МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ЦЕНТР ЗАОЧНОЇ, ДИСТАНЦІЙНОЇ ТА ВЕЧІРНЬОЇ ФОРМ НАВЧАННЯ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК

Секція інформаційно-комунікаційних технологій

ВИПУСКНА РОБОТА

на тему:

«Мобільний додаток книга-рецептів для ОС Android»

Завідувач
випускаючої кафедри **Довбиш А. С.**

Керівник роботи **Шутилєва О.В.**

Студента групи ІНз-61С **Прокопенко А.Р.**

СУМИ 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

Кафедра комп'ютерних наук

Затверджую_____

Зав. кафедрою Довбиш А.С.

«_____» _____2020 р.

ЗАВДАННЯ

до випускної роботи

Студента четвертого курсу, групи ІНз-61С спеціальності «Комп'ютерні науки» заочної форми навчання Прокопенко Артем Романович

Тема: «Мобільний додаток книга-рецептів для ОС Android»

Затверджена наказом по СумДУ

№_____ від _____2020 р.

Зміст пояснювальної записки: 1) інформаційний огляд; 2) постановка задачі та опис інструменту вирішення поставлених задач; 3) практична реалізація.

Дата видачі завдання «_____»_____2020 р.

Керівник випускної роботи _____ Шутилєва О.В.

Завдання прийняв до виконання _____ Прокопенко А.Р.

РЕФЕРАТ

Записка: 51стор., 14 рис., 1 табл.,1 додаток, 10 джерел.

Мета роботи – дослідити основні аспекти створення додатку для ОС Android та розробити власний.

Об’єкт дослідження – функціонування додатків для ОС Android.

Предмет дослідження – додатки ОС Android.

Методи роботи – теоретично аналіз літератури за темою дослідження; практична розробка додатку.

Результатом проведеної роботи є розроблений Android додаток «Книга рецептів» за допомогою Android Studio, XML, Java, який завжди під рукою та дозволяє користувачам легко знаходити рецепти на будь-який смак та готувати смачні та незвичайні страви.

ДОДАТОК, JAVA, XML, ANDROID STUDIO.

ЗМІСТ

ВСТУП

Сьогоднішній день є часом науково-технічного прогресу, дуже складно уявити собі життя і побут сучасного суспільства без використання мобільних пристроїв. Прискорюється ритм життя, разом прискорюється і процес створення суспільством технічних новинок для своєї зручності. Взяти, наприклад, мобільні телефони. Ми користуємося ними всюди – вдома, в поїздці, на роботі, на відпочинку. Це зручно і можна навіть сказати комфортно.

Очевидно, що мобільний телефон вже давно перестав бути просто засобом спілкування. Звичайна розмова через телефон поступово стає другорядною функцією, зникаючи в величезному наборі функцій, що реалізуються мобільним телефоном він став багатофункціональним пристроєм, що дозволяє людині користуватися практично всіма сучасними технологіями.

Всі кожен день зустрічаються з проблемою, що б такого приготувати на сніданок, обід та вечерю і не завжди є доступ до комп’ютера, тому треба щоб книга рецептів була завжди під рукою смартфоні.

Таким чином для розв'язання проблеми було сформовано мету випускної роботи: розробка додатку книги рецептів на ОС Android.

Для досягнення мети необхідно вирішити наступний перелік задач:

- Провести літературний огляд;
- Обрати засоби та методи реалізації;
- Розробити додаток;
- Додати набір рецептів;
- Провести тестування продукту.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Актуальність теми

Людині завжди характерно бажання досягти найбільшого комфортного існування, а в нас час його атрибутом вважається мобільний телефон з виходом в інтернет. У зв’язку з цим, стала найбільш актуальною розробка різних додатків для мобільних пристроїв з розширенням можливостей мобільного інтернету.

Сьогодні мобільні додатки розв’язують різноманітні задачі в сфері інформаційних технологій, в тому числі створення 3Д анімацій. Одні з них забезпечують якісний зв'язок з мережею, інші вибирають найоптимальніший маршрут, треті надають допомогу в пошуку вигідних магазинів. Доставку продуктів додому забезпечують інші додатки. Але в основі всіх лежить спеціальна утиліта, яка прискорює вирішення поставленого завдання, покращуючи якість і підвищуючи рівень комфортного життя.

Додатки для мобільних пристроїв задовольняють робочі і розважальні потреби. Одні успішно допомагають бізнесменам і офісним працівникам контролювати бізнес і вести по ньому звіти, розробляти дизайн в оригінальному і фірмовому стилі. Інші забезпечують якісне прослуховування музики і перегляд фільмів, підтримують засоби спілкування і виконують ряд інших функцій.

Попит на подібні додатки для мобільних пристроїв стабільно зростає вже кілька останніх років. Напрошується висновок, що на сьогоднішній день актуальність розробки таких програм цілком доцільна і обов'язково отримає відповідне визнання користувачів.

1.2 Типи мобільних додатків

1.2.1. Нативні додатки

Додатки називають нативними тому, що вони написані «рідною» для певної платформи мовою програмування. Для Android цією мовою є Java, тоді як для iOS – objective-C або Swift.

Нативні додатки знаходяться на самому пристрої, доступ до яких можна отримати, натиснувши на іконку. Вони встановлюються через магазин додатків (Play Market на Android, App Store на iOS і ін.). Розроблені спеціально для конкретної платформи і можуть використовувати всі можливості пристрою – камеру, GPS-датчик, акселерометр, компас, список контактів і все інше. Також вони можуть розпізнавати стандартні жести, встановлені операційною системою або зовсім нові жести, які використовуються в конкретному додатку.

У силу того, що нативні додатки оптимізовані під конкретну ОС, вони органічно вписуються в будь-який смартфон, відрізняючись високою швидкістю роботи та продуктивністю. Нативні додатки можуть отримати доступ до системи оповіщення пристрою, а також, в залежності від призначення може повністю або частково обходитися без наявності інтернет-з'єднання.

Переваги нативних додатків:

- Швидкість роботи та продуктивність;
- Високий ступінь безпеки;
- Розширений інтерфейс;
- Відносно висока вартість розробки;
- Максимально можлива функціональність;
- Здатність працювати без Інтернету;
- Зручність для кінцевого користувача.

Недоліки нативних додатків:

- Охоплення платформ;
- Тривалі терміни розробки;
- Необхідність випускати оновлення в косметичних цілях.

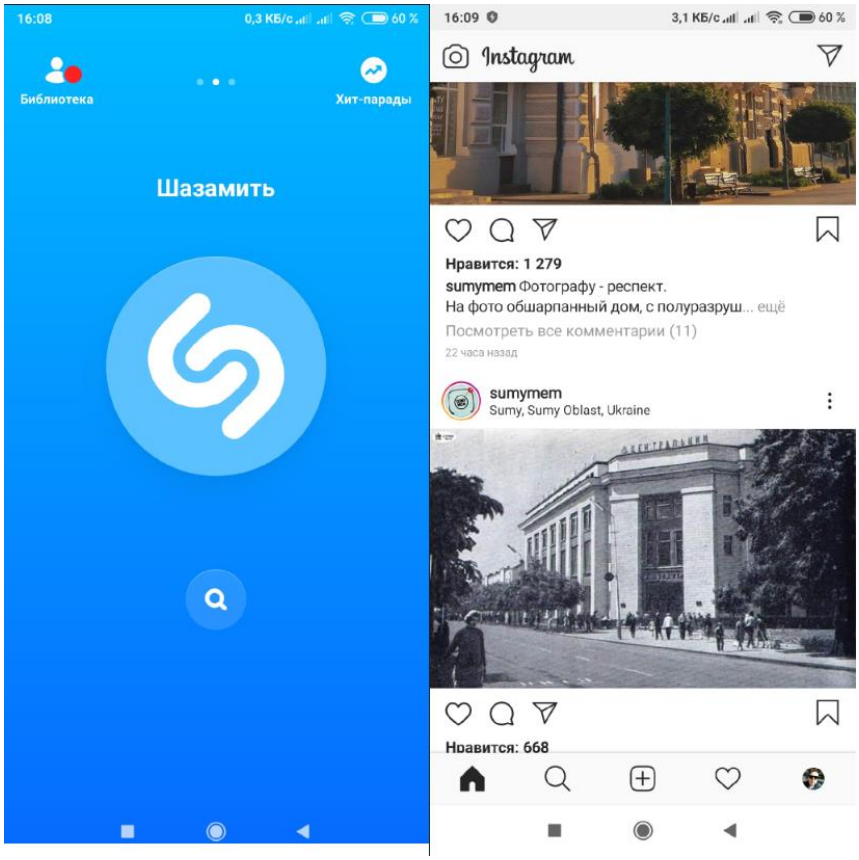

Рисунок 1.1 – додаток Shazam та Instagram

1.2.2. Мобільні веб-додатки

Насправді мобільні веб-додатки не є додатками як такими. Адже річ у тому, що веб-додаток, по суті, являє собою сайт, який адаптований і оптимізований під будь-який смартфон. І для того, щоб скористатися ним, достатньо мати на пристрої браузер, знати його адресу і розташовувати інтернет-з'єднанням (завдяки йому відбувається оновлення інформації в даному виді додатків).

Запускаючи мобільні веб-додатки, користувач виконує всі ті дії, які він виконує при переході на будь-який веб-сайт, а також отримує можливість «встановити» їх на свій робочий стіл, створивши закладку сторінки веб-сайту.

Веб-додатки відрізняються кросплатформеністю, тобто здатні функціонувати, незалежно від платформи девайса. Перевагою виступає і те, що вони не використовують його програмне забезпечення. А через те, що є

мобільною версією сайту з розширеним інтерактивом, веб-додатки не займають зайве місце в пам'яті смартфона.

Веб-додатки стали широко популярні в той час, коли почав розвиватися HTML5 і люди усвідомили, що можуть отримати доступ до безлічі функцій нативних додатків, просто зайшовши на веб-сайт через звичайний браузер. На сьогодні складно сказати, де саме розташовується чітка межа між веб-додатками і звичайними веб-сторінками, оскільки функціонал HTML5 зростає з кожним днем і все більше і більше сайтів його використовують.

Розробляються веб-додатки за допомогою інструментів і фреймворків, які стали традиційними. Внаслідок чого процес їх розробки останнім часом істотно прискорився. Фахівців з їхньої розробки, до всього іншого, так і зовсім достатньо.

Водночас камінь в город веб-додатків слід кинути за нездатність працювати з ними без Інтернету. Причому з цього випливає й інший мінус - їх продуктивність, яка знаходиться на середньому рівні, в порівнянні з іншими видами додатків. Більш того, вона залежить від можливостей інтернет-з'єднання провайдера послуг.

Переваги мобільних веб-додатків:

- Повне охоплення платформ;
- Простий і швидкий процес розробки;
- Кількість компетентних розробників;
- Відсутність необхідності завантаження з магазину додатків.

Недоліки мобільних веб-додатків:

- Обов'язкове підключення до Інтернету;
- Убогий інтерфейс програми;
- Неможливість відправити push-повідомлення;
- Продуктивність і швидкість роботи;
- Незадовільний рівень безпеки.

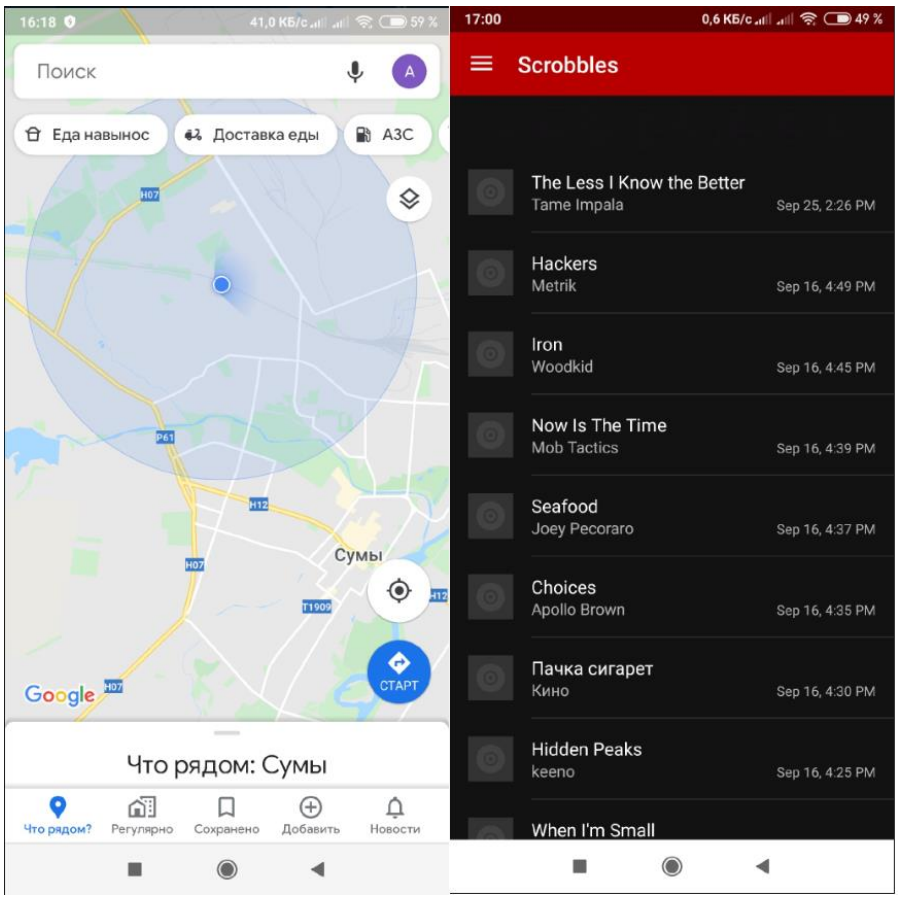

Рисунок 1.2 – додаток google maps та last.fm

Приклади мобільних веб-додатків:

- last.fm вважається веб-додатком, хоча, по суті, це в той же час і веб-сайт.
- maps.google.com - веб-сайт, але в той же час це і веб-додаток.

1.2.3. Гібридні додатки

Гібридні додатки являють собою поєднання веб і нативних додатків. Особливо, мається на увазі їх кросплатформенність і доступ до функціонала смартфона. Такі додатки можуть бути завантажені виключно з маркетів, на приклад, Google Play і App Store. Разом з тим вони мають у своєму розпорядженні опцією автономного оновлення інформації, а для їх роботи необхідно інтернет-підключення. Без наявності останнього веб-функції просто не працюють.

Серед багатьох компаній вибір найчастіше падає на розробку саме гібридного додатка. Це можна пояснити тим, що гібридні програми здатні

поєднувати гідності нативних з технологічною актуальністю, яка забезпечується останніми веб-технологіями. Однак, на відміну від нативних, вартість створення гібридних на порядок нижче, а його швидкість - вище. Спорідненість гібридних додатків з веб-додатками, своєю чергою, дає плоди у вигляді того, що в них можна легко і оперативно вносити корективи. Тобто розробникам не доводиться, як у випадку з нативними, повторно розміщувати додаток в магазині заради усунення помилок попередньої версії.

Розробка гібридного додатка бачиться перспективною ще й тому, що вона має на увазі створення його відразу під дві платформи. Як наслідок, це позбавляє головного болю, пов'язаного з окремою розробкою програми під кожну ОС. Крім усього іншого, потрібно взяти на замітку, що якість і можливості гібридних додатків залежать, перш за все, від фреймворка, який використовує розробник. Також варто приділити належну увагу факторам, які роблять гібридні програми варіантом на тлі інших.

Отже, варто розробляти його, якщо:

- Є необхідність заощадити в бюджетному плані;
- Потрібно створити відносно нескладне додаток з простою анімацією;
- Є завдання оперативної розробки програми як мінімум на 2 платформи.

Переваги гібридних додатків:

- Вартість і швидкість розробки;
- Кількість розробників;
- Кросплатформеність;
- Опція автономного оновлення.

Недоліки гібридних додатків:

- Некоректна робота при відсутності інтернет-з'єднання;
- Середня швидкість роботи на тлі нативних;
- Мінімалізм щодо візуальних елементів.

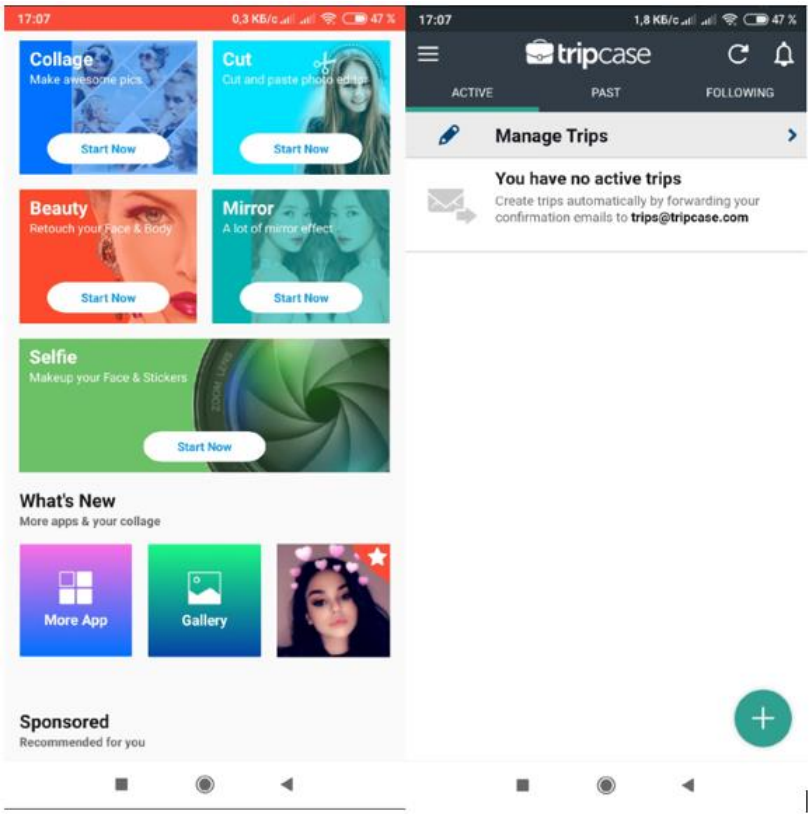

Рисунок 1.3 – додаток HeartCamera і TripCase

1.3 Аналіз існуючих аналогів

Мобільний додаток – це спеціальний пакет, який користувач може отримати, встановивши зі спеціального ринку додатків (Google Play, App Store).

Мобільні додатки – програмне забезпечення для мобільних пристроїв. Їх можливості частково залежать від можливостей мобільних пристроїв, частково від можливостей, закладених розробником додатків. Використовуючи мобільні додатки, користувачі отримують послуги і товари, дізнаються новини, працюють і відпочивають. Власники додатків продають інформацію, послуги, товари, заробляють на рекламі, підвищують мобільність робочих місць, управляють бізнес-процесами.

Мобільні додатки пишуться на мовах програмування високого рівня, а потім компілюються в машинний код операційної системи для отримання максимальної продуктивності. Сучасні мобільні пристрої оснащені додатковими модулями, такими як фотокамери, акселерометри, гіроскопи, модулі

бездротового зв'язку дають унікальні можливості для розширення функціоналу програми.

Перед тим як приступати до розробки додатку, потрібно проаналізувати аналоги. Сьогодні існує велика кількість додатків книги кулінарних рецептів. Перш за все, проблема полягає в тому, що частина з них коштують грошей, а інша частина має велику кількість реклами. Недолік додатків книг рецептів – це недостатня оптимізація коду, що впливає на швидкодію, а своєю чергу швидкодія погіршує зручність користування. Також вони мають багато зайвих елементів. Деякі з них не здатні працювати без інтернету. Практично не реально знайти додатки які в змозі похизуватися наявностю української мови, що є великим мінусом для україномовного користувача. Розглянемо існуючі безкоштовні додатки:

Першим аналогом є «Книга рецептов: Рецепты на каждый день – це додаток виробника, який спеціалізується на розробці різнопланових додатків.

Даний додаток містить детально описані етапи приготування та наявністю фото до кожного з них.

Рисунок 1.4 – Додаток «Книга рецептов: Рецепты на каждый день»

Виділимо головні переваги цього додатку:

- Простий і зручний дизайн;

- Помітка переглянутих рецептів;
- Велика кількість рецептів;
- Багато фотографій з поетапним приготуванням ;
- Здатність роботи без інтернету.

Звичайно додаток має і недоліки:

- Не досконалий дизайн;
- Велика кількість реклами, яка відкривається майже при кожному кліку;
- Неможливість відключення цієї реклами;
- Невдале меню;
- Відсутність української мови.

Наступний аналог розглянутих додатків– «Простые рецепты – це додаток виробника, який спеціалізується на розробці тільки книг рецептів. Даний додаток пропонує великий вибір рецептів на різний смак.

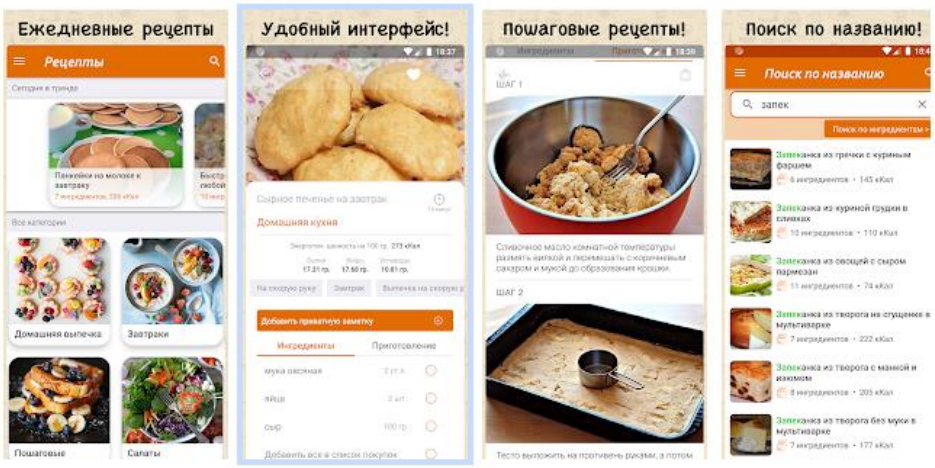

Рисунок 1.5 – додаток «Простые рецепты»

Переваги / особливості цього додатку :

- Цікавий дизайн;
- Пошук по ключовому слову, інгредієнтам;
- Розписана енергетична цінність продукту;
- Можливість скласти список покупок;
- Здатність роботи без інтернету.

Як й попередній, цей аналог має недоліки:

- Велика кількість реклами;
- Відсутність української мови;
- Відсутність фотографій етапів приготування.

Один із найпопулярніших аналогів «Поваренная книга Рецепты» – це додаток від Riafy Technologies, який спеціалізується на різнопланових книгах рецептів. При першому запуску додатку на пропонується обрати мову, рівень кулінарних здібностей, цілі.

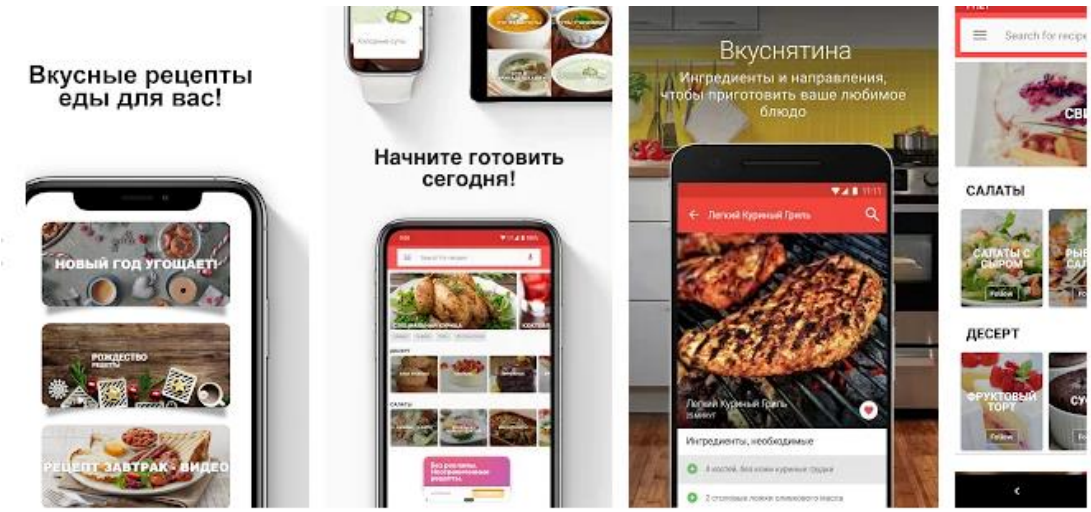

Рисунок 1.6 – Додаток «Поваренная книга Рецепты»

Переваги/особливості цього додатку :

- Цікавий дизайн;
- Пошук по ключовому слову, інгредієнтам;
- Розписана енергетична цінність продукту;
- Можливість скласти список покупок;
- Великий функціонал пошуку.

Як й попередній цей аналог має недоліки:

- Велика кількість реклами;
- Відсутність української мови;

- Відсутність фотографій етапів приготування;
- Не дуже вдале меню.

Згідно з аналізом аналогів, була сформована порівняльна таблиця розглянутих прикладів та майбутнього додатку. Результати аналізу представлено у таблиці 1.1.

Таблиця 1.1 – Порівняльна характеристика аналогів

Назва критерію	Назва додатку		
	Рецепты на каждый день	Простые рецепты	Поваренная книга Рецепты
1. Використання ресурсу без інтернету	+	+	-
2. Зручність навігації	-	+	+
3. Наявність реклами	+	+	+
4. Наявність списку покупок	+	+	+

1.4 Мета та задачі додатку книги рецептів

Метою випускної роботи є розробка android додатку книги рецептів з використанням Java та XML. Продукт дозволить безкоштовно та українською мовою знаходити для приготування рецепти страв на різний смак. Окрім цього, він кардинально відрізнятиметься від інших додатків завдяки простому дизайну.

При подальшій підтримці та модернізації цього додатку існує можливість розробити пошук по назвах та ключовим словам, можливість розрахунку калорій страв, реалізація списку покупок, додавання меню, улюблених рецептів, створення зворотного зв'язку, відгуків, багатомовності.

Розроблений додаток повинен підтримувати наступні функції використання:

- можливість перегляду рецептів;

- можливість увійти в додаток без реєстрації та інтернету;
- можливість запустити додаток на будь-якому телефоні ОС Android.

Функціональні вимоги :

- адаптивність;
- наявність великої кількості рецептів;
- підтримка смартфонів с різною діагоналлю екрана та версією ОС.

Нефункціональні вимоги:

- простота дизайну;
- зрозуміла навігація.

Користувач додатку має отримати максимально зручне у повсякденному використанні мобільне розширення, яке було б водночас максимально інформативним та швидким у користуванні. Для якісної та безвідмовної роботи треба залучити найостанніші розробки у сфері інформаційних технологій. Мобільний додаток має надавати інформаційну базу, що буде зображено у вигляді книги рецептів.

Для реалізації поставленої мети в дипломному проекті необхідно реалізувати наступні задачі:

- провести аналіз предметної області;
- провести аналіз технологій розробки додатків;
- розробити інтерфейс додатку;
- провести проектування бази даних;
- провести тестування коректної роботи додатку.

2. МЕТОДИКА ВИРІШЕННЯ ПОСТАВЛЕНИХ ЗАДАЧ

2.1 Мобільні операційні системи

Операційна система Android з кожним роком набирає все більшу і більшу популярність. Популярна операційна система для телефонів і інших гаджетів створена на базі Linux, яка була і залишилася конкурентом Windows від компанії Microsoft. ОС Android – це відкрита продукція, для якого є в наявності велика кількість різноманітних програм в безмежному безкоштовному доступі. До того ж ОС безперервно розвивається і поліпшується. Зараз ОС Android можна зустріти не стільки на телефонах, але і на планшетних пристроях, а також і в фоторамках.

Основна частина додатків для даної системи написана на мові Java. Завдяки опису операційної системи Android можна з'ясувати, як вона працює, які застосовуються рушії і бібліотеки.

Основними перевагами даної ОС є:

- Кастомізація, тобто будь-який телефон на платформі Android легко налаштувати під себе, від таких простих речей, як зовнішній вигляд і вибір програм за замовчуванням, закінчуючи глибокими налаштуваннями самої системи. І в цьому з Android навряд чи хто здатний змагатися;
- Величезний вибір додатків і можливості їх установки з будь-яких джерел без особливих маніпуляцій – це значна перевага;
- Відкритість системи, тобто файлова система телефонів на Android відкрита для змін. Існує можливість використовувати пам'ять телефону як флешку: завантажувати туди файли, змінювати прошивку на будь-яку з представлених на ринку. Безліч призначених для користувача модифікацій можуть дати саме той набір функцій, який підходить саме для користувача, а отримання Root-прав, знімає взагалі будь-які обмеження у роботі з системою;
- Доступність телефонів в будь-якому ціновому сегменті, всіх видів, розмірів, кольорів і параметрів.

Окрім переваг існують і недоліки, які необхідно зазначити:

- Оновлення – безліч телефонів залишається без підтримки дуже скоро через те, що під кожну модель потрібно розробити свою версію прошивки. А через те, що модельний ряд настільки широкий, часто витрачати на це час розробники не хочуть. Адже куди простіше випускати нові моделі відразу з останньою версією прошивки, стимулюючи продажі;
- Швидкодія і плавність інтерфейсу зменшується з часом.

2.2 Компоненти додатків Android

Компоненти програми є цеглинками, з яких складається додаток для Android. Кожен компонент являє собою окрему точку, через яку система може увійти в додаток. Не всі компоненти є точками входу для користувача, а деякі з них залежать один від одного. При цьому кожен компонент є самостійною структурною одиницею і відіграє певну роль - кожен з них являє собою унікальний елемент структури, який визначає роботу програми в цілому.

Компоненти програми можна віднести до одного з чотирьох типів. Компоненти кожного типу призначені для певної мети, вони мають власний життєвий цикл, який визначає спосіб створення і припинення існування компонента. Поговоримо про кожне окремо.

2.2.1. Activities

Activity – це зовнішній інтерфейс для операції, яку може зробити користувач.Тобто, це один поточний екран як окрема одиниця активності, свого роду робоче вікно з одним призначеним для користувача .

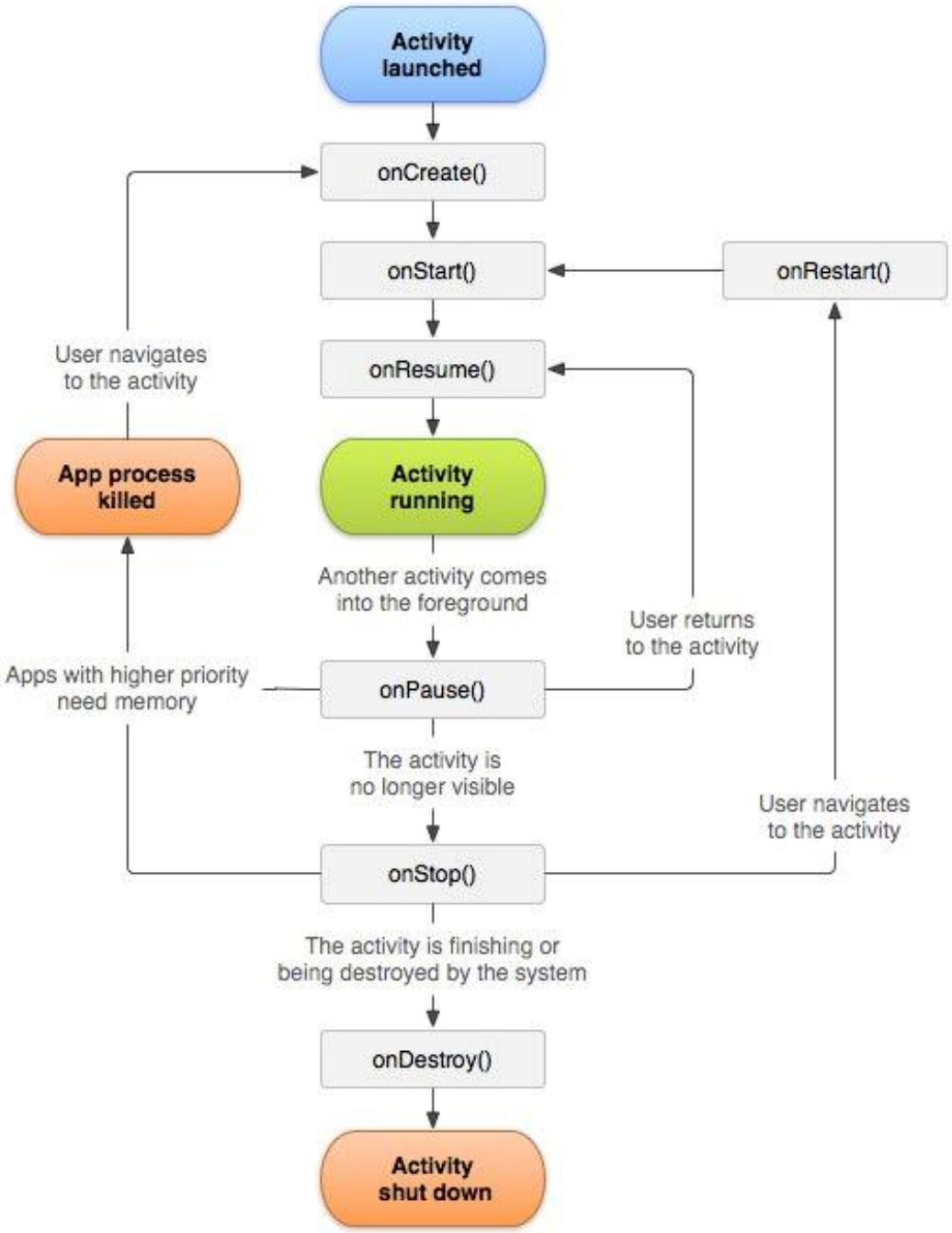

Рисунок 2.1 – Додаток «Поваренная книга Рецепты»

Всі activity даного додатка формують єдиний призначений для користувача інтерфейс, але незалежні між собою. Додаток може складатися з усього одного activity або відразу з декількох. Якими саме будуть activity і скільки їх буде, залежить від конкретного додатка і його дизайну. Зазвичай, одне з activity позначається як перше, це означає, що воно буде надано користувачеві під час

запуску програми.. Таким чином, перехід від основного activity до іншого здійснюється в той час, коли поточний activity викликає наступне.

Кожне activity надає вікно за умовчанням. Як правило вікно створюється в повноекранному вигляді, але воно також може і не займати весь екран і перебувати поверх інших вікон. Activity також може залучити додаткові вікна.

Загальний вигляд вікна будується за допомогою підпорядкування візуальних компонентів – об'єктів, похідних від базового класу View. Кожен компонент представляє собою прямокутник всередині вікна. Батьківські компоненти містять дочірні і організують їх розташування. Таким чином здійснюється інтерактивна взаємодія з користувачем.. В операційній системі Android вже є набір готових візуальних компонент, які доступні для використання розробниками. Набір містить кнопки, текстові поля, смуги прокрутки, меню, прапорці-перемикачі і багато іншого.

2.2.2. Sevices

Ці компоненти додатку, які працюють у фоновому режимі,вони потрібні для забезпечення роботи віддалених процесів, але в загальному випадку це просто режим, який функціонує, коли додаток згорнуто. Прикладом такого процесу може стати введення-виведення файлу, коли користувач робить щось інше або взаємодіяти з постачальником контенту. Сервіс сам по собі не взаємодіє с користувачем і не має свого інтерфейсу. Servises можуть запускаються, управлятися і бути пов'язаними з іншими компонентами, наприклад, activity. Також може запускатися разом з системою.

2.2.3. Content providers

Даний компонент надає взаємодію з наборами даних.Ці дані можуть зберігатися в базах даних, файловій системі, в інтернеті або в будь-якому іншому постійному місці.. За допомогою content provider інший додаток може запитувати дані і, якщо виставлені відповідні дозволи, змінювати їх.

Восновному, content provider можна використовувати для считування і запису даних, які використовуються додатком і не є відкритими для інших.

Дані компоненти реалізуються як підклас ContentProvider. І для того, щоб інші програми могли взаємодіяти з даними, їм необхідно надати стандартний набір API.

2.2.4. Broadcast receivers

Цей сервіс керує поширенням системних повідомлень, відстеження та реагування на дії. Додатки можуть повідомлення, наприклад, малий заряд батареї, єкран вимкнений або сигналізувати про те, що інформація завантажена на пристрій і доступна до використання. Broadcast receiver не надає призначеного для користувача інтерфейсу, проте, він здатний створювати повідомлення. Однак частіше broadcast receiver взаємодіє з іншими компонентами для того, щоб самому виконувати мінімальний обсяг роботи.

2.3 Мова програмування Java

Одна з найпопулярніших мов для розробки додатків по під операційну систему Android є Java. Застосування Java в створенні мобільних додатків є невипадковим вибором, бо це відкрита і потужна мова. Java застосовується в розробці додаткового функціонала веб – серверів, повномасштабних корпоративних додатків, проектують додатки для різних пристроїв крім смартфонів і планшетів, і це тільки мала частина можливостей і області застосування цієї мови. Java – багато платформна мова, з її допомогою програмісти можуть створювати додатки, не звертаючи увагу на апаратні особливості пристрою, а з використання Android API та інших допоміжних засобів розробники здатні швидко вивчити проектування додатків для операційної системи Android.

Головною перевагою операційної системи Android є відкритість. Дана платформа поширюється на вільній основі і заснована на використанні

відкритого коду. Завдяки цьому, розробники можуть отримати доступ до вихідного коду Android і на його основі реалізовувати нові функції програми.

За допомогою мови Java, яка відноситься до об'єктно орієнтованим мов програмування, розробники отримують доступ до великої кількості потужних бібліотек, що прискорюють написання програми. Створення призначеного для користувача графічного інтерфейсу є контрольованим подією. З його допомогою можна зібрати інтерфейс з заготовлених елементів, у вигляді текстових полів, кнопок і перемикачів змінюючи їх розмір, перетягуючи в будь-які частини екрану і додаючи підписи.

Інші переваги Java:

- Безпека: для перевірки достовірності даних використовується методи, засновані на шифруванні з відкритим ключем.
- Динаміка: Java легко адаптується під мінливі умови та є більш динамічною мовою на відміну від C і C ++. Програми можуть виконувати велику кількість під час обробки інформації, яка може бути використана для перевірки й дозволу доступу до об'єктів на час виконання.
- Портативність: архітектурно-нейтральний і не має залежності від реалізації аспектів специфікацій - все це робить Java портативним. Компілятор в Java написаний на ANSI C з чистою переносимістю, який є підмножиною POSIX.
- Інтерпретування: Java байт-код переводиться на льоту в машинні інструкції та ніде не зберігається, роблячи процес більш швидким і аналітичним, оскільки зв'язування відбувається як додаткове з невеликою вагою процесу. Висока продуктивність: введення Just-In-Time компілятора, дозволило отримати високу продуктивність.

2.4 Середовище розробки Android Studio

Середовище розробки Android Studio визнано найпопулярнішою і зручною платформою для розробки і тестування додатків на Android. Google розробила це програмне забезпечення з величезним набором інструментів для спрощення

процесу розробки мобільних додатків. У порівнянні з попереднім способом створення додатків в середовищі розробки Eclipse, процес став простіше і динамічніше. Це сталося завдяки появі можливості зображати головні робочі елементи в самій структурі створюваного додатка, що дозволило ефективніше підійти до розробки.

Розробники виділяють можливість перегляду в реальному часі всіх змін проекту. Так само середовище дозволяє розробляти програми для різних версій Android.

Завдяки новим вбудованим інструментам та доопрацювання інтерфейсу користувача процес взаємодії з середовищем став в рази зручніше Eclipse. В наслідок чого написання коду стало більш раціонально, що дозволило легко орієнтуватися при розробці великих за обсягом проектів. є функція перетягування функціональних елементів в самій програмі, що спрощує редагування інформації.

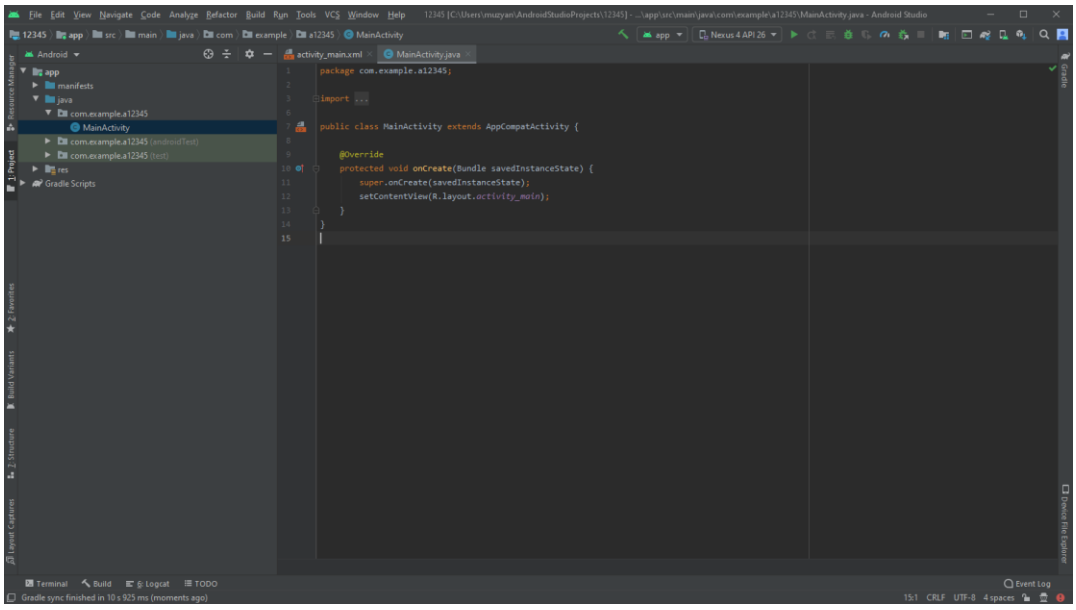

Рисунок 2.2 – Інтерфейс середовища Android Studio

Переваги Android Studio:

- Підтримка роботи з декількома мовами;
- Підсвічування синтаксису;

- Можливість розробки додатків для планшетів, телевізорів, годинників та інших розумних пристроїв;
- Можливість швидко локалізувати додатки;
- Тестування створеного додатка в емуляторі;
- Велика бібліотека з готовими компонентами і шаблонами;
- Ре факторинг вже готового коду;
- Попередня перевірка додатки на наявність помилок;
- Наявність літератури багатьма мовами;
- Відправка push-повідомлень для додатків через будь-які хмарні сервіси відразу на пристрої під Android;
- зрозумілий інтерфейс;
- зручна структура проекту;
- підтримка відстеження ефективної роботи рекламних оголошень;
- наявність інструментів для підвищення якості проектів і монетизації.

Інструменти Android Studio:

- AVD Manager – емулятор для запуску додатків;
- Android Device Monitor – проводить моніторинг віртуального або фізичного пристрою під час роботи;
- Android Debug Bridge – інструмент командного рядка, за допомогою якого можна копіювати файли з пристрою і назад, видаляти й встановлювати додатки;
- FlowUp – допомагає відстежувати загальну продуктивність додатки;
- Плагін Gradle для збірки програм;
- JRebel – значно скорочує час складання додатка;
- Android Asset Studio – генерація різних типів іконок;
- Android Wi-Fi ADB – дозволяє запускати додаток на пристрої використовуючи Wi-Fi;
- FindBugs – пошук багів прямо в процесі розробки;
- Drawble Optimizer – оптимізація зображень.

Недоліки Android Studio:

- Досить повільна збірка проекту без додаткових інструментів;
- Високі системні вимоги до ПК;
- Повільна робота емуляторів.

3. ПРАКТИЧНА РЕАЛІЗАЦІЯ КЛІЄНТСЬКОЇ ЧАСТИНИ ДОДАТКУ КНИГИ РЕЦЕПТІВ

Вивчивши всі переваги і недоліки різних середовищ розробки, було обрано Android studio, бо вона є офіційним середовищем розробки мобільних додатків для Android і має безліч вбудованих функцій і бібліотек. У якості мови програмування будемо використовувати Java, для розмітки – XML.

Для початку необхідно створити вид головної сторінки. При створенні проекту в якості шаблону використаємо Empty Activity.

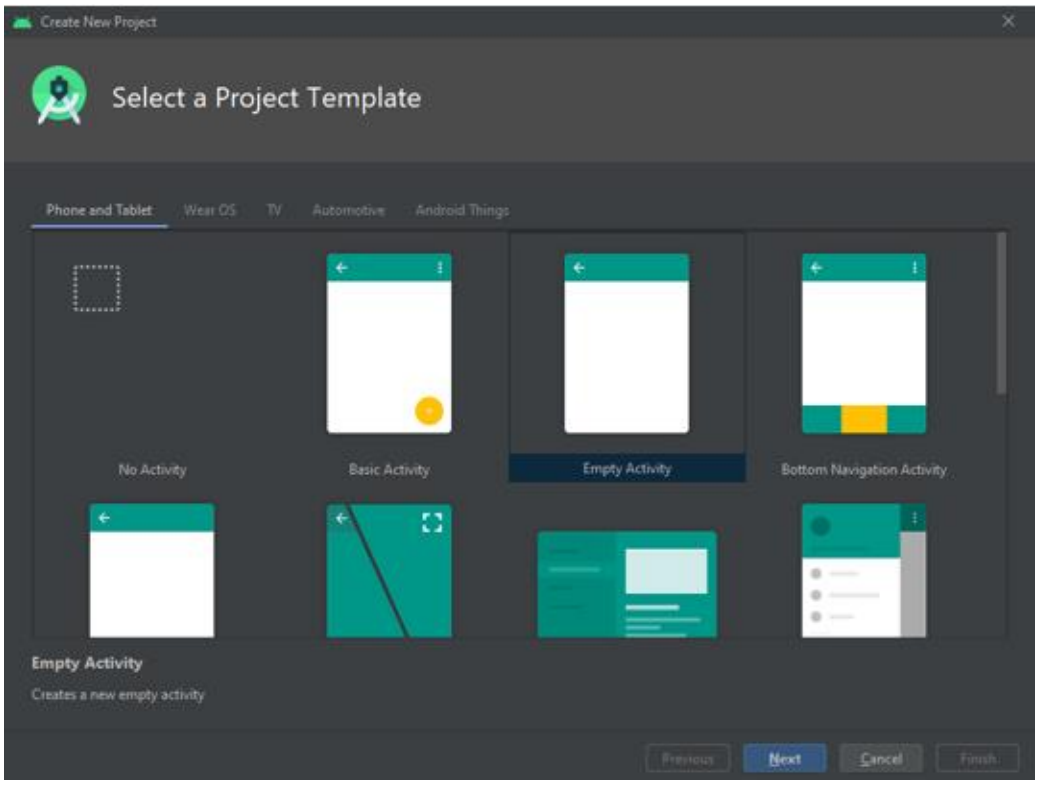

Рисунок 3.1 – Вибір шаблону

Після створення проекту в першу чергу необхідно відредагувати файл Activity_main.xml, цей елемент буде відповідати за те в якому виді буде представлена головна сторінка для користувача.

Для головної сторінки використаємо контейнер LinearLayout, який впорядковує всі дочірні елементи в одному напрямку. Напрямок розмітки

вказується за допомогою атрибута android: orientation. Далі використаємо віджет RecyclerView який дозволить показувати список елементів. Також необхідно вказати атрибут android:scrollbars”verical”, для можливості гортати список по вертикалі, за допомогою android:id="@+id/recyclerView" надалі цей список заповниться автоматично.

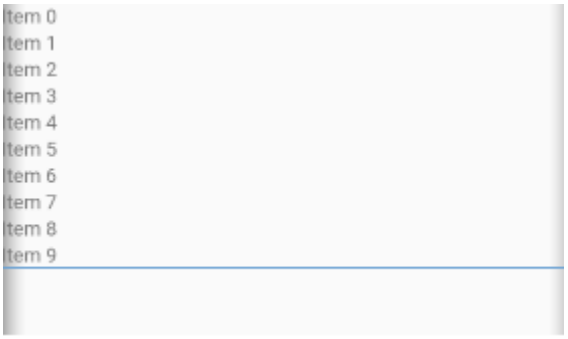

Рисунок 3.2 – Створення списку елементів

Далі створимо файл FoodData.java. З допомогою модифікатора private задаємо змінні. Клас String вказує на те що змінні будуть в текстовій формі, int – цілим.

```
public class FoodData {

    private String itemName;
    private String itemDescription;
    private String itemIngredients;
    private String itemRecipe;
    private int itemImage;
```

Створимо конструктор, android studio дозволяє автоматично генерувати його.

```
    public FoodData(String itemName, String itemDescription, String itemIngredients, String itemRecipe, int itemImage) {
        this.itemName = itemName;
        this.itemDescription = itemDescription;
        this.itemIngredients = itemIngredients;
        this.itemRecipe = itemRecipe;
        this.itemImage = itemImage;
    }
```

Для того, щоб отримати дані, доступ к яким на пряму обмежений, потрібно задати метод читання. Це також можна зробити автоматично:

```
        public String getItemName() {
        return itemName;
        }

        public String getItemDescription() {
        return itemDescription;
        }

        public String getItemIngredients() {
        return itemIngredients;
        }

        public String getItemRecipe() {
        return itemRecipe;
        }

        public int getItemImage() {
        return itemImage;
        }
    }
```

Наступним кроком є створення файлу recycler_row_item.xml який буде відповідати за дизайн компонентів списку головної сторінки. Тут також використовуємо контейнер Linearlayout та компонент CardView.

Рисунок 3.3 – Компоненти списку

Після цього потрібно створити новий java class під назвою MyAdapter. Щоб створити адаптер для RecyclerView, наслідуючи від RecyclerView.Adapter. Цей адаптер являє шаблон проектування viewholder, що має на увазі використання призначеного для користувача класу, який розширює RecyclerView.ViewHolder.

```
class FoodViewHolder extends RecyclerView.ViewHolder{

    ImageView imageView;
    TextView mTitle,mDescription;
    CardView mCardView;

    public FoodViewHolder( View itemView) {
    super(itemView);

    imageView = itemView.findViewById(R.id.ivImage);
    mTitle = itemView.findViewById(R.id.tvTitle);
    mDescription = itemView.findViewById(R.id.tvDescription);
```

```
mCardView = itemView.findViewById(R.id.myCardView);
```

Далі, слід перевизначити метод onCreateViewHolder. Як випливає з назви, цей метод викликається, коли кастомний ViewHolder слід активувати. Ми вказуємо макет для кожного елемента RecyclerView. Потім LayoutInflater заповнює макет, і передає його в конструктор ViewHolder

```
public FoodViewHolder onCreateViewHolder(@NonNull ViewGroup viewGroup, int i) {
View mView = LayoutInflater.from(viewGroup.getContext()).inflate(R.layout.recycler_row_item,viewGroup,false);

return new FoodViewHolder(mView);
```

Перевизначаємо onBindViewHolder та визначаємо вміст кожного елемента RecyclerView. Тут можна вносити значення полів, ім’я, хещтег, фото CardView:

```
@Override
  public void onBindViewHolder(@NonNull final FoodViewHolder foodViewHolder, int i) {

foodViewHolder.imageView.setImageResource(myFoodList.get(i).getItemImage());
foodViewHolder.mTitle.setText(myFoodList.get(i).getItemName());
foodViewHolder.mDescription.setText(myFoodList.get(i).getItemDescription());
```

Після запуску додатку, з’являється головний екран з привітанням (рис.3.3). Привітання є активним текстовим полем, після натискання на нього виконується перехід на наступний екран, на якому можна обирати необхідні категорії (рис.3.4).

Рисунок 3.3 – Симуляція «Головного екрану» після запуску додатку

Рисунок 3.4 – Доступні категорії додатку

Для кожної з категорії існують свої підкатегорії. Якщо натиснути на другу категорію «Країна кухні» Відкриваються найбільш популярні страви із зазначенням країни походження:

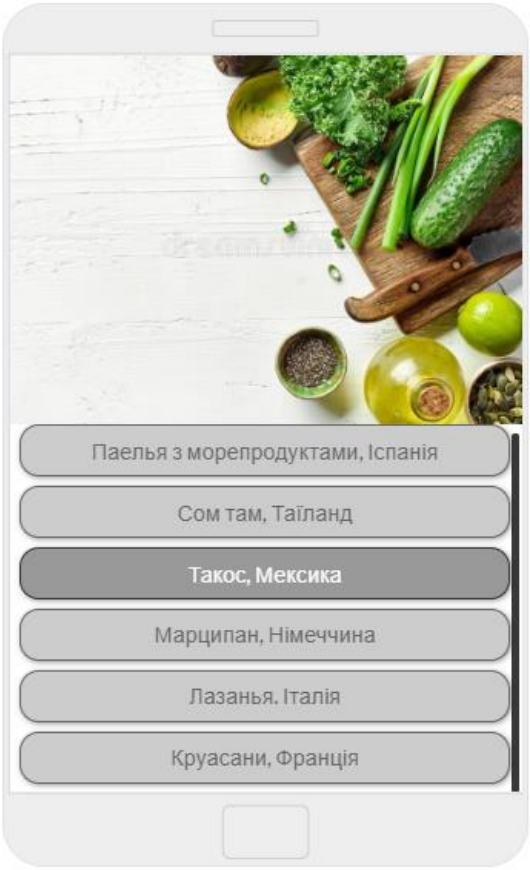

Рисунок 3.5 – Вміст категорії «Країна кухні»

Після натискання на страву виконується перехід на страву з покроковими рекомендаціями по приготуванню. Окрім текстових рекомендацій наявні зображення проміжних приготувань (рис.3.6).

Код додатку знаходиться у Додатку А.

Рисунок 3.6 – Покроковий рецепт Лазаньї

Тестування додатку не показало суттєвих помилок у роботі. У перспективі розвитку будуть додаватись рецепти страв та можливість створювати власні, пропонувати страву на основі існуючих у холодильнику продуктів.

ВИСНОВКИ

Під час виконання випускної роботина тему «Мобільний додаток книга рецептів для ОС Android» був розроблений додаток для ОС Android. Додаток був розроблений згідно вимог, що були зазначені у завданні:

- Можливість перегляду рецептів;
- Можливість увійти в додаток без реєстрації та інтернету;
- Багато місця подальшого наповнення функціоналом.

Під час виконання випускної роботи було проведено детальний аналіз предметної області: визначено питання актуальності проблеми, проведено аналіз аналогів додатків, сформовано мету та задачі для реалізації продукту, проведено аналіз наявних технологій розробки.

Після того, як був реалізований додаток, було проведено його тестування. Результати даного етапу:

- Адаптивність працює коректно;
- Не має раптових вилетів та випадкових помилок.

СПИСОК ЛІТЕРАТУРИ

1. Glas G. Web app vs. Native app [Електронний ресурс] / Grant Glas // https://www.app-press.com/ – Режим доступу до ресурсу: https://www.app-press.com/blog/web-app-vs-native-app.

2. Web application [Електронний ресурс] – Режим доступу до ресурсу: https://www.pcmag.com/encyclopedia/term/web-application.

3. Native App vs. Mobile Web App: A Quick Comparison [Електронний ресурс] – Режим доступу до ресурсу: https://www.webfx.com/blog/web-design/native-app-vs-mobile-web-app-comparison/.

4. John Wiley & Sons. Reto Meier Professional Android 4 Application Development. Wrox, 2012.

5. Основы создания приложений [Електронний ресурс] – Режим доступу до ресурсу: https://developer.android.com/guide/components/fundamentals?hl=ru.

6. Introduction to Activities [Електронний ресурс] – Режим доступу до ресурсу: https://developer.android.com/guide/components/activities/intro-activities?hl=ru.

7. Комфортная работа с Android Studio [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/ru/post/433604/.

8. Adarsh F. Android Studio 4.0 [Електронний ресурс] / Fernando Adarsh. – 28. – Режим доступу до ресурсу: https://android-developers.googleblog.com/2020/05/android-studio-4.html.

9. Сайт Александра Климова. Теория [Електронний ресурс] – Режим доступу до ресурсу: http://developer.alexanderklimov.ru/android/theory/.

10. Копитко М. Основи програмування мовою Java / М. Копитко, К. Іванків. – Львів, 2002. – 83 с. – (Видавничий центр ЛНУ імені Івана Франка).

ДОДАТОК А

Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <androidx.recyclerview.widget.RecyclerView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/recyclerView"
        android:scrollbars="vertical"
        >
    </androidx.recyclerview.widget.RecyclerView>

</LinearLayout>
```

Recycler_row_item.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:orientation="vertical">

    <androidx.cardview.widget.CardView
        android:id="@+id/myCardView"
```

```
android:layout_width="wrap_content"
android:layout_height="250dp"
app:cardElevation="3dp"
android:layout_margin="4dp"
app:cardUseCompatPadding="true"
>

<LinearLayout
   android:layout_width="match_parent"
   android:weightSum="5.5"
   android:orientation="vertical"
   android:layout_height="match_parent">

   <ImageView
      android:id="@+id/ivImage"
      android:layout_width="match_parent"
      android:layout_height="0dp"
      android:layout_weight="4"
      android:scaleType="centerCrop"
      android:src="@drawable/image"

      />

   <LinearLayout
      android:layout_width="match_parent"
      android:layout_height="0dp"
      android:weightSum="2"
      android:layout_weight="1.5"
      >
      <LinearLayout
         android:layout_width="match_parent"
         android:layout_height="match_parent"
```

```
    android:padding="10dp"
    android:orientation="vertical"
    >
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="Title"
        android:textColor="@color/colorPrimary"
        android:textSize="18sp"
        android:maxLines="1"
        android:id="@+id/tvTitle"
        android:textStyle="bold"
        />
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="Description"
        android:textStyle="bold"
        android:textSize="15sp"
        android:maxLines="1"
        android:id="@+id/tvDescription"
        />


  </LinearLayout>



</LinearLayout>
```

```
        </LinearLayout>



    </androidx.cardview.widget.CardView>


</LinearLayout>
```

Activity_detail.xml

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".DetailActivity"
    >
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical"
        >
        <ImageView
            android:layout_width="match_parent"
            android:layout_height="200dp"
            android:id="@+id/ivImage2"
            android:src="@drawable/image"
            android:scaleType="centerCrop"
```

```
    />
<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:gravity="center"
    android:layout_marginTop="10dp"
    android:padding="20dp"
    android:text="Description"
    android:textSize="19sp"
    android:id="@+id/txtDescription"
    />
<ImageView
    android:layout_width="match_parent"
    android:layout_height="25dp"
    android:src="@drawable/photo4"
    android:scaleType="centerCrop"
    />
<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="10dp"
    android:text="Ingredients"
    android:textSize="19sp"
    android:id="@+id/txtIngredients"
    />
<ImageView
    android:layout_width="match_parent"
    android:layout_height="25dp"
    android:src="@drawable/photo1"
    android:scaleType="centerCrop"
    />
<TextView
```

```
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="10dp"
            android:text="Recipe"
            android:textSize="19sp"
            android:id="@+id/txtRecipe"
            />
        <ImageView
            android:layout_width="match_parent"
            android:layout_height="25dp"
            android:src="@drawable/ph5"
            android:scaleType="centerCrop"
            />
    </LinearLayout>
</ScrollView>
```

FoodData.java

```
package com.example.book2;

public class FoodData {

    private String itemName;
    private String itemDescription;
    private String itemIngredients;
    private String itemRecipe;
    private int itemImage;

    public FoodData(String itemName, String itemDescription, String itemIngredients, String
itemRecipe, int itemImage) {
        this.itemName = itemName;
        this.itemDescription = itemDescription;
```

```
        this.itemIngredients = itemIngredients;
        this.itemRecipe = itemRecipe;
        this.itemImage = itemImage;
    }

    public String getItemName() {
        return itemName;
    }

    public String getItemDescription() {
        return itemDescription;
    }

    public String getItemIngredients() {
        return itemIngredients;
    }

    public String getItemRecipe() {
        return itemRecipe;
    }

    public int getItemImage() {
        return itemImage;
    }
}
```

MyAdapter.java

```
package com.example.book2;

import android.content.Context;
import android.content.Intent;
```

```
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.cardview.widget.CardView;
import androidx.recyclerview.widget.RecyclerView;

import java.util.List;


public class MyAdapter extends RecyclerView.Adapter<FoodViewHolder>{

    private Context mContext;
    private List<FoodData> myFoodList;

    public MyAdapter(Context mContext, List<FoodData> myFoodList) {
        this.mContext = mContext;
        this.myFoodList = myFoodList;
    }

    @Override
    public FoodViewHolder onCreateViewHolder(@NonNull ViewGroup viewGroup, int i) {
        View mView =
LayoutInflater.from(viewGroup.getContext()).inflate(R.layout.recycler_row_item,viewGroup,false)
;

        return new FoodViewHolder(mView);
    }
```

```
    @Override
    public void onBindViewHolder(@NonNull final FoodViewHolder foodViewHolder, int i) {

        foodViewHolder.imageView.setImageResource(myFoodList.get(i).getItemImage());
        foodViewHolder.mTitle.setText(myFoodList.get(i).getItemName());
        foodViewHolder.mDescription.setText(myFoodList.get(i).getItemDescription());

        foodViewHolder.mCardView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                Intent intent = new Intent (mContext,DetailActivity.class);

intent.putExtra("Image",myFoodList.get(foodViewHolder.getAdapterPosition()).getItemImage());

intent.putExtra("Description",myFoodList.get(foodViewHolder.getAdapterPosition()).getItemDescr
iption());

intent.putExtra("Ingredients",myFoodList.get(foodViewHolder.getAdapterPosition()).getItemIngre
dients());

intent.putExtra("Recipe",myFoodList.get(foodViewHolder.getAdapterPosition()).getItemRecipe());



                mContext.startActivity(intent);
            }
        });

    }

    @Override
    public int getItemCount() {
        return myFoodList.size();
    }
```

```
}

class FoodViewHolder extends RecyclerView.ViewHolder{

    ImageView imageView;
    TextView mTitle,mDescription;
    CardView mCardView;

    public FoodViewHolder( View itemView) {
        super(itemView);

        imageView = itemView.findViewById(R.id.ivImage);
        mTitle = itemView.findViewById(R.id.tvTitle);
        mDescription = itemView.findViewById(R.id.tvDescription);



        mCardView = itemView.findViewById(R.id.myCardView);


    }
}
```

DetailActivity.java

```
package com.example.book2;

import android.os.Bundle;
import android.widget.ImageView;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;
```

```
public class DetailActivity extends AppCompatActivity {

    TextView foodDescription;
    TextView foodIngredients;
    TextView foodRecipe;
    ImageView foodImage;

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

        foodDescription =(TextView)findViewById(R.id.txtDescription);
        foodImage = (ImageView)findViewById(R.id.ivImage2);
        foodIngredients=(TextView)findViewById(R.id.txtIngredients);
        foodRecipe =(TextView)findViewById(R.id.txtRecipe);


        Bundle mBundle = getIntent().getExtras();
        if(mBundle!=null){

            foodDescription.setText(mBundle.getString("Description"));
            foodIngredients.setText(mBundle.getString("Ingredients"));
            foodRecipe.setText(mBundle.getString("Recipe"));
            foodImage.setImageResource(mBundle.getInt("Image"));

        }

    }
}
```

MainActivity.java

```
package com.example.book2;

import android.os.Bundle;

import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.GridLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import java.util.ArrayList;
import java.util.List;

public class MainActivity extends AppCompatActivity {

    RecyclerView mRecyclerView;
    List<FoodData> myFoodList;
    FoodData mFoodData;

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

        mRecyclerView = (RecyclerView)findViewById(R.id.recyclerView);

        GridLayoutManager gridLayoutManager =  new GridLayoutManager(MainActivity.this,1);
        mRecyclerView.setLayoutManager(gridLayoutManager);

        myFoodList = new ArrayList<>();
```

```
mFoodData = new FoodData ("Чікен-рол","Закуски * Європейська кухня * Роли","Вірменський лаваш 5 штук\n" +
"Кетчуп - 200 г\n" +
"Гірчиця - 150 г\n" +
"Майонез - 150 г\n" +
"Твердий сир - 10 чайних ложок\n" +
"Помідори - 1 штука\n" +
"Солоні огірки - 3 штуки\n" +
"Зелений салат - 1 пучок\n" +
"Куряча грудка - ½ штуки\n" +
"Панірувальні сухарі - за смаком\n" +
"Сметана - 4 столові ложки\n" +
"Яйце куряче - 3 штуки\n" +
"Манна крупа - 3 столові ложки\n" +
"Сіль - за смаком\n" +
"Рослинна олія - за смаком\n" +
"Перець чорний мелений - за смаком","1. Філе злегка відбити, як на відбивні. Порізати на шматочки вздовж грудки. Посолити, поперчити, занурити в 1 збите яйце, потім в панірувальні сухарі і - на розігріту сковороду, смажити в олії.\n" +
"\n" +
"2. Омлет: 2 яйця збити, додати сіль, перець, молоко / сметана, додати манку. Влити на сковороду і готувати на маленькому вогні до готовності. Коли охолоне, порізати на довгі смужки.\n" +
"\n" +
"3. Помідор і огірки нарізати кружечками, сир натерти на тертці. Листя салату добре вимити і висушити.\n" +
"\n" +
"4. У глибокій тарілці майонез змішати з кетчупом.\n" +
"\n" +
"\n" +
"5. На велику тарілку / блюдо викласти лаваш, змастити гірчицею. Викласти кілька листя салату, помідори і огірків, потім додаємо м'ясо, омлет, 2 чайні ложки сиру, зверху викласти соус (3-4 столові ложки).\n" +
"\n" +
"6. Тепер звертаємо наш рол і насолоджуємося :)",R.drawable.image);
```

```
myFoodList.add(mFoodData);

mFoodData = new FoodData("Сирний суп по-французьки з куркою","Основна страва*Франція",

"Куряче філе - 500 г\n" +

"Плавлений сир - 200 г\n" +

"Картофель - 400 г\n" +

"Лук - 150 г\n" +

"Морковь - 180 г\n" +

"Вершкове масло - за смаком\n" +

"Сіль - за смаком\n" +

"Перець чорний мелений - за смаком\n" +

"Зелень - за смаком\n" +

"Лавровий Ліст - 3 штуки\n" +

"Грінки - за смаком\n" +

"Чорний перець горошком - 2 штуки",

"1.У каструлю на 3 літри покласти м'ясо і налити води. Як тільки бульйон почне кипіти, додати 1 чайну ложку солі, пару горошків запашного перцю і чорного, 2-3 листочка лаврового листа. Варити від моменту закипання 20 хвилин. Потім м'ясо вийняти.\n" +

"\n"+

"2. Картоплю почистити і нарізати кубиками. Цибулю нарізати кубиками. Моркву натерти на тертці. М'ясо порізати невеликими шматочками. Плавлений сир (якщо у вигляді брусочки) натерти на тертці або порізати кубиками.\n" +

"\n"+

"3. У киплячий бульйон додати картоплю. З моменту закипання 5-7 хвилин.\n" +

"\n"+

"4. У цей час зробити слабку зажарку на вершковому маслі. Спочатку покласти цибулю, потім моркву. Злегка посолити і поперчити. Готову зажарку додати в суп і варити ще 5-7 хвилин.\n" +

"\n"+

"5. Потім додати порізане м'ясо. Варити 3-4 хвилини, додати плавлений сир, гарненько перемішать і вимкнути вогонь.\n" +

"\n"+

"6. Перед подачею посипати зеленню. За бажанням подавати з грінками",R.drawable.image2);
```

```
myFoodList.add(mFoodData);

mFoodData = new FoodData("Салат зі смаженим баклажаном і перцем","Салати*Веганські їжа",

        "Мікс Тоскана «Біла Дача» - 200 г\n" +

            "Баклажани - 1 штука\n" +

            "Перець чілі - 2 штуки\n" +

            "Кінза - 50 г\n" +

            "Чеснок - 2 зубчики\n" +

            "Червоний солодкий перец - 1 штука\n" +

            "Волоські орехі - 50 г\n" +

            "Лімон - 1 штука\n" +

            "Зерна граната - 80 г\n" +

            "Оливкова масло - 20 мл\n" +

            "Горіхове масло - 30 мл\n" +

            "Сахар - 1 чайна ложка\n" +

            "Мелений чорний перец - за смаком\n" +

            "Сіль - за смаком",

        "1. Солодкий перець змастити олією і відправити в духовку, розігріту до 200 градусів на функції «гриль», на 12 хвилин.\n" +

            "\n"+

            "2. Баклажан нарізати кільцями товщиною 1 см, потім розрізати на 4 сектори. Перекласти баклажан в чашу, посолити, перемішати і дати постояти хвилин 20.\n" +

            "\n"+

            "3. Чилі позбавити від насіння і нарізати соломкою. Кінзу, часник і волоські горіхи подрібнити.\n" +

            "\n"+

            "4. Баклажан промити, обсушити на паперових рушниках і обсмажити на сухій сковороді.\n" +

            "\n"+

            "5. Зняти шкіру з запеченого перцю, видалити насіння і нарізати кубиком. У великій чаші змішати сік лимона, цукор, сіль, перець, часник, потім збити, поступово вливаючи оливкова і горіхове масло.\n" +

            "\n"+
```

```
"6. До соусу покласти смажені баклажани і добре перемішати, потім додати решту підготовлені інгредієнти. В кінці додати зерна граната і салат Тоскана. Перемішати і подавати.",R.drawable.image3);

        myFoodList.add(mFoodData);

        mFoodData = new FoodData ("Класичний ростбіф","Основні страви*Англійська кухня",

            "Яловича вирізка - 600 г\n" +

                "Цибуля ріпчаста - 1 головка\n" +

                "Морква - 1 штука\n" +

                "Стебло селери - 1 штука\n" +

                "Чебрець - 6 стебел\n" +

                "Рослинна олія - 30 мл\n" +

                "Сіль - за смаком\n" +

                "Перець чорний молотий - за смаком","1. Яловичу вирізку очистити від плівок і жиру і перев'язати кулінарною ниткою по всій довжині так, щоб вона перетворилася в акуратну рівну ковбаску.\n" +

                "\n" +

                "2. На рослинному маслі в сильно розігрітій сковороді обсмажити вирізку до характерного коричневого кольору і приємного смаженого запаху.\n" +

                "\n" +

                "3. Перекласти вирізку в деко і на тому ж маслі злегка обсмажити крупно нарізані цибулю, селеру і моркву.\n" +

                "\n" +

                "4. Додати до овочів чебрець і відправити їх в деко з вирізкою. Запікати приблизно сорок хвилин при температурі духовці 180 градусов.Перед подачею дати м'ясу відпочити приблизно 20 хвилин.",R.drawable.image4);

        myFoodList.add(mFoodData);









        MyAdapter myAdapter = new MyAdapter(MainActivity.this,myFoodList);

        mRecyclerView.setAdapter(myAdapter);

    }

}
```