралії та Європі, особливо у Великій Британії. Програми Android встановлюються за допомогою спеціальної утиліти, розробленої для Blackberry.

Переваги Blackberry:

•висока якість, оскільки Blackberry позиціонується, як пристрої бізнес-класу;

•схожість з Android та iOS;

•гнучкість меню налаштувань; швидка синхронізація зі штатними засобами та хмарними рішеннями.

Недоліки операційної системи Blackberry:

•невелика кількість застосунків і рідкість їх оновлення.

Зосереджуючись на потребах бізнес-клієнтів, смартфони Blackberry пропонують найкращий захист персональних даних, а також зручний доступ до Інтернету та електронної пошти. Ці смартфони створені для того, щоб зробити ведення бізнесу простим, зручним і безпечним. Отже, RIM віддав перевагу фізичній QWERTY клавіатурі, яка є стандартною для майже всіх Blackberry. У сучасних моделях можна працювати з документами в таких форматах, як Word, PDF, Excel, PowerPoint, ASCII текст, HTML, WordPerfect і ZIP. Кодування характеристик AES захищає дані смартфонів Blackberry від перехоплення.

Таким чином, телефони Blackberry часто використовують у державних установах. Технологія PUSH-повідомлень, розроблена спеціально для пристроїв BlackBerry, дозволяє адресату отримувати всі e-mail повідомлення негайно, навіть без постійного з'єднання з сервером. Після відправлення листа на поштову скриньку оператор мобільного зв'язку сповіщає смартфон про надходження повідомлення. Тільки після цього смартфон з'єднується з сервером, щоб завантажити пошту.

5) Firefox OS

Мобільний операційний систем, розроблений Mozilla у липні 2012 року. Спочатку проект працював під назвою Boot to Gecko (B2G), але згодом вирішили, що операційна система буде продаватися під брендом Firefox, щоб зацікавити користувачів новими смартфонами. 26 липня 2011 представник

Фонду Mozilla оголосив про початок розробки операційної системи на основі рушії Gecko, який використовується в браузері Mozilla Firefox.

У лютому 2012 року Telefonica, іспанська компанія з телекомунікацій, працювала з Mozilla Foundation, щоб розробити проект Open Web Device з операційною системою Boot to Gecko.

Крім того, повідомляється про співпрацю з Qualcomm і Deutsche Telekom.

У грудні 2015 року з'явилася інформація про те, що проект платформи для смартфонів було закрито.

При цьому інші пристрої на цій платформі також можуть з'явитися. Після цього стало відомо, що Mozilla мала на увазі лише припинити співпрацю з операторами зв'язку. Сама операційна система для смартфонів залишиться у вигляді відкритої платформи, яка дозволяє виробникам пристроїв використовувати її, як вони хочуть. Мобільна платформа, розроблена в рамках проекту Firefox OS, базується на концепції використання браузерного оточення замість робочого стола. На відміну від ChromeOS, Firefox OS орієнтована на мобільні пристрої та пропонує розширений Web API для розробки спеціалізованих мобільних веб-застосунків, які використовують можливості сучасних телефонів.

Ядро Linux і компоненти низькорівневої платформи Android служать основою. Для запуску застосунків використовується веб-стек Mozilla замість віртуальної машини Dalvik. Набір веб-застосунків Gaia складає інтерфейс користувача платформи. Веб-браузер, календар-планувальник, калькулятор, адресна книга, програма для роботи з веб-камерою, додаток для роботи з електронною поштою, клієнт електронної пошти, система пошуку, музичний плеєр, програма для перегляду відео, інтерфейс SMS/MMS, конфігуратор, менеджер фотографій, робочий стіл і менеджер програм підтримують різні режими відображення елементів (карти та мережі).

Влітку 2014 року Mozilla зробила несподівану заяву, дозволивши програмам, які працюють під Firefox OS, працювати на смартфонах на Android. Щоб це зробити, потрібно було відкрити фірмовий магазин додатків у Firefox. По суті, це означає встановлення веб-сторінок у браузері, які не мають звичайного інтерфейсу з адресним рядком, а просто веб-сторінки. У цьому ринку було мало корисного програмного забезпечення, і рідне програмне забезпечення працювало гірше. Таким чином, ні Android, ні Firefox OS не стали кращими за це рішення.

Поглядаючи на хронологію розробки Firefox OS, легко помітити, що розробники до кінця не розуміли мети проекту: спочатку він виглядав як універсальна система для смартфонів різних категорій, потім лише для бюджетних, а потім ми розробляємо оболонку для кнопкових телефонів, телевізорів і «розумних» годинників. Хоча розробка проекту тривала лише чотири роки, що є надзвичайно коротким періодом, і очевидно, що на нього не витрачалося багато ресурсів. Ubuntu Touch, яка була створена майже одночасно з Firefox OS, також зазнала такої ж долі.

б) Sailfish

Це операційна система з відкритим кодом, але з закритим кодом інтерфейсу користувача та компонентами з відкритим кодом на базі ядра Linux, орієнтована на смартфони. З 2012 року компанія Jolla, заснована колишніми співробітниками Nokia, працює над розробкою нових смартфонів на базі Linux-платформи MeeGo у співпраці з проектом MeeGo і підтримкою Sailfish Alliance. Особливістю інтерфейсу Sailfish є активне використання екранних жестів для управління та вертикальна модель розміщення контенту. За допомогою екранних жестів гортання можна перейти від одного екрана до іншого, наприклад, «перегорнути» домашній екран, щоб перейти на екран зі списком застосунків або на екран з оглядом подій. При неповному зрушенні вмісту вниз на екрані можна отримати доступ до меню за допомогою жесту. Домашній екран служить інтерфейсом, який дозволяє швидко запускати та переходити між запущеними

програмами. Він відображає огляд запущених програм і інформацію про їхні активності.

В листопаді 2013 року компанія Jolla представила смартфон. Він має бюджетний процесор Snapdragon 400, 1 ГБ оперативної пам'яті та 16 ГБ постійної пам'яті. Він відрізнявся від Sailfish OS. У системі вже були включені найпопулярніші програми, такі як WhatsApp, Twitter, Skype і браузери Opera і Firefox.

Jolla вирішила продемонструвати універсальність Sailfish OS, випустивши свій перший планшет Jolla Tablet на краудфандингову платформу Indiegogo в листопаді 2014 року.

Залучено 2,5 мільйона доларів, хоча планувалося зібрати всього 380 тисяч. Проект був визнаний успішним.

2015 рік був поганим для Jolla. Розглядаючись як кросплатформова операційна система номер 11 для смартфонів, планшетів, комп'ютерів, телевізорів та інших пристроїв, таких як бортові системи автомобілів, ніхто з великих компаній не погодився замінити Android на Tizen, оскільки виникли проблеми з постачання компонентів для планшета. За основу платформи взяли HTML5, просту та перспективну мову, яка дозволяє створювати додатки, які легко підлаштовуються під різні пристрої. Журналісти зустріли Tizen скептично, оскільки це вже не перша мобільна Linux-система з відкритим кодом. Дуже важливо розглянути історію цього проекту через MeeGo, LiMo та попередні версії. Тим більше, що Tizen не мав привабливих ідей або дорогих рекламних кампаній. Хіба що Samsung організовувала конкурси серед розробників, які надавали значні грошові призи.

Компанія оголосила про завершений Tizen 2.0 під назвою Magnolia під час виставки MWC 2013. Хоча більш прийнятним було б називати всі попередні версії бета або альфа, а цей реліз позначити як 1.0. тому що основні зміни, такі як підтримка нативних додатків і повноцінний набір попередньо стандартних

програм, були додані лише в цьому оновленні. Щоправда, на той момент Tizen 2.0 продовжував працювати погано. Отже, їй не вдалося привернути увагу.

Незважаючи на обіцянки в 2013 році представити смартфони на Tizen, перший, Samsung Z, з'явився тільки влітку 2014 року. Перший смартфон з Tizen був неприємним для журналістів, оскільки він був одноманітним і мав систему, схожу на Android з оболонкою TouchWiz, але без більшості сторонніх додатків.

Крім того, нещодавно «розумні» годинники Samsung Galaxy Gear, які працюють на модифікованому Android (не Android Wear), отримали оновлення Tizen. Схоже, було легше встановити власну операційну систему на екран годинника, ніж використовувати смартфон Android, який не призначений для цього.

Після цього в серпні був представлений новий годинник Gear S із зігнутим дисплеєм, який працював на Tizen.

Важливо розуміти, що Tizen змінювався залежно від пристрою; наприклад, система на смартфоні, ноутбуці, годинниках або камері виглядала та працювала по-іншому. Це окремі операційні системи для користувача, які ніяк не пов'язані. Хоча ці смартфони продавалися компаніям, ніхто не перевіряв безпеку Tizen, оскільки він був непопулярним. У квітні 2017 року дослідник інформаційної безпеки Аміхай Нейдерман опублікував великий звіт після вивчення Tizen. У ньому він виявив сорок проблем із нульовим днем і назвав його «можливо, найгіршим кодом, який коли-небудь бачив у житті».

Дослідження стверджує, що кожна з цих вразливостей дозволяє хакерам віддалено отримати доступ до смартфону або будь-якого іншого пристрою на базі Tizen. Однією з таких вразливостей є можливість завантажити будь-яке шкідливе програмне забезпечення на телевізор Samsung. Після публікації Samsung не заперечувала проблему і зв'язалася з Нейдерманом, щоб виправити будь-які недоліки.

Тизен став альтернативою Samsung у випадку проблем з Android. Наприклад, Huawei не розробив жодних альтернативних рішень, оскільки Harmony OS або Hongmeng компанії все ще не підходить для роботи зі смартфонами. Таким чином, китайська компанія змушена випускати флагман Mate 30 на базі Android без послуг Google.

Хоча Samsung поступово випускає оновлення Tizen, видно, що вони не хочуть витрачати багато часу та грошей на це. Все не так погано, оскільки «розумні» годинники та дванадцять телевізорів є значно активнішою системою.

РОЗДІЛ 2. ПРОЄКТУВАННЯ

Програма «список покупок» на Android робить створення та управління списками покупок простішим. Нижче наведено детальний опис потенційного функціоналу цієї програми:

1.Основні функції, пов'язані зі створенням і редагуванням списків

•Створення нового списку: Користувач може створити новий список покупок, давши йому назву.

•Редагування списку: тут ви можете змінити назву товару, додати або видалити його.

•Додавання товарів: простий інтерфейс, за допомогою якого ви можете додати товари до списку. Ви можете вказати кількість, категорію (наприклад, продукти харчування, побутова хімія тощо) та одиницю виміру (наприклад, кілограми, літри тощо).

•Видалення товарів: простий спосіб видалити товари зі списку.

Управління товарами:

•Відмічання придбаних товарів: Можливість відмітити куплені товари. Означені товари можуть залишатися в списку, або вони можуть перейти в інший розділ, наприклад «Куплені».

•Сортування товарів: сортування за пріоритетом, категорією або алфавітним порядком.

2.Додаткові функції

Нагадування та оповіщення

•Нагадування: встановлення нагадувань для певних списків або товарів (наприклад, купити молоко після роботи).

•Сповіщення: Push-сповіщення про невиконані покупки або про те, що час нагадування наближається.

Шалони та мої покупки

•Шаблони списків: використовуйте ці шаблони для списків, які часто використовуються.

•Історія покупок: можна відстежувати історію покупок товарів і швидко додавати їх до поточного списку з минулих списків.

Спільні списки

•Спільні списки: це можливість поділитися списком з іншими користувачами через електронну пошту, месенджери або прямо в програмі.

•Синхронізація в реальному часі: синхронізує списки між користувачами та різними пристроями для спільного використання.

3.Додаткові можливості

Можливості для голосового введення та сканування штрих-кодів є додатковим бонусом.

•Голосове введення: Для зручності підтримується голосове введення товарів.

•Сканування штрих-кодів: це функція, яка дозволяє додавати продукти до списку за допомогою сканування штрих-кодів.

Підключення до додаткових послуг

•Інтеграція з календарем: є можливість синхронізувати нагадування з календарем.

•Інтеграція з рецептами: можливість включати інгредієнти для рецептів у список покупок, створений кулінарними програмами або веб-сайтами.

4.Налаштування та персоналізація:

•Налаштування теми інтерфейсу підтримує різні теми оформлення, такі як світло, темна та користувацька.

•Налаштування шрифтів: виберіть стиль і розмір шрифту, щоб зробити шрифт зручним для читання.

•Персональні налаштування пріоритетів товарів: встановіть пріоритети для різних товарів, щоб було простіше сортувати.

•Категорії: налаштування категорій товарів відповідно до уподобань користувача

5.Безпека даних і зберігання

Зберігання та відновлення інформації

•Хмарне збереження: дозволяє спискам бути доступними з різних пристроїв у хмарі.

•Резервне копіювання: можливість створення та відновлення резервних копій списків

Цей функціонал робить управління покупками простим і ефективним, а процес менш стресовим для користувача.

6.Розширені можливості

Можливості для автоматики та інтелектуальних функцій

•Автоматичне додавання товарів: визначає товари, які купуються часто, і автоматично додає ці товари до списку.

•Рекомендації: Пропозиція продуктів базується на попередніх покупках і сезонності.

•Інтеграція з голосовими асистентами: Підтримка голосових команд для Google Assistant або Amazon Alexa.

Контроль бюджету та сканування чеків

•Сканування чеків: можливість автоматичного додавання товарів, які ви купили, до історії покупок або списку покупок.

•Бюджетування: відстеження витрат, планування покупок і аналіз витрат за категоріями

7.Взаємодія з магазинами: пошук магазинів і планування маршрутів

•Мапи магазинів: дозволяють переглядати найближчі магазини та побудувати маршрути.

•Акції та знижки: показують актуальні знижки та акції в найближчих магазинах.

•Онлайн-покупки Інтеграція з онлайн-магазинами: можливість безпосередньо використовувати програму для додавання продуктів до кошика в онлайн-магазинах.

•Порівняння цін: це функція, яка дозволяє вам порівнювати ціни різних товарів у різних магазинах.

8.Люди та соціальні функції

Купівля в Інтернеті

•Список покупок для груп: це дає можливість створювати списки для різних груп користувачів, таких як колеги та члени сім'ї.

•Завдання та розподіл: розподіл завдань щодо придбання товарів конкретним учасникам групи.

Інтернет

•Ділення рецептами: можливість ділитися рецептами або списками покупок через соціальні мережі.

•Оцінки та рейтинги товарів: користувачі залишили відгуки та оцінки цього товару.

9.Додаткові інтеграції та API

•Підтримка API Відкрите API: це API, яку можна використовувати для інтеграції з іншими сервісами або додатками.

•Інтеграція з фінансовими додатками дає вам можливість керувати своїми витратами на покупки з фінансовими додатками.

•Зовнішні сервіси: інтеграція з фітнес-трекерами: автоматичне додавання здорових продуктів до списку за допомогою синхронізації з фітнес-додатками.

•Партнерство з виробниками: співпраця з виробниками товарів для отримання акцій або знижок, які недоступні для інших.

10.Аналітика та звітність

•Персоналізація досвіду користувача Статистика покупок: детальна статистика витрат, кількості куплених товарів і середньої вартості.

•Персональні звіти: створення індивідуальних звітів про покупки, щоб оцінити витрати та споживання.

•Адаптивний інтерфейс Гнучкі налаштування: дозволяє налаштовувати інтерфейс відповідно до потреб кожного користувача.

•Підтримка багатьох мов: багатомовність, щоб було зручно використовувати в різних країнах.

11.Зворотний зв'язок і підтримка

•Служба підтримки Чат підтримки: вбудований чат дозволяє швидко спілкуватися з технічною підтримкою.

•Часті запитання та допомога: Розділ містить відповіді на найпоширеніші запитання та інструкції щодо використання програми.

Відгуки та оцінка

•Зворотний зв'язок: є можливість залишити коментарі та пропозиції щодо того, як можна покращити програму.

•Оцінка функціоналу: Користувачі можуть оцінити різні функції програми, що допомагає розробникам визначити, які оновлення є пріоритетними.

12.Безпека та конфіденційність

Шифрування захищає дані користувачів.

- Двофакторна аутентифікація: Двофакторна аутентифікація підтримується для підвищення безпеки.
- Політика конфіденційності Прозорість: детальна інформація про політику конфіденційності та використання даних користувачів.
- Контроль над даними: дозволяє видалити обліковий запис і всі його дані.

РОЗДІЛ 3. ОПИС РОБОТИ

3.1 Чому саме Андроїд

Сьогодні Android використовується понад 88% смартфонів. Зважаючи на те, що шість мільярдів людей володіють смартфонами, приблизно п'ять мільярдів із них використовують Android. Ви бачите привабливість Android.

Якщо припустити, що двадцять відсотків власників Android прагнуть стати розробниками, це означає, що один мільярд людей працює або бажає працювати в проектах **Android.**

MVVM

Давайте спочатку розглянемо переваги MVVM.

Основна причина використання MVVM*(Model-View-View-Model)* полягає в тому, що він відокремлює наше представлення (Дії та фрагменти) від нашої бізнес-логіки, що має величезну різницю. Для невеликих проектів впровадження MVVM достатньо. Але з часом наш проект стає все більш заплутаним і складним. модель перегляду стане настільки складною, що ми не зможемо визначити її Clean Architecture допоможе у цій ситуації.

Clean Architecture

Одним словом, ваш код у Clean Architecture розташований у формі цибулі з одним правилом залежності: внутрішні шари не повинні знати про зовнішні шари.

Шаблон «Clean Architecture» Роберта С. Мартіна чудово підходить для написання роз'єднаного коду, оскільки він дозволяє розділити вашу взаємодію з даними на простіші сутності, які називаються «випадками використання».(рис.3.1, ст.39)

Таким чином, ми зможемо точно визначити, як працює кожен компонент, без будь-яких проблем, що збільшує час виробництва, що завжди є перевагою.

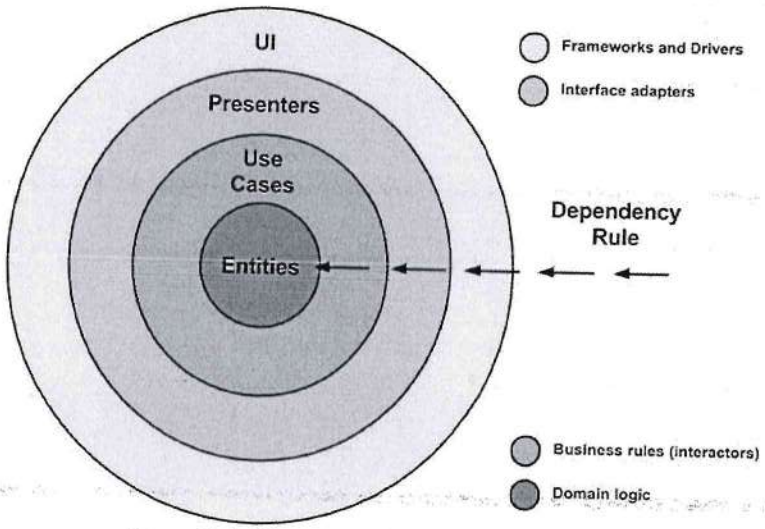

Рис 3.1 Схема Clean Architecture

Нічого немає ідеального. Ситуація з цією архітектурою також така ж. Ніжче наведено переваги та недоліки чистої архітектури. Зважте їх залежно від ваших потреб.

Різні рівні чистої архітектури MVVM:

1: Рівень демонстрації

2: Рівень домінування

3: Рівень обробки даних

Рівень представлення

Наші дії, фрагменти та модель перегляду містяться на цьому шарі. Ми навіть не використовуємо бізнес-логіку в своєму бізнесі.

Щоб виконати необхідні дії, програма переходить від дії до моделі представлення та знову від моделі представлення до рівня домену. Рівень даних ніколи не спілкується з моделлю представлення.

Приклад, у якому ми додаємо два варіанти використання до нашого ViewModel, ми розглянемо. В основному дія UseCase визначає, як відбувається зв'язок між ViewModel і рівнем даних.(рис.3.2)

```
class PostListViewModel(
        val useCaseHandler: UseCaseHandler,
        val getPosts: GetPosts,
        val savePost: SavePost): ViewModel() {
    fun getAllPosts(userId: Int, callback:
PostDataSource.LoadPostsCallback) {
        val requestValue = GetPosts.RequestValues(userId)
        useCaseHandler.execute(getPosts, requestValue, object :
        UseCase.UseCaseCallback<GetPosts.ResponseValue> {
            override fun onSuccess(response: GetPosts.ResponseValue)
{
                callback.onPostsLoaded(response.posts)
            }
            override fun onError(t: Throwable) {
                callback.onError(t)
            }
        })
    }
    fun savePost(post: Post, callback:
PostDataSource.SaveTaskCallback) {
        val requestValues = SavePost.RequestValues(post)
        useCaseHandler.execute(savePost, requestValues, object :
        UseCase.UseCaseCallback<SavePost.ResponseValue> {
            override fun onSuccess(response: SavePost.ResponseValue)
{
                callback.onSaveSuccess()
            }
            override fun onError(t: Throwable) {
                callback.onError(t)
            }
        }
    }
}
```

Рис 3.2 Рівень представлення

Рівень зміни

У цьому шарі містяться всі можливі застосування нашого проекту. У цьому випадку ми розглянемо абстрактний клас під назвою UseCase.(рис.3.3)

```
abstract class UseCase<Q : UseCase.RequestValues, P :
UseCase.ResponseValue> {
    var requestValues: Q? = null
    var useCaseCallback: UseCaseCallback<P>? = null
    internal fun run() {
        executeUseCase(requestValues)
    }
    protected abstract fun executeUseCase(requestValues: Q?)
    /**
     * Data passed to a request.
     */
    interface RequestValues
    /**
     * Data received from a request.
     */
    interface ResponseValue
    interface UseCaseCallback<R> {
        fun onSuccess(response: R)
        fun onError(t: Throwable)
    }
}
```

Рис 3.3 Рівень зміни

У цьому випадку варіанти використання виконуються обробником варіантів використання. У цьому місці ми можемо виконати наш UseCase у фоновому потоці та отримати відповідь у основному потоці.

Рівень інформації

Усі репозиторії, які можуть використовувати доменний рівень, містяться на цьому рівні. Цей рівень більше схожий на API для зовнішніх джерела даних класів(рис3.4)

```
interface PostDataSource {
    interface LoadPostsCallback {
        fun onPostsLoaded(posts: List<Post>)
        fun onError(t: Throwable)
    }
    interface SaveTaskCallback {
        fun onSaveSuccess()
        fun onError(t: Throwable)
    }
    fun getPosts(userId: Int, callback: LoadPostsCallback)
    fun savePost(post: Post)
}
```

Рис 3.4 Рівень інформації

Давайте створимо PostDataRepository, який реалізує PostDataStructure. Ця структура визначатиме, чи отримаємо ми дані з віддаленого сервера чи локальної бази даних.(рис.3.5)

```
class PostDataRepository (
        private val localSourceItIs: PostDataSource,
        private val letsGoWithRemoteDataSource: PostDataSource):
PostDataSource {

    companion object {
        private var INSTANCE: PostDataRepository? = null
        fun getInstance(localDataSource: PostDataSource,
                        remoteDataSource: PostDataSource):
PostDataRepository {
            if (INSTANCE == null) {
                INSTANCE = PostDataRepository(localDataSource,
remoteDataSource)
            }
            return INSTANCE!!
        }
    }

    var isCacheDirty = false
    override fun getPosts(userId: Int, callback:
PostDataSource.LoadPostsCallback) {
        if (isCacheDirty) {
            getPostsFromServer(userId, callback)
        } else {
            localDataSource.getPosts(userId, object :
PostDataSource.LoadPostsCallback {
                override fun onPostsLoaded(posts: List<Post>) {
                    refreshCache()
                    callback.onPostsLoaded(posts)
                }
                override fun onError(t: Throwable) {
                    getPostsFromServer(userId, callback)
                }
            })
        }
    }
    override fun savePost(post: Post) {
        localDataSource.savePost(post)
        remoteDataSource.savePost(post)
    }
    private fun getPostsFromServer(userId: Int, callback:
PostDataSource.LoadPostsCallback) {
        remoteDataSource.getPosts(userId, object :
PostDataSource.LoadPostsCallback {
            override fun onPostsLoaded(posts: List<Post>) {
                refreshCache()
                refreshLocalDataSource(posts)
                callback.onPostsLoaded(posts)
```

Рис 3.5 PostDataRepository 1част.

```
            }
            override fun onError(t: Throwable) {
                callback.onError(t)
            }
        })
    }
    private fun refreshLocalDataSource(posts: List<Post>) {
        posts.forEach {
            localDataSource.savePost(it)
        }
    }
    private fun refreshCache() {
        isCacheDirty = false
    }
}
```

Рис 3.6 PostDataRepository 2част.

Цей клас містить дві змінні: localSourceItIs і letsGoWithRemoteDataSource. Вони успадковують PostDataSource.

3.2 Android Studio

Програма ADT для Eclipse була змінена Android Studio. Середовище базується на вихідному коді IntelliJ IDEA Community Edition, розробленому

JetBrains. Android Studio доступний під ліцензією Apache 2.0 і створений в рамках відкритої моделі розробки.

Бінарні складання доступні для Linux (Ubuntu використовувався для тестування), macOS і Windows. Середовище дозволяє програмістам створювати засоби для смартфонів і планшетів, а також для носимих пристроїв Wear OS, телевізорів (Android TV), окулярів Google Glass і інформаційно-розважальних систем автомобілів (Android Auto). Підготовлений інструмент для автоматичного імпорту поточного проекту в Android Studio для застосунків, спочатку розроблених за допомогою Eclipse і ADT Plugin.

Середовище розробки змінено, щоб воно могло виконувати стандартні завдання, вирішені під час процесу розробки застосунків для платформи Android. У середовище включені інструменти для проектування додатків для пристроїв з різною роздільністю екрана, таких як планшети, ноутбуки, смартфони та окуляри, а також засоби для спрощення тестування програм на сумісність з різними версіями платформи. Android Studio підтримує використання засобів безперервної інтеграції та нова уніфікована підсистема складання, тестування та розгортання застосунків, заснована на складальному інструментарії Gradle, окрім інших функцій, які доступні в IntelliJ IDEA.

Для прискорення розробки застосунків доступний візуальний редактор і набір типових елементів інтерфейсу для компонування застосунку, який надає зручний попередній перегляд різних станів інтерфейсу застосунку (наприклад, можна побачити, як інтерфейс буде виглядати для різних версій Android і розмірів екрану). Майстер створення власних елементів оформлення, який підтримує використання шаблонів, присутній для створення нестандартних інтерфейсів. Середовище містить функції, які дозволяють завантажувати стандартні приклади коду з GitHub.

До комплекту також входять розширені інструменти рефакторингу, перевірки сумісності з минулими випусками, виявлення проблем з продуктивністю, відстеження споживання пам'яті та оцінки зручності

використання, які адаптовані до особливостей платформи Android. Режим швидкого внесення правок доданий до редактора. Підтримка Android API покращує систему виявлення помилок, статичного аналізу та підсвічування. Підтримка оптимізатора коду ProGuard вбудована в систему. Включені інструменти для створення цифрових підписів. Надається інтерфейс управління перекладами.

3.3 Архітектура коду

Загальна структура файлу конфігурації: кореневий елемент <component name="ProjectCodeStyleConfiguration"> починає файл конфігурації. Усі налаштування стилю коду проекту містяться в цьому елементі. Він складається з двох основних частин: налаштування загального стилю коду для проекту та специфічні налаштування для різних мов програмування, таких як XML і Kotlin.

Загальні налаштування стилю коду

```
<code_scheme name="Project" version="173">
 <JetCodeStyleSettings>
  <option name="CODE_STYLE_DEFAULTS" value="KOTLIN_OFFICIAL" />
 </JetCodeStyleSettings>
```

Ця частина містить загальне налаштування стилю коду для проекту.

- Ім'я схеми коду для проекту визначається таким чином: code_scheme name="Project".
- Вказує версію схеми.
- JetCodeStyleSettings містить налаштування для Kotlin.
- Можна задати значення за замовчуванням для стилю коду Kotlin, використовуючи офіційні рекомендації, за допомогою опції name="CODE_STYLE_DEFAULTS" value="KOTLIN_OFFICIAL".

Це гарантує, що всі проектні файли Kotlin дотримуватимуться офіційних стандартів форматування, що сприяє підтримці єдиного стилю коду

Налаштування XML

```
<codeStyleSettings language="XML">
 <indentOptions>
  <option name="CONTINUATION_INDENT_SIZE" value="4" />
 </indentOptions>
```

Цей розділ займається налаштуванням формату XML-файлів. Назва налаштувань коду стилю «XML»

- codeStyleSettings language="XML"-вказує, що ці налаштування застосовуються до мови XML.
- indentOptions — це місце, де можна вибрати опції відступів.

Розмір відступу для продовження рядка на чотири пробіли визначається параметром «CONTINUATION_INDENT_SIZE». Це означає, що коли рядок коду занадто довгий і ділиться на кілька рядків, кожен наступний рядок отримає відступ у чотири пробіли.

Правила впорядкування атрибутів:для покращення читабельності та підтримки коду важливо мати стандартизований порядок атрибутів у XML. Ця конфігурація містить ряд правил, які регулюють порядок атрибутів.

```
<arrangement>
 <rules>
  <section>
   <rule>
    <match>
     <AND>
      <NAME>xmlns:android</NAME>
```

```
    <XML_ATTRIBUTE />
    <XML_NAMESPACE>^$</XML_NAMESPACE>
   </AND>
  </match>
 </rule>
</section>
```

Це правило перше визначає порядок атрибутів xmlns:android.

- <Match> описує вимоги, які повинні відповідати атрибутам.
- Використовуйте "AND", щоб об'єднати кілька критеріїв.
- «NAME» означає, що це правило стосується атрибуту xmlns:android.
- Правило XML_ATTRIBUTE вказує, що воно стосується XML-атрибутів.
- Указує, що атрибут не повинен належати до жодного простору імен (namespace).

Інші види атрибутів організовуються відповідно до певних критеріїв за допомогою різних правил:

Атрибути, що починаються з xmlns:

```
<section>
 <rule>
  <match>
   <AND>
    <NAME>xmlns:.*</NAME>
    <XML_ATTRIBUTE />
    <XML_NAMESPACE>^$</XML_NAMESPACE>
   </AND>
```

```
  </match>
  <order>BY_NAME</order>
 </rule>
</section>
```

Це правило присвоює імена всім атрибутам, що починаються з xmlns:

Атрибути, які належать до простору імен Android з тегами :id і :name:

```
<section>
 <rule>
  <match>
   <AND>
    <NAME>.*:id</NAME>
    <XML_ATTRIBUTE />

<XML_NAMESPACE>http://schemas.android.com/apk/res/android</XML_NAME
SPACE>
   </AND>
  </match>
 </rule>
</section>
<section>
 <rule>
  <match>
   <AND>
    <NAME>.*:name</NAME>
```

```
    <XML_ATTRIBUTE />

<XML_NAMESPACE>http://schemas.android.com/apk/res/android</XML_NAME
SPACE>
    </AND>
   </match>
  </rule>
</section>
```

Інші характеристики без простору імен:

```
<section>
 <rule>
  <match>
   <AND>
    <NAME>name</NAME>
    <XML_ATTRIBUTE />
    <XML_NAMESPACE>^$</XML_NAMESPACE>
   </AND>
  </match>
 </rule>
 <rule>
  <match>
   <AND>
    <NAME>style</NAME>
    <XML_ATTRIBUTE />
```

```
    <XML_NAMESPACE>^$</XML_NAMESPACE>
   </AND>
  </match>
 </rule>
 <rule>
  <match>
   <AND>
    <NAME>.*</NAME>
    <XML_ATTRIBUTE />
    <XML_NAMESPACE>^$</XML_NAMESPACE>
   </AND>
  </match>
  <order>BY_NAME</order>
 </rule>
</section>
```

Інші функції, доступні в просторі ім'я Android або будь-якому іншому просторі ім'я:

```
<section>
 <rule>
  <match>
   <AND>
    <NAME>.*</NAME>
    <XML_ATTRIBUTE />
```

```
<XML_NAMESPACE>http://schemas.android.com/apk/res/android</XML_NAME
SPACE>
    </AND>
   </match>
   <order>ANDROID_ATTRIBUTE_ORDER</order>
  </rule>
</section>
<section>
  <rule>
   <match>
    <AND>
     <NAME>.*</NAME>
     <XML_ATTRIBUTE />
     <XML_NAMESPACE>.*</XML_NAMESPACE>
    </AND>
   </match>
   <order>BY_NAME</order>
  </rule>
</section>
```

Цей набір правил гарантує, що всі атрибути XML будуть організовані за певним шаблоном, що полегшує роботу з кодом.

Налаштування Kotlin

```
<codeStyleSettings language="kotlin">
  <option name="CODE_STYLE_DEFAULTS" value="KOTLIN_OFFICIAL" />
```

</codeStyleSettings>

В цьому розділі знову використовується офіційний стиль кодування мови Kotlin.

- Назва налаштувань коду стилю «kotlin» вказує, що ці налаштування використовуються для мови Kotlin.

- Використання офіційного стилю кодування для Kotlin визначається параметром «CODE_STYLE_DEFAULTS», який має значення «KOTLIN_OFFICIAL».

Висновок Цей конфігураційний файл налаштовує стиль коду проекту: Для мови Kotlin використовується офіційний стиль кодування.

- Деталі правил форматування XML, включаючи відступи та порядок атрибутів, визначені.

- Підтримка єдиного стилю коду покращується за допомогою використання стандартних правил форматування, що покращує читабельність, зрозумілість і легкість підтримки коду.

Таке налаштування гарантує, що всі розробники дотримуються однакових правил кодування, що робить його корисним для командної роботи над проектом.Весь код можна побачити в Додатку А.

ВИСНОВОК

Розробка програми «Список покупок» для Android на мові програмування Kotlin демонструє сучасний підхід до створення мобільних додатків з використанням новітніх технологій і інструментів. Kotlin ідеально підходить для розробки через його інтеграцію з Android Studio, підвищену безпеку коду та простоту використання, що значно полегшує процес.

Основні функції, необхідні для повноцінної роботи програми, виконувалися під час розробки було: створення, редагування та видалення елементів списку, сортування та можливість позначати покупки, які були зроблені. Використання архітектурного патерну MVVM (Model-View-ViewModel) спрощує тестування та підтримку коду, забезпечуючи чітке розділення логіки програми.

Крім того, було підкреслено, наскільки простим є додаток для використання та наскільки він ефективний. Принципи матеріального дизайну використовувалися при розробці інтерфейсу користувача, що покращує взаємодію з програмою та робить її простішою для розуміння.

Таким чином, програма «список покупок», яка була розроблена, є надійним і корисним інструментом для щоденного використання, і вона повністю відповідає сучасним вимогам до мобільних додатків. Додаток легко розширювати та підтримувати завдяки Kotlin, що гарантує, що воно залишається актуальним і адаптується до нових потреб користувачів протягом усього життя.

СПИСОК ЛІТЕРАТУРИ

1.Програми для списку покупок в Android [Електронний ресурс]. – Режим доступу до ресурсу: https://daad.org.ua/3126-dodatki-dlya-spisku-pokupok-na-androyid.html

2.MVVM + Clean Architecture-Android [Електронний ресурс]. – Режим доступу до ресурсу: https://medium.com/@binsdreams/mvvm-clean-architecture-android-3a3899fed973

3.Android studio [Електронний ресурс]. – Режим доступу до ресурсу: https://developer.android.com/studio

4.Kotling [Електронний ресурс]. – Режим доступу до ресурсу: https://kotlinlang.org

5.IntelliJ IDEA wiki [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/IntelliJ_IDEA

6.IOS wiki [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/IOS

7.FP wiki[Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Функційне_програмування

8.API wiki [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Прикладний_програмний_інтерфейс

9.Wi-fi wiki [Електронний ресурс]. – Режим доступу до ресурсу: wiki https://uk.wikipedia.org/wiki/Wi-Fi

10.GPS wiki [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/GPS

11.КПК wiki [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/КПК

12.NFC wiki [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Near-field_communication

13.OS wiki [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Операційна_система

14.SDK wiki [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/SDK

15.ПК wiki [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Персональний_комп%27ютер

16.FM-радіо wiki [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/FM-радіо

17.PDF wiki [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/PDF

18.ASCII wiki [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/ASCII

19.ZIP wiki [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Zip

20.AES wiki [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Advanced_Encryption_Standard

21.Web wiki [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Всесвітнє_павутиння

22.ГБ wiki [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Гігабайт

23.HTML5 wiki [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/HTML5

24.MWC wiki [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Mobile_World_Congress

25.MVVM wiki [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Model-View-ViewModel

26.XML wiki [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/XML

27. IntelliJ IDEA wiki [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/IntelliJ_IDEA

ДОДАТОК А

Код програми

Код файлу Projeck.xml:

```
<component name="ProjectCodeStyleConfiguration">
 <code_scheme name="Project" version="173">
  <JetCodeStyleSettings>
   <option name="CODE_STYLE_DEFAULTS" value="KOTLIN_OFFICIAL" />
  </JetCodeStyleSettings>
  <codeStyleSettings language="XML">
   <indentOptions>
    <option name="CONTINUATION_INDENT_SIZE" value="4" />
   </indentOptions>
   <arrangement>
    <rules>
     <section>
      <rule>
       <match>
        <AND>
         <NAME>xmlns:android</NAME>
         <XML_ATTRIBUTE />
         <XML_NAMESPACE>^$</XML_NAMESPACE>
        </AND>
       </match>
      </rule>
```

```
</section>
<section>
 <rule>
  <match>
   <AND>
    <NAME>xmlns:.*</NAME>
    <XML_ATTRIBUTE />
    <XML_NAMESPACE>^$</XML_NAMESPACE>
   </AND>
  </match>
  <order>BY_NAME</order>
 </rule>
</section>
<section>
 <rule>
  <match>
   <AND>
    <NAME>.*:id</NAME>
    <XML_ATTRIBUTE />
<XML_NAMESPACE>http://schemas.android.com/apk/res/android</XML_NAME
SPACE>
   </AND>
  </match>
```

```
        </rule>
      </section>
      <section>
       <rule>
        <match>
         <AND>
          <NAME>.*:name</NAME>
          <XML_ATTRIBUTE />
<XML_NAMESPACE>http://schemas.android.com/apk/res/android</XML_NAME
SPACE>
         </AND>
        </match>
       </rule>
      </section>
      <section>
       <rule>
        <match>
         <AND>
          <NAME>name</NAME>
          <XML_ATTRIBUTE />
          <XML_NAMESPACE>^$</XML_NAMESPACE>
         </AND>
        </match>
```

```
  </rule>
 </section>
 <section>
  <rule>
   <match>
    <AND>
     <NAME>style</NAME>
     <XML_ATTRIBUTE />
     <XML_NAMESPACE>^$</XML_NAMESPACE>
    </AND>
   </match>
  </rule>
 </section>
 <section>
  <rule>
   <match>
    <AND>
     <NAME>.*</NAME>
     <XML_ATTRIBUTE />
     <XML_NAMESPACE>^$</XML_NAMESPACE>
    </AND>
   </match>
   <order>BY_NAME</order>
  </rule>
```

```
</section>
<section>
 <rule>
  <match>
   <AND>
    <NAME>.*</NAME>
    <XML_ATTRIBUTE />

<XML_NAMESPACE>http://schemas.android.com/apk/res/android</XML_NAME
SPACE>
   </AND>
  </match>
  <order>ANDROID_ATTRIBUTE_ORDER</order>
 </rule>
</section>
<section>
 <rule>
  <match>
   <AND>
    <NAME>.*</NAME>
    <XML_ATTRIBUTE />
    <XML_NAMESPACE>.*</XML_NAMESPACE>
   </AND>
  </match>
```

```
            <order>BY_NAME</order>
          </rule>
        </section>
      </rules>
    </arrangement>
  </codeStyleSettings>
  <codeStyleSettings language="kotlin">
    <option name="CODE_STYLE_DEFAULTS" value="KOTLIN_OFFICIAL" />
  </codeStyleSettings>
 </code_scheme>
</component>
```

Код файлу codeStyleConfig.xml:

```
<component name="ProjectCodeStyleConfiguration">
 <state>
  <option name="USE_PER_PROJECT_SETTINGS" value="true" />
 </state>
</component>
```

Код файлу gradle.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<project version="4">
 <component name="GradleMigrationSettings" migrationVersion="1" />
 <component name="GradleSettings">
  <option name="linkedExternalProjectsSettings">
   <GradleProjectSettings>
```

```
      <option name="externalProjectPath" value="$PROJECT_DIR$" />
      <option name="gradleJvm" value="jbr-17" />
      <option name="modules">
       <set>
        <option value="$PROJECT_DIR$" />
        <option value="$PROJECT_DIR$/app" />
       </set>
      </option>
      <option name="resolveExternalAnnotations" value="false" />
     </GradleProjectSettings>
   </option>
  </component>
</project>
```

Код файлу misc.xml:

```
<project version="4">
  <component name="ProjectRootManager" version="2" languageLevel="JDK_1_8"
project-jdk-name="1.8" project-jdk-type="JavaSDK">
    <output url="file://$PROJECT_DIR$/build/classes" />
  </component>
  <component name="ProjectType">
    <option name="id" value="Android" />
  </component>
</project>
```

Код файлу modules.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<project version="4">
 <component name="ProjectModuleManager">
  <modules>
   <module fileurl="file://$PROJECT_DIR$/.idea/Grocerylist.iml"
filepath="$PROJECT_DIR$/.idea/Grocerylist.iml" />
  </modules>
 </component>
</project>
```

Код файлу vcs.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project version="4">
 <component name="VcsDirectoryMappings">
  <mapping directory="" vcs="Git" />
 </component>
</project>
```

Код файлу workspace.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project version="4">
 <component name="AutoImportSettings">
  <option name="autoReloadType" value="NONE" />
 </component>
 <component name="ChangeListManager">
```

```
    <list default="true" id="fb7a28d7-19bc-4d97-85cb-8c6b3e74d32f"
  name="Changes" comment="" />
    <option name="SHOW_DIALOG" value="false" />
    <option name="HIGHLIGHT_CONFLICTS" value="true" />
    <option name="HIGHLIGHT_NON_ACTIVE_CHANGELIST" value="false" />
    <option name="LAST_RESOLUTION" value="IGNORE" />
  </component>
  <component name="ClangdSettings">
    <option name="formatViaClangd" value="false" />
  </component>
  <component name="ProjectColorInfo"><![CDATA[{
  "associatedIndex": 7
}]]></component>
  <component name="ProjectId" id="2gYBlydbsrNiXLNURMnI4SUYdwB" />
  <component name="ProjectViewState">
    <option name="hideEmptyMiddlePackages" value="true" />
    <option name="showLibraryContents" value="true" />
  </component>
  <component name="PropertiesComponent"><![CDATA[{
  "keyToString": {
    "Gradle.Upgrade Gradle wrapper.executor": "Run",
    "RunOnceActivity.OpenProjectViewOnStart": "true",
    "RunOnceActivity.ShowReadmeOnStart": "true",
    "RunOnceActivity.cidr.known.project.marker": "true",
```

```
  "RunOnceActivity.readMode.enableVisualFormatting": "true",
  "android.gradle.sync.needed": "true",
  "cf.first.check.clang-format": "false",
  "cidr.known.project.marker": "true",
  "kotlin-language-version-configured": "true",
  "last_opened_file_path": "E:/робота/Grocerylist"
 }
}]]></component>
 <component name="SpellCheckerSettings" RuntimeDictionaries="0" Folders="0" CustomDictionaries="0" DefaultDictionary="application-level" UseSingleDictionary="true" transferred="true" />
 <component name="TaskManager">
  <task active="true" id="Default" summary="Default task">
   <changelist id="fb7a28d7-19bc-4d97-85cb-8c6b3e74d32f" name="Changes" comment="" />
   <created>1715864786067</created>
   <option name="number" value="Default" />
   <option name="presentableId" value="Default" />
   <updated>1715864786067</updated>
  </task>
  <servers />
 </component>
</project>
```

ДОДАТОК Б

Інтерфейс

Рис 1 Інтерфейс додавання продукту

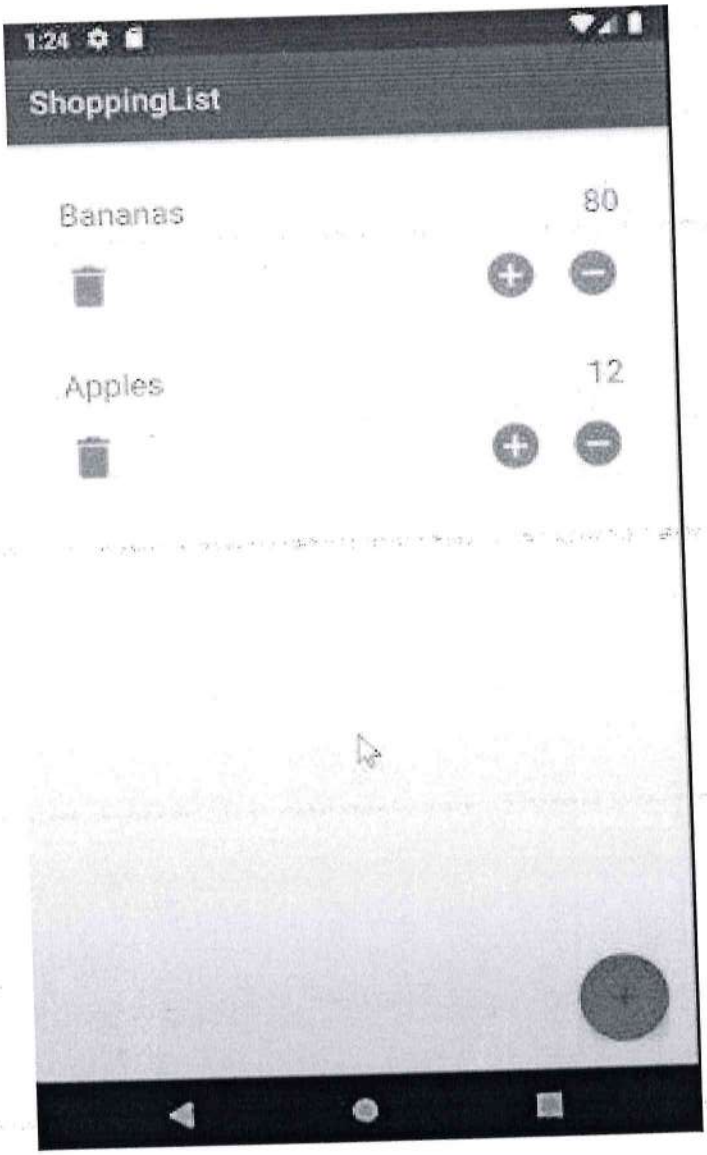

Рис 2 Інтерфейс списку

ДОДАТОК В

Логотипи компаній

Рис 3 лого Kotlin

Рис 4 android sudio