МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних наук,
управління та адміністрування
Кафедра інформаційних технологій

Бакалаврська кваліфікаційна робота

на тему: Аналіз природно-заповідного фонду України в контексті сталого розвитку територій

Виконав студент 4 курсу групи К-42
Напрям 122 комп'ютерні науки,
Равічев Максим Сергійович

_____ _

Керівник асистент
Шуптар-Пориваєва Наталія Йосипівна

Консультант д.т.н., професор
Кругляк Юрій Олексійович

Рецензент к.геогр.н., доцент
Бургаз Олексій Анатоолійович

Одеса 2019

ЗМІСТ

Перелік скорочень та термінів 5
Вступ 6
1 Аналіз предметної області 7
1.1 Характеристика об’єкту розробки 7
1.2 Опис предметної області 7
1.3 Аналіз існуючих аналогів 9
1.3.1 «Калькулятор калорий ХиКи» 9
1.3.2 Додаток «Худеем вместе. Дневник калорий» 10
1.4 Постановка задачі 13
2 Характеристика обраних програмних засобів 14
2.1 Мова програмування Java 14
2.2 Середовище програмування Android Studio 16
2.3 Система керування базами даних SQLite 17
3 Проектування системи 19
3.1 Побудова функціональних діаграм 19
3.2 Проектування бази даних 24
3.3 Проектування структури додатку 25
4 Реалізація інформаційної системи 26
4.1 Реалізація екранів мовою XML 26
4.2 Створення бази даних 34
4.3 Написання коду Java 37
4.3.1 Метод DeshifrEDAOPS 37
4.3.2 Метод DeshifrMAINOPS 39
4.3.3 Метод ShifrMAINOPS 42
Висновки 45
Перелік джерел посилання 46

ПЕРЕЛІК СКОРОЧЕНЬ ТА ТЕРМІНІВ

Скорочення

БД – База даних

ПК – Персональний комп’ютер

СКБД – Система керування базами даних

DB – Data base

IDE – Integrated Development Environment

SQL – Structured Query Language

XML – eXtensible Markup Language

ВСТУП

Останнім часом розширилося застосування персональних комп'ютерів для виконання різнорідних завдань. Сьогодні всі організації, як державні так і приватні, використовують обчислювальні машини у роботі. ПК може працювати більш продуктивно і дешевше для підприємства. Окрім цього автоматизація робочих місць виключає людський фактор. Через це людей і замінюють на ПК і програми.

Не є виключенням й електронні калькулятори калорій. Повна автоматизація робочих місць не бажана, але неминуча. Але якщо брати у розрахунок людську природу, то люди прагнуть робити як найменше і отримувати як найбільше. Це веде до не здорового способу життя – тобто до ожиріння. Тут мають допомагати лікарі-дієтологи. Аби дієта була збалансована, треба мати свого власного дієтолога. Це дорого, та не завжди практично, бо для людей, у яких життя розписане по хвилинам, ходити на прийом дієтолога раз у тиждень – непомірна розкіш.

Тут і приходить на допомогу власний менеджер раціону. Складання власної дієти, та слідкування за кількістю вжитих калорій більш складна справа ніж здається. Вести облік важливо не тільки людям які бажають схуднути, або навпаки набрати вагу. Це життєво необхідно людям з обмеженим харчуванням та лікарськими дієтами.

Менеджер раціону має обчислювати по вхідним даним норму кілокалорій в день і процентне відношення жирів, білків, вуглеводів. По отриманим даним є можливість скласти власний раціон. Тобто крім усього, у програмі має міститися таблиця страв і кількість калорій, жирів, білків, вуглеводів у них, на 100 грам. Завдяки цьому можна вирахувати калорійність будь якої кількості тої або іншої страви.

Пояснювальна записка містить 46 сторінок, 13 рисунків, 4 таблиці, 6 джерел посилання.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Характеристика об’єкту розробки

Дієтологія є важливою наукою і не менш важливою частиною сучасної медицини. Важливою вона є не тільки для спортсменів або моделей. Адже не раціональне вживання їжі може призвести не тільки до схуднення або набору ваги, а і до хвороб шлунку. Аби не допустити цього, багато людей через мас. медіа розкривають важливі факти щодо правильного харчування.

Є багато ресурсів з яких можна дізнатися про поживність тієї або іншої їжі. Але більшість цих ресурсів не дає можливості робити підрахунки протягом усього дня. Сьогодні, людина завжди знаходиться в русі. В неї нема часу сісти з олівцем та блокнотом і вираховувати скільки білків вона вжила протягом дня.

Метою є розробка андроїд-додатку менеджер раціону, який надає можливість вирахування денної норми калорій, та несе роль облікового додатку вживаних калорій протягом дня. Завдяки щоденному списку вживаних страв, які заносить туди користувач, є можливість відстежувати кількість калорій, процентне відношення калорій, жирів та білків. Це, в свою чергу, дає можливість виводити статистику за будь який проміжок часу.

1.2 Опис предметної області

Дієтологія це, наука про раціональне харчування здорової и хворої людини. Під раціональним харчуванням розуміють правильно організоване, фізіологічно і клінічно обґрунтоване постачання організму добре приготовленої і смачної їжі, що містить оптимальну кількість всіх харчових речовин, необхідних для його розвитку і функціонування. Таке харчування має сприяти збереженню здоров'я, гарного самопочуття, максимальної тривалості життя, поліпшення адаптації до стресових впливів. Крім того, спеціально підібрані дієти – лікува-

льне харчування – є одним з важливих, а іноді і основним методом лікування багатьох захворювань.

При складанні дієт зазвичай виділяють наступні критерії для оцінки раціонів харчування:

- енергетична цінність їжі (вона повинна відповідати затратам енергії людиною в процесі життєдіяльності);
- збалансованість харчування (їжа повинна містити оптимальне, якісне і кількісне співвідношення всіх необхідних людині харчових речовин, в тому числі і незамінних, відповідно до потреб окремої людини або груп населення);
- спосіб технологічної обробки їжі (він повинен виключати утворення токсичних сполук під час приготування їжі, не викликати зменшення її біологічної цінності);
- режим харчування (розподіл прийомів їжі протягом доби має відповідати режиму і характеру трудової та інших видів діяльності).

Для людей з різними захворюваннями, Інститутом харчування розроблені дієти, які затверджені Міністерством охорони здоров'я як обов'язкові для всіх лікувально-профілактичних установ. Вони мають номерну систему (з 1 по 15). Крім перерахованих вище критеріїв при їх складанні використовуються принципи механічного, фізичного і хімічного щадіння хворих органів, враховуються особливості порушення обміну речовин при кожному з захворювань[1][1)].

В ідеалі для кожної окремо взятої людини раціон харчування повинен підбиратися індивідуально. При цьому дієтологи повинні враховувати не тільки стать, вік, професію, інтенсивність фізичних навантажень, наявність захворювань, але і кліматичні умови і особливості національного харчування тієї групи людей, до якої належить дана людина.

---

1) [1] Раціональне харчування: закони та принципи. URL: https://pidruchniki.com/85492/bzhd/ratsionalne_harchuvannya_zakoni_printsipi. (дата звернення 12.02.2019).

Наприклад, в Гренландії, де ще недавно 100% раціону становило м'ясо, багато людей не можуть перетравлювати вуглеводи рослинного походження. А представники кочового племені тубу, що живе в самому центрі Сахари, взагалі не знають смаку м'яса і обходяться декількома фініками на обід, жменею проса, змоченого пальмовою олією або товченими корінням на вечерю. Для багатьох європейців такий раціон міг би стати згубним.

Таким чином, адекватне дієтичне харчування має на увазі відповідність раціону як характером обміну речовин в організмі, так і сформованим в процесі еволюції особливостям переробки їжі в шлунково-кишковому тракті. Більшість вчених і медиків вважають, що люди відрізняються один від одного за особливостями обміну речовин і всім їм не можна рекомендувати однакову їжу, навіть якщо вона корисна і безпечна для переважної більшості людства.

1.3 Аналіз існуючих аналогів

1.3.1 «Калькулятор калорий ХиКи»

Додаток «Калькулятор калорий ХиКи» сама розрахує необхідну норму калорій і БЖВ відповідно до ваших параметрів і активності! Але головний функціонал додатку – це щоденник калорій. Окрім вжитих калорій, також рахуються калорії які були витрачені внаслідок активності протягом дня.

Також додаток надає багато статистичної інформації щодо вжитих або витрачених калорій протягом заданого проміжку часу [2][2)].

Плюси:

- багатофункціональність;
- інформативний інтерфейс;
- велика база данних.

Мінуси:

[2)] [2] Android-додаток «Калькулятор калорий ХиКи». URL: https://play.google.com/store/apps/details?id=ru.hikisoft.calories. (дата звернення 26.03.2019).

– Не зручний інтерфейс – у звичайного користувача на те, що би розібратися і почати користуватися додатком піде деякий час (рис. 1);

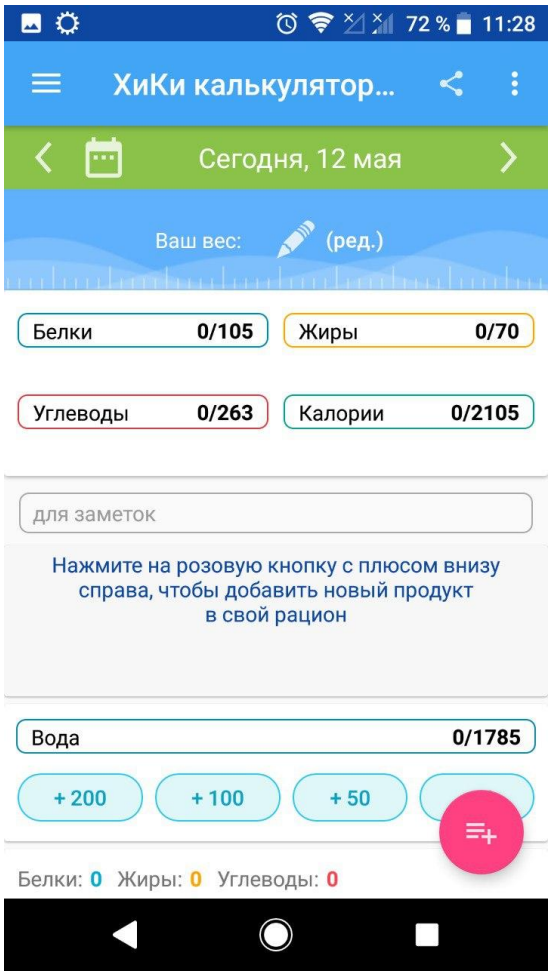

Рисунок 1 – Додаток «Калькулятор калорий ХиКи»

1.3.2 Додаток «Худеем вместе. Дневник калорий»

Найпопулярніший додаток для схуднення і підтримки здорового способу життя [3][3)]!

Що може запропонувати додаток:

[3)] [3] Android-додаток «Худеем вместе. Дневник калорий»». URL: https://play.google.com/store/apps/details?id=ru.hudeem.adg. (дата звернення 26.03.2019).

– дуже широка база страв, по якій можна визначити кількість споживаних калорій. Вона включає в себе не тільки страви домашнього приготування, а й ті, які можна придбати в магазині.

Особливо широко представлений асортимент швидкого харчування і покупних готових страв;

– список дієт. У кожної з дієт представлені не тільки набір денне меню, а й історія виникнення, а також велику кількість іншої доступної інформації, аби користувач міг легше зробити свій вибір;
– розділ з демотиваторами. Даний розділ є дуже корисним для тих людей, які найкраще сприймає наочну візуалізацію. У ньому представлено багато зображень і текстів, які дають можливість не забувати про те, для чого починався весь цей шлях, і дотримуватися його в подальшому;
– вправи. В цьому розділі завжди можна буде знайти вправи для підтримки себе у формі. Більш того, можна буде розробити спеціальну програму, згідно з якою будуть проходити періодичні тренування. Дозволяє програмі допомогти і в цьому;
– корисне. В даному розділі зібрані дуже багато статей про схуднення. У тому числі багато міфів, через які люди вибирають свій хибний шлях до схуднення, а також корисні поради, що містяться у величезній кількості важливої інформації для тих, що худнуть;
– вітаміни. Цей розділ пропонує користувачам інформацію про те, які існують вітаміни і для чого вони потрібні організму;
– розрахунки. Дана категорія дуже важлива, адже включає в себе можливість спостерігати за динамікою схуднення, бачити зміни в харчуванні, а також здійснювати контроль за споживанням їжі. Також тут представлена база продуктів, таблиця продуктів, розрахунок індексу маси та інші, не менш корисні розділи. Наприклад, калькулятори культуристів. Даний розділ призначений для людей, які займаються цим видом спор-

ту професійно, адже їхні потреби відрізняються від потреб людини що худне (рис. 2);

Рисунок 2 – Додаток «Худеем вместе. Дневник калорий»

За допомогою різноманітних дієт і калькуляторів, можна дізнатися багато нового про режими харчування, ви можете вести свій щоденник, який дуже допомагає в курсі схуднення. Графік БЖУ допомагає відслідковувати весь архів з щоденника в зручному графічному вигляді.

У додатку зібрано велику колекцію дієт як від простих людей, так і від інститутів краси харчування і здоров'я. Наприклад, такі як:

– дієта Аткінса;
– «Велика дієта»;
– «Економна» ;
– Олени Малишевої;
– дієта інституту харчування АМН СРСР;
– «Кремлівська дієта» і багато інших.

Плюси:

- зручний і красивий інтерфейс;
- багатофункціональність;
- можливість добавляти коментарі.

Мінуси:

- реклама.

1.4 Постановка задачі

Розробити Android додаток для розрахунку вживаних калорій. Функціональні можливості додатку для користувача:

- розрахунок ідеальної ваги;
- розрахунок денної норми калорій ;
- розрахунок кількості жирів, білків та вуглеводів в заданій кількості продукту;
- перегляд бази даних продуктів та їх поживності;
- додавання продуктів до категорії вживаних;
- розрахунок усіх вживаних протягом дня продуктів.

Для досягнення цієї мети необхідно:

- вивчити літературу Android-програмування;
- вивчити мову програмування Java;
- дослідити сучасні методи створення Android-додатків;
- проаналізувати аналоги;
- обрати та обґрунтувати архітектуру та програмні засоби для проектування та реалізації додатку.
- спроектувати базу даних та застосування;

– розробити власний Android-додаток з використанням отриманих знань;
– провести тестування.

2 ХАРАКТЕРИСТИКА ОБРАНИХ ПРОГРАМНИХ ЗАСОБІВ

2.1 Мова програмування Java

Для створення свого програмного продукту я вибрав мову програмування Java. Тому що на Java працює понад три мільйони гаджетів – від смартфонів і кавоварок до систем "розумний будинок" і електрокарів.

За результатами опитування Eclipse Foundation за 2016 рік, Java – ліди-руюча

Давайте детальніше розглянемо кожне з них.

– Android програми – якщо хочете побачити, де використовується Java, не потрібно далеко йти. Просто візьміть свій телефон на Android, абсолютно всі програми написані на Java, з використанням Google і Android API, які схожі з JDK. Пару років назад Android надав необхідні можливості, завдяки чому сьогодні багато Java програмісти – Android розробники. До речі, Android використовує іншу JVM і інший і інший спосіб компоновки, але код все ще написаний на Java.
– Серверні додатки в сфері фінансових послуг – Java дуже широко застосовується у фінансовій сфері. Багато світові інвестиційні банки, типу Goldman Sachs, Citigroup, Barclays, Standard Charted і інші використовують Java для написання фронт-енд і бек-енд офісних електронних систем, систем регулювання і конфірмації, проектів обробки даних і деяких інших. Переважно Java використовується при написанні серверних додатків, в більшості своїй без будь-якого призначеного для користувача інтерфейсу, які отримують дані з одного сервера, обробляють їх і відправляють далі. Java Swing був також популярний

для створення «толстокліентних» інтерфейсоф, але зараз C # швидко захоплює ринок в цій області, а Swing вже видихається.

- Веб-додатки – також Java широко використовується в електронній комерції і в області веб-додатків. Величезна кількість RESTful сервісів було створено з використанням Spring MVC, Struts 2.0 і схожих фрейм-мворків. Навіть найпростіші додатки, засновані на Servlet, JSP і Struts, досить популярні в різних державних проектах. Багато веб-додатки державних, оздоровчих, страхових, освітніх,оборонних та деяких інших відділень написані на Java.
- Програмні засоби – багато корисні програмні засоби і засоби розробки написані і розроблені на Java, наприклад Eclipse, IntelliJ Idea і Netbeans IDE. Мені здається це, до того ж, найбільш використовувані додатки, написані на Java. Був час, коли Swing був дуже популярний при створенні «товстих клієнтів», переважно у фінансовій сфері. Сьогодні Java FX набирає все більшої популярності, але це все ще не заміна Swing, до того ж C # практично повністю витіснив Swing з фінансової області.
- Трейдингові додатки – сторонні трейдинговие додатки, які також частина великої індустрії фінансових послуг, теж використовують Java. Популярні додатки, типу Murex, які використовуються в багатьох банках, написані на Java.
- Вбудовувані системи – широка Java і в області вбудованих систем. Можна побачити на що здатна платформа, вам потрібно всього 130 KB для використання Java (на смарт-картах і сенсорах). Спочатку Java розроблялася для вбудованих систем. Насправді ця область була частиною початкової кампанії Java «пиши один раз, запускай десь завгодно» і схоже, що вона приносить свої плоди.

З цієї причини Java популярна і при написанні високопродуктивних систем, тому що хоч продуктивність програє в порівнянні з рідною мовою, але ви

можете пожертвувати безпекою, мобільністю і надійністю заради більшої швидкості і потрібно всього один недосвідчений C ++ программист, щоб зробити додаток повільним і ненадійним.

В наші дні часто Java - вибір за замовчуванням в наукових додатків, включаючи обробку природної мови. Основна причина в тому, що Java більш безпечна, мобільна і надійна і має кращі інструменти паралелізації, ніж C ++ та іншими мовами.

2.2 Середовища програмування Android Studio

Для розробки додатку, мною було розглянуто 3 середовища розробки. А саме – Eclipse, IntellJ IDEA і Android Studio.У грудні 2014 року вийшла версія 1.0 середовища розробки Android Studio, заснованого на базі IntellJ IDEA і призначеного саме для комфортної розробки android-додатків. Того же року була зупинена підтримка плагіну Android Development Tools (ADT) для Eclipse.

Для створення свого програмного продукту я обрав середовище розробки Android Studio. Це середовище розробки забезпечує найшвидші інструменти для побудови додатків на кожному типі Android пристрою.

Сучасне інтегроване середовище, створене спеціально для розробників Android-додатків. Включає в себе колосальний перелік можливостей для роботи з кодом, файлами, тестування і калібрування додатків.

Розроблена компанією Google, Android Studio покликана зробити процес написання програм для android настільки зручним і ефективним, наскільки це можливо. Уклавши в собі самі передові і унікальні розробки, це дивовижне середовище стало справжньою панацеєю в індустрії Android Development, замінивши собою популярний Eclipse [5][5)].

Інтерфейс програми досконально продуманий і дуже зручний у використанні, не дивлячись на величезний набір інструментів. Опрацьована структура

[5)] [5] П. Дейтел, Х. Дейтел, А. Уолд. Android для разработчиков. 3-е изда-ние, 2016 П. Дейтел, Х. Дейтел, А. Уолд: Питер. 14 с.

проекту, де розробник може легко взаємодіяти з елементами програми та навіть перетягувати їх за допомогою миші. Будь-які зміни в проекті, можна спостерігати в реальному часі і навіть на різних пристроях. Завдяки унікальній системі тестування, ви можете дізнатися, скільки ресурсів споживає додаток, і якщо буде потрібно оптимізувати його.

Деякі особливості будуть пізніше розгорнуті для користувачів так як програмне забезпечення розвивається; наразі, передбачені такі функції:

- живі макети (layout): редагувальник WYSIWYG – живе кодування – подання (rendering) програми в реальному часі;
- консоль розробника: підказки по оптимізації, допомога по перекладу, стеження за напрямком, агітації та акції – метрики Google аналітики;
- резерви бета релізів та покрокові релізи;
- базування на Gradle;
- android-орієнтований рефакторинг та швидкі виправлення;
- lint утиліти для охоплення продуктивності, юзабіліті, сумісності версій та інших проблем;
- використання можливостей ProGuard та підписів до програм;
- шаблони для створення поширених Android дизайнів та компонентів;
- багатий редактор макетів (layouts) що дозволяє користувачам перетягнути і покласти (drag-and-drop) компоненти користувацького інтерфейсу, як варіант, переглянути одночасно макети (layouts) на різних конфігураціях екранів.

2.3 Система керування базами даних SQLite

В якості СКБД я вирішив обрати SQLite за його зручність та здатність працювати на різноманітних пристроях. SQLite – компактна вбудована реляційна база даних. «Вбудована» означає, що SQLite не використовує парадигму клієнт-сервер, тобто движок SQLite не є окремо працюючим процесом, з яким

взаємодіє програма, а надає бібліотеку, з якої програма компонується і движок стає складовою частиною програми.

Сама бібліотека SQLite написана на C, але існує велика кількість прив'язок до інших мов програмування, в тому числі PHP.

Простота і зручність вбудовування SQLite привели до того, що бібліотека використовується в браузерах, музичних плеєрах і на на будь- якому смартфоні.

При випуску версії вона проходить через ряд найсерйозніших автоматичних тестів (проводиться ~ 2 млн тестів), покриття коду тестами 100% (з серпня 2009).

SQLite є безтиповою базою даних. Точніше, є тільки два типи – цілочисельний "integer" і текстовий "text" [6][6)].

Причому "integer" використовується переважно для первинного ключа таблиці, а для інших даних піде "text". Довжина рядка, що записується в текстове поле, може бути будь-хто.

Всі бази даних зберігаються в файлах, по одному файлу на базу. Кількість баз даних, а так само таблиць в них, обмежено тільки вільним місцем, наявними на сайті.

Оскільки движок бази і інтерфейс до неї реалізовані як єдине ціле, величезний перевагою SQLite є висока продуктивність – для більшості типових завдань додаток, побудоване на SQLite, працює швидше,

Перш за все, SQLite призначена для невеликих і середніх за обсягом додатків. Особливо актуальне використання SQLite в разі, коли в основному проводяться операції запису і зчитування даних.

Однак при надзвичайно активному зверненні до даних або в разі частих угруповань SQLite працює повільніше своїх конкурентів через вбудованого механізму блокування файлів (тільки при модифікації даних) і необхідності перевірки типу полів для вибору способу сортування.

[6)] [6] Сайт «Освой программирование играючи. Сайт Александра Климова» URL: http://developer.alexanderklimov.ru/android. (дата звернення 28.03.2019).

3 ПРОЕКТУВАННЯ СИСТЕМИ

3.1 Побудова функціональних діаграм

На рис. 3 представлена контекстна діаграма створення додатку.

В якості основного процесу беремо створення додатку.

В якості вхідних даних беремо мову програмування Java, мову розмітки XML, мову структурованих запитів SQL та IDE Android Studio.

В якості управляючих беремо Літературу вивчення Java, XML, SQL, Android Studio та інтернет.

В ролі механізмів виступають дизайнер, верстальник, програміст та Android Studio.

На виході ми отримуємо додаток (рис. 3).

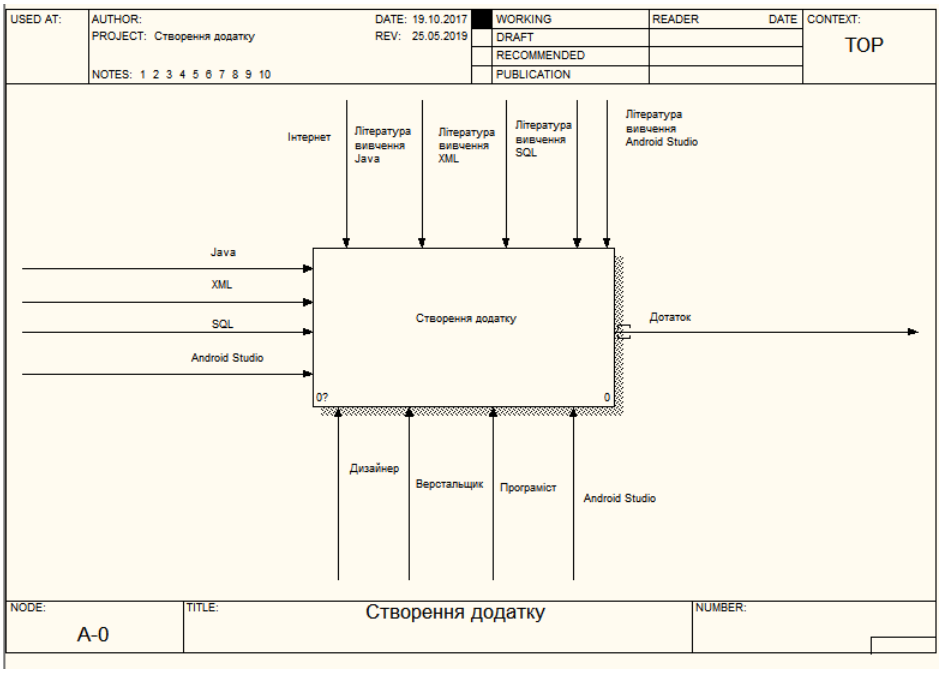

Рисунок 3 – Контекстна діаграма створення додатку

На рис 4 представлена діаграма першого рівня декомпозиції створення додатку.

На першому рівні декомпозиції в якості етапів взяли «Створити графічний інтерфейс», «Створити базу даних», «Написати код».

На етапі «Створити графічний інтерфейс» вхідними даними є XML та Android Studio, управляючі – «Література вивчення Android Studio», «Література вивчення XML» та інтернет, механізми – «Дизайнер», «Верстальник» та «Android Studio», на виході отримуємо графічний інтерфейс.

На етапі «Створити базу даних» вхідними даними є Android Studio, SQL та Java, управляючі – «Література вивчення SQL», «Література вивчення Java» та інтернет, механізми – «Програміст» та Android Studio, на виході отримуємо базу даних.

На етапі «Написати код» вхідними зданими є графічний інтерфейс, Android Studio, база даних та Java, управляючі – «Література вивчення Java», «Література вивчення SQL» та інтернет, механізми – «Програміст» та Android Studio, на виході отримуємо додаток (рис. 4).

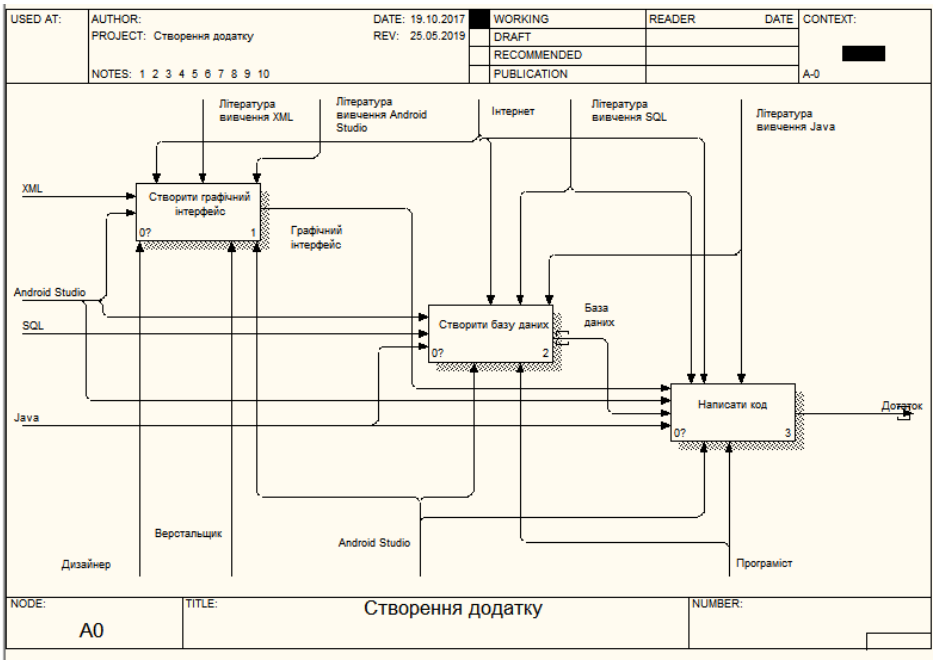

Рисунок 4 – Діаграма декомпозиції створення сайту

На рис. 5 представлено діаграму декомпозиції другого рівня етапу «Створити графічний інтерфейс».

На другому рівні декомпозиції в якості етапів взяли «Додання робочих екранів», «Вибрати типи розподілу елементів на екрані», «Додати елементи», «Налаштування зовнішнього вигляду елементів».

На етапі «Додання робочих екранів» вхідними даними є Android Studio, управляючі – «Література вивчення Android Studio» та інтернет, механізми – дизайнер, верстальщик та Android Studio, на виході отримуємо пусті робочі екрани.

На етапі «Вибрати типи розподілу елементів на екрані» вхідними даними є пусті робочі екрани та Android Studio, управляючі – «Література вивчення Android Studio» та інтернет, механізми – дизайнер, верстальщик та Android Studio, на виході отримуємо розмічені робочі екрани.

На етапі «Налаштування зовнішнього вигляду елементів» вхідними даними є заповнені робочі екрани, XML та Android Studio, управляючі – «Література вивчення Android Studio», «Література вивчення XML» та інтернет, механізми – дизайнер, верстальщик та Android Studio, на виході отримуємо графічний інтерфейс (рис. 5).

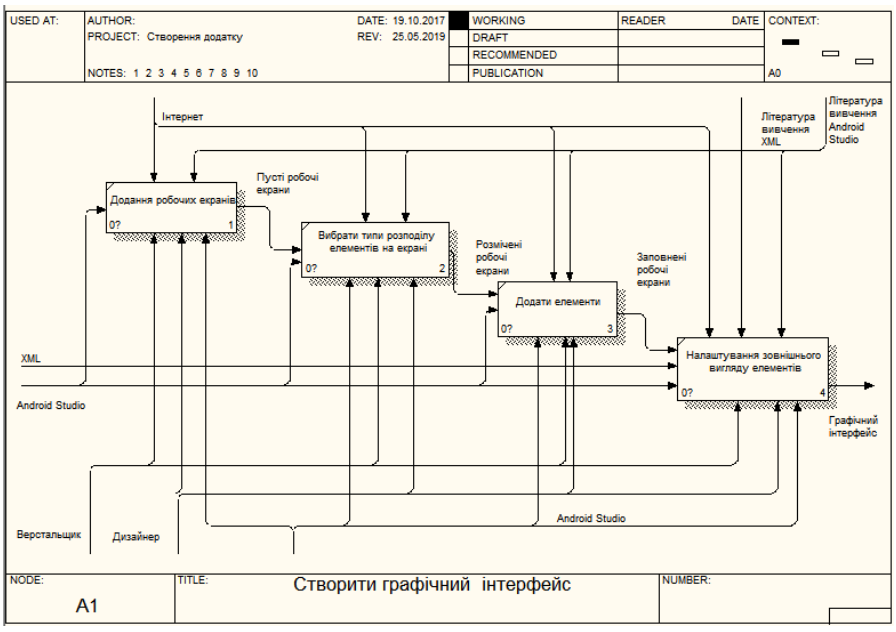

Рисунок 5 – Діаграма декомпозиції створення графічного інтерфейсу

На рис. 6 представлено діаграму декомпозиції другого рівня етапу «Створити базу даних».

На другому рівні декомпозиції в якості етапів взяли «Створення допоміжного класу», «Створення таблиць», «Додавання до бд стартових записів».

На етапі «Створення допоміжного класу» вхідними даними є Android Studio та Java, управляючі – «Література вивчення Java» та інтернет, механізми – програміст та Android Studio, на виході отримуємо DBHelper.

На етапі «Створення таблиць» вхідними даними є DBHelper, SQL та Android Studio, управляючі – «Література вивчення SQL» та інтернет, механізми – програміст та Android Studio, на виході отримуємо таблиці бд.

На етапі «Додавання до бд стартових записів» вхідними даними є SQL, Java, таблиці бд та Android Studio, управляючі – «Література вивчення SQL», «Література вивчення Java» та інтернет, механізми – програміст та Android Studio, на виході отримуємо базу даних (рис. 6).

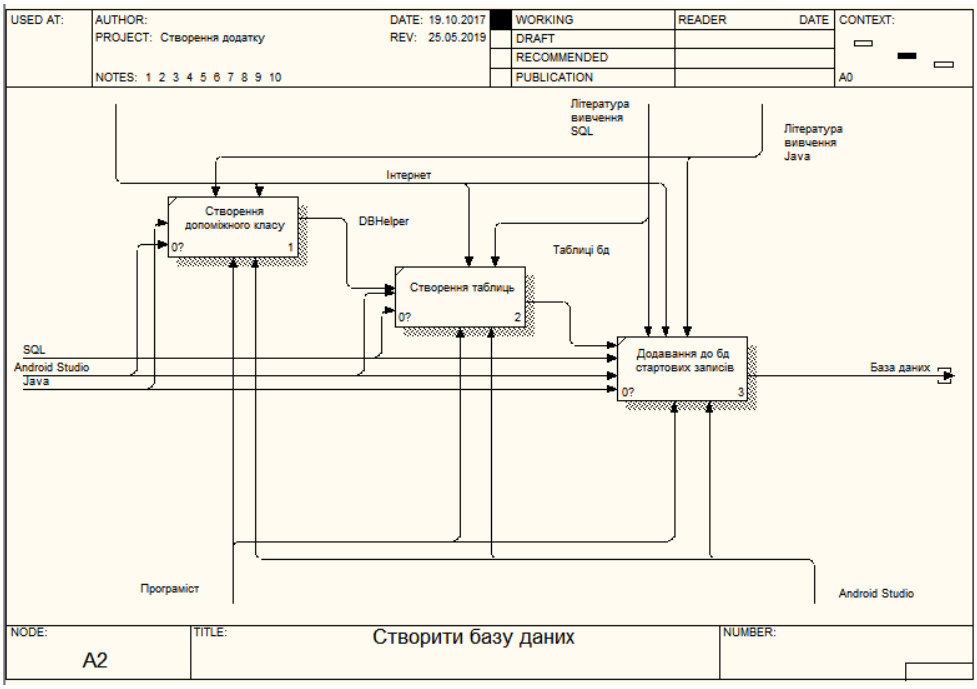

Рисунок 6 – Діаграма декомпозиції створення бази даних

На рис. 7 представлено діаграму декомпозиції другого рівня етапу «Написати код».

На етапі «Обробники подій» вхідними даними є графічний інтерфейс Android Studio та Java, управляючі – «Література вивчення Java» та інтернет, механізми – програміст та Android Studio, на виході отримуємо інформацію.

На етапі «Формули підрахунків» вхідними даними є інформація, дані, Java та Android Studio, управляючі – «Література вивчення Java» та інтернет, механізми – програміст та Android Studio, на виході отримуємо результати підрахунків.

На етапі «Обробник взаємодій з бд» вхідними даними є база даних, Java, результати підрахунків та Android Studio, управляючі – «Література вивчення SQL», «Література вивчення Java» та інтернет, механізми – програміст та Android Studio, на виході отримуємо дані.

На етапі «Виведення інформації» вхідними даними є результати підрахунків, графічний інтерфейс, дані, Java та Android Studio, управляючі – «Література вивчення Java» та інтернет, механізми – програміст та Android Studio, на виході отримуємо додаток (рис. 7).

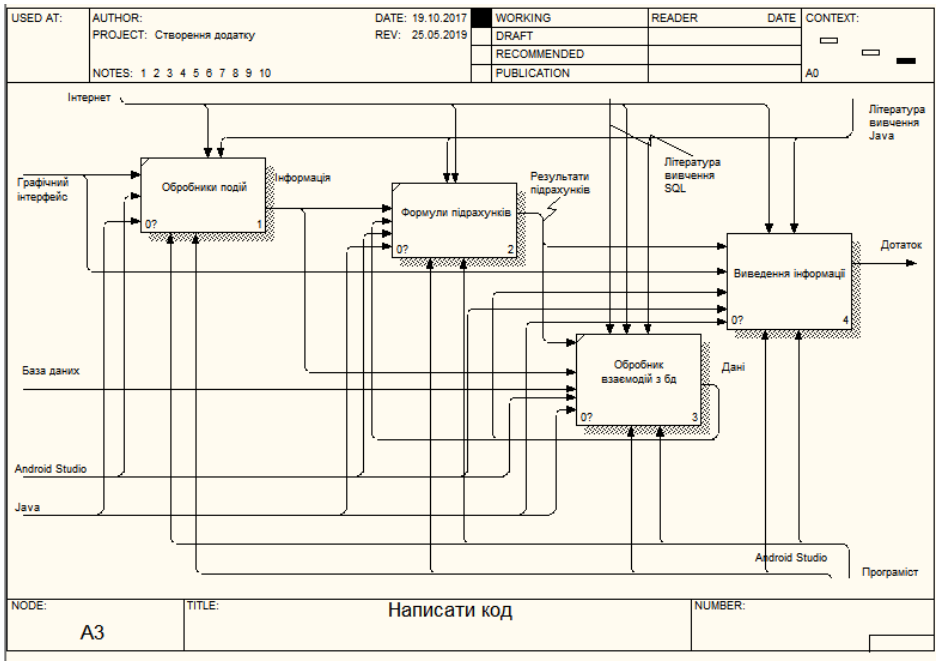

Рисунок 7 – Діаграма декомпозиції написання коду

3.2 Проектування бази даних

База даних була створена за допомогою допоміжного класу DBHelper і складається з 4 таблиць.

Табл. 1 призначена для зберігання їжі та її параметрів.

Таблиця 1 – Структура таблиці TABLE_EDA

Назва поля	Тип	Опис поля
EDA_ID	INTEGER PRIMARY KEY	Унікальний ідентифікатор їжі
EDA_NAME	TEXT	Найменування їжі
EDA_CLASS	integer	Тип їжі
EDA_OPS	TEXT	Кількість жирів, білків та вуглеводів у 100 грамах їжі, записаних у вигляді «8.8;0.6;20.8»

Табл. 2 призначена для зберігання типів їжі.

Таблиця 2 – Структура таблиці TABLE_CLASS

Назва поля	Тип	Опис поля
CLASS_ID	INTEGER PRIMARY KEY	Унікальний ідентифікатор типу їжі
CLASS_NAME	TEXT	Найменування типу їжі

Табл. 3 призначена для зберігання даних користувача.

Таблиця 3 – Структура таблиці TABLE_VES

Назва поля	Тип	Опис поля
VES_ID	INTEGER PRIMARY KEY	Унікальний ідентифікатор даних користувача

VES_DATE	TEXT	Дата внесення даних
VES_REKOMENDUEMUY	integer	Результат розрахунків ідеальної ваги
VES_TEKUSHIY	integer	Вага користувача
VES_KKAL	TEXT	Зберігає результат розрахунків денної норми калорій у вигляді «minkal;maxkal»

Табл. 4 призначена для зберігання даних про споживану користувачем їжу

Таблиця 4 – Структура таблиці TABLE_RATION

Назва поля	Тип	Опис поля
RATION_ID	INTEGER PRIMARY KEY	Унікальний ідентифікатор даних користувача
RATION_DATE	TEXT	Дата внесення даних
RATION_DAYLYNORM	integer	Зберігає VES_ID який був останнім у талиці TABLE_VES
RATION_OPS	TEXT	Зберігає дані внесеної користувачем їжі, у вигляді «кількість їжі(шт)# EDA_ID:кількість їжі(грам);». Наприклад - 2#1:50;14:150;

3.3 Проектування структури додатку

На рис. 8 представлена діаграма класів

Додаток складається з 3 класів і 3 абстрактних класів. При запуску додатку користувач потрапляє на форму класу «Менеджер рациона» і має змогу перейти до форми розрахунку ідеальної ваги, якій відповідає клас «Расчет идеального веса». Після підрахунку ідеальної ваги користувач переходить до форми виводу оброблених даних, якій відповідає клас «Идеальный вес». Після отримання даних, користувач має змогу, через форму класу «Менеджер рациона», розрахувати кількість калорій у вибраній їм їжі (рис 8).

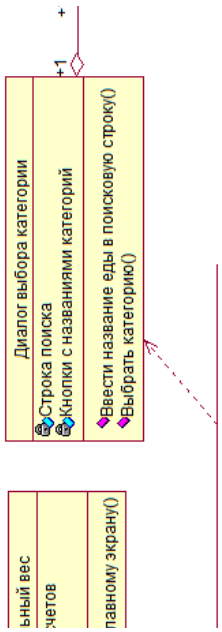

Рисунок 8 – Діаграма класів додатку «Менеджер рациона»

4 РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

У ході проведення дипломної роботи, було створено додаток, що включає в себе головний екран, екрани вводу-виводу розрахунків ідеальної ваги, діалогові вікна та підтримку бази даних SQLite для зберігання їжі та даних користувача.

4.1 Реалізація екранів мовою XML

На рис. 9 представлено головний екран додатку «Менеджер рациона».

Головний екран додатку містить текстові поля виводу інформації, кнопки доданої користувачем їжі та кнопку додавання їжі. Також на головному екрані є можливість відкрити головне меню додатку (рис. 10).

Таким чином головний екран складається з 4 xml-документів:

– activity_main_menu.xml – відповідає за вигляд меню;
– activity_main_menu_drawer.xml – відповідає за елементи меню;
– app_bar_main_menu.xml – відповідає за «верхній шар» ;
– content_main_menu.xml – відповідає за елементи на головному екрані;
– nav_header_main_menu.xml – відповідає за вигляд «шапки» меню;

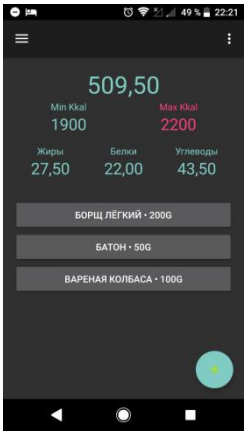

Рисунок 9 – Головний екран додатку «Менеджер раціона»

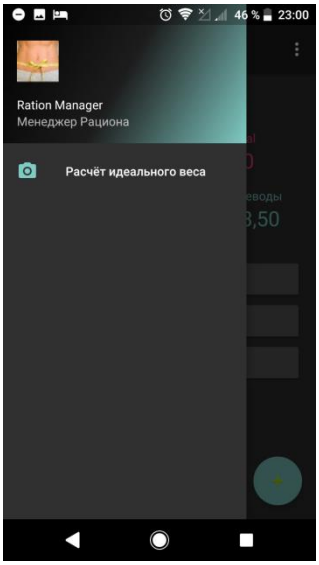

Рисунок 10 – Головне меню додатку «Менеджер рациона»

Розглянемо детальніше content_main_menu.xml.

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentStart="true"
        android:orientation="vertical"
        android:visibility="visible">
        <ScrollView...>
    </LinearLayout>
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_marginTop="30dp"
        android:orientation="vertical">
        <ScrollView
            android:layout_width="match_parent"
            android:layout_height="match_parent">
            <LinearLayout
                android:id="@+id/Picipic"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:orientation="vertical">
            </LinearLayout>
        </ScrollView>
    </LinearLayout>
</LinearLayout>
```

Головний екран складається з 3 LinearLayout, які забезпечують вертикальне розташування об’єктів у них, що зазначено у строці android:orientation = “vertical”. Всередині першого LinearLayout, знаходятся два інших. Всередені обох, розташовані елементи ScrollView, які забезпечують «листання» екрану по вертикалі. Другий ScrollView необхідний для подальшого додавання у нього кнопок. Всередині першого ScrollView розташований ще один вертикальний LinearLayout у якому, в свою чергу, розташовані TextView та чотири горизонтальних LinearLayout. Завдяки використанню різних LinearLayout, можна досягнути розділення робочого екрану на «таблицю».

На відміну від TableLayout, така взаємодія LinearLayout, мені здається більш гнучкою та зручною у використанні.

```
<ScrollView
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical">
        <TextView
            android:id="@+id/ViewKkal"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="top|center_horizontal"
            android:clickable="true"
            android:gravity="top|center"
            android:onClick="Showmore"
            android:text="0"
            andro-
id:textColor="@color/accent_material_dark_1"
            android:textSize="36sp" />
        <LinearLayout...>
        <LinearLayout...>
        <LinearLayout...>
        <LinearLayout...>
    </LinearLayout>
</ScrollView>
```

У елементі TextView, який відповідає за відображення суми калорій, є 2 елементи відповідних за розташування – android:layout_gravity, та android:gravity, де перший відповідає за розташування тексту відповідно до самого елемента TextView, а другий за розташування TextView, відповідно до LinearLayout. У першій строчці, ми бачимо присвоєння цьому TextView власного id, завдяки якому ми можемо працювати саме з цим TextView. Також даний елементі TextView, має свій метод onClick який знаходиться у класі MainMenu і робить видимим наступні два горизонтальних LinearLayout. Вони потрібні для зображення мінімуму та максимуму калорій необхідних користувачу. Нижче знаходяться два горизонтальні LinearLayout які відповідають за зображення денної кількості вжитих жирів, білків, вуглеводів.

Кнопка з «+» знаходиться у app_bar_main_menu.xml і називається FloatingActionButton, або fab скорочено. Він відповідає за визов діалогових ві-

кон з вибором їжі та її кількості. Для діалогу використовуються activity_spisok_1.xml та activity_spisok_2.xml. Перший має лише ScrollView, та вертикальний LinearLayout всередині нього. activity_spisok_2.xml у свою чергу виглядає таким чином (рис 11):

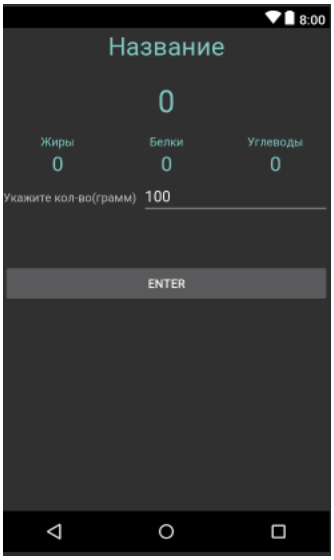

Рисунок 11 – Макет activity_spisok_2.xml

При виборі через діалогові вікна їжі верхній TextView змінює текст на назву вибраної їжі. Наступні 3 строки поводять себе так як і на головному екрані. При вказанні у EditText деякої кількості грамів відображаються змінені характеристики, розраховані для заданої кількості вибраної їжі. Знизу знаходиться кнопка підтвердження вибору кількості їжі, яка закриває усі діалогові вікна і додає їжу та її кількість до бази даних.

При переході користувачем до розрахунку ідеальної ваги він побачить вікно для введення своїх даних(рис 12):

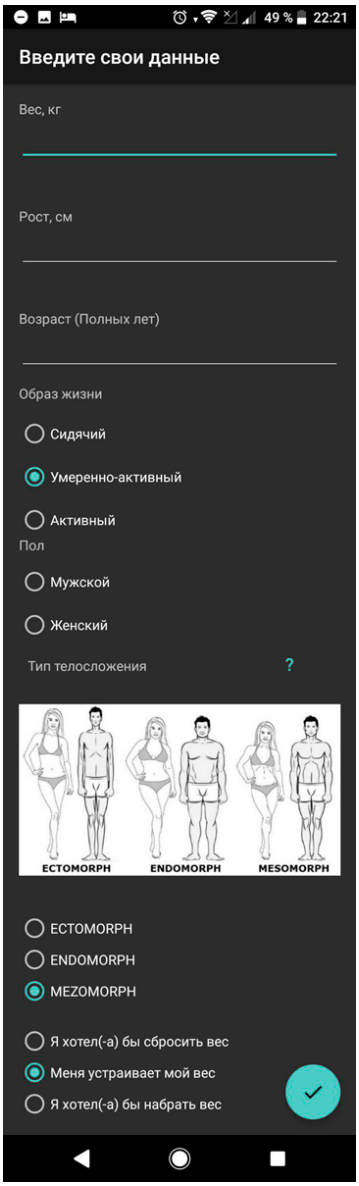

Рисунок 12– Розвернутий вигляд екрану введення даних для розрахунку ідеальної ваги

Екран введення даних складається з 2 XML файлів:

– activity_vvod.xml – відповідає за «верхній шар»;
– content_vvod.xml – відповідає за елементи на екрані;

Розглянемо content_vvod.xml.

```
<ScrollView
    android:layout_width="match_parent"
```

```
        android:layout_height="match_parent"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true">
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:orientation="vertical"
            android:layout_weight="1">
            <LinearLayout...>
            <TextView...>
            <EditText...>
            <TextView...>
            <EditText... >
            <TextView... >
            <EditText... >
            <TextView... >
            <RadioGroup...>
            <RadioGroup... >
            <LinearLayout...>
            <ImageView...>
            <RadioGroup...>
            <RadioGroup...>
        </LinearLayout>
    </ScrollView>
```

Усі елементи цього екрану знаходяться у ScrollView та LinearLayout, який забезпечує форму цього екрану, як список. Всередині знаходяться також декілька елементів RadioGroup які містять декілька елементів RadioButton, для вибору одного з декількох варіантів відповіді.

Після введення усіх даних користувач натискає на fab яких знаходиться у activity_vvod.xml і переходить до екрану виводу даних(рис 13).

Рисунок 13 – Екран виводу даних розрахунків ідеальної ваги

Великими синіми цифрами, через стрілку, відображені попередня ідеальна вага та теперішня. Нижче відображені минула та теперішня вага, денний мінімум та максимум калорій рекомендованих до споживання. Fab зберігає дані та відкриває головний екран.

Складається екран виводу даних з :

– activity_vuvod.xml – відповідає за «верхній шар»
– content_vuvod.xml – відповідає за елементи на екрані

Розглянемо детальніше content_vuvod.xml. На відміну від попередніх екранів, на цьому застосовується саме TableLayout. Складається цей TableLayout з пяти TableRow – табличних строк, всередині яких знаходяться декілька TextView.

```
<TableLayout
    android:layout_width="match_parent"
```

```
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true">
        <TableRow
            android:layout_width="match_parent"
            android:layout_height="match_parent">
            <TextView
                android:id="@+id/textView8"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_gravity="center_horizontal"
                android:layout_weight="1"
                android:gravity="center_horizontal"
                android:text="Идеальный вес"
                android:textAlignment="center"
                android:textSize="30sp"
                android:textStyle="normal|bold" />
        </TableRow>
        <TableRow...>
        <TableRow...>
        <TableRow...>
        <TableRow...>
    </TableLayout>
```

На відміну від використання вертикальних та горизонтальних LinearLayout, в одну строку таблиці можливо вставити декілька елементів і вони будуть займати місце в залежності від своєї «ваги». За це відповідає параметр android:layout_weight. Таким чином більший елемент, якщо він має більшу вагу, «задавить» інші елементи розширившись залежно від своїх параметрів android:layout_width та android:layout_height. Ці параметри можуть набувати трьох значень – match_parent, wrap_content та none, де перший це розмір «скільки можна», другий – залежно від необхідної для повної видимості елементу, а третій це зовсім не займати місця.

4.2 Створення бази даних

Для створення бази даних цього додатку, було використано знайдений мною макет допоміжного класу, який забезпечує роботу з базою даних. Називається цей клас DataBaseHelper, і має усього 2 методи – onCreate та onUpdate. Метод onUpdate використовується при спробі підключення до бд більш нової

версії ніж існуюча. Метод onCreate визивається в тому випадку, коли ми намагаємося підключитися до бд якої не існує. Розглянемо його більш детально.

```
@Override
public void onCreate(SQLiteDatabase db) {
    ContentValues contentValues = new ContentValues();

    db.execSQL("CREATE TABLE " + TABLE_EDA + "(" + EDA_ID +
" INTEGER PRIMARY KEY," + EDA_NAME
            + " TEXT," + EDA_CLASS + " integer," + EDA_OPS
+ " TEXT" + ");");

    db.execSQL("CREATE TABLE " + TABLE_CLASS + "(" +
CLASS_ID + " INTEGER PRIMARY KEY," + CLASS_NAME + " TEXT" +
");");

    db.execSQL("CREATE TABLE " + TABLE_VES + "(" + VES_ID +
" INTEGER PRIMARY KEY," + VES_DATE + " TEXT,"
            + VES_REKOMENDUEMUY + " integer," + VES_KKAL +
" TEXT," + VES_TEKUSHIY + " integer" +");");

    db.execSQL("CREATE TABLE " + TABLE_RATION + "(" +
RATION_ID + " INTEGER PRIMARY KEY," + RATION_DATE
            + " TEXT," + RATION_DAYLYNORM + " integer," +
RATION_KKALSUM + " integer," + RATION_OPS + " TEXT" +");");
```

Розглянемо таблиці на прикладі першої. Створюються таблиці за допомогою методу execSQL який всього лише виконує SQL-код який був у нього записаний. У нашому випадку записаний код "CREATE TABLE " + TABLE_EDA + "(" + EDA_ID + " INTEGER PRIMARY KEY," + EDA_NAME + " TEXT," + EDA_CLASS + " integer," + EDA_OPS + " TEXT" + ");" . Після "CREATE TABLE " знаходиться назва таблиці і відкриваються дужки, всередині яких через коми, записуються кожна назва стовпця та його тип. Усього у SQLite є 2 типи – integer та text. Також є можливість об’явити ключове поле – PRIMARY KEY. При створені такої бази даних, дуже важливим є пробіли між назвою стовпця та його типом. Задля того що би мати можливість звертатися до кожного стовпця кожної таблиці усі вони були оголошені як змінні у самому класі DataBaseHelper.

```
        int[]SRATION_ID ={1,2};
        String[]SRATION_DATE ={"31-03-2019", "10-05-2019"};
        Integer[]SRATION_DAYLY={1,1};
        int[]SRATION_SUM={2400,2367};
        String[] SRATION_OPS={"3#1:100;2:150;1:100",
    "3#2:50;2:150;1:250;"};

        int[] STARTCLASS_ID={1,2,3,4,5,6,7,18,19,20};
        String[]
    STARTCLASS_NAME={"Мясо","Овощи","Фрукты","Молочные","Выпечк
    а","Супы","Крупы","Бобовые","Напитки","Соусы"};
```

Для заповнення таблиць начальними даними використовуються масиви інформації які за допомогою циклів та contentValues добавляються у нову бд в методі onCreate.

```
 for (int i = 0; i < STARTEDA_ID.length; i++ ){
    contentValues.clear();
    contentValues.put(EDA_ID,STARTEDA_ID[i]);
    contentValues.put(EDA_NAME,STARTEDA_NAME[i]);
    contentValues.put(EDA_CLASS,STARTEDA_CLASS[i]);
    contentValues.put(EDA_OPS,STARTEDA_OPS[i]);
    db.insert(TABLE_EDA, null, contentValues);}

for (int i = 0; i < SRATION_ID.length; i++ ){
    contentValues.clear();
    contentValues.put(RATION_ID,SRATION_ID[i]);
    contentValues.put(RATION_DATE,SRATION_DATE[i]);
    contentValues.put(RATION_DAYLYNORM,SRATION_DAYLY[i]);
    contentValues.put(RATION_KKALSUM,SRATION_SUM[i]);
    contentValues.put(RATION_OPS,SRATION_OPS[i]);
db.insert(TABLE_RATION, null, contentValues);}

for (int i = 0; i < STARTCLASS_ID.length; i++ ){
    contentValues.clear();
    contentValues.put(CLASS_ID,STARTCLASS_ID[i]);
    contentValues.put(CLASS_NAME,STARTCLASS_NAME[i]);
    db.insert(TABLE_CLASS, null, contentValues);}
```

Розглянемо більш детальніше таблицю TABLE_RATION, а саме інформацію, що наступає до стовпця RATION_OPS. Як ми бачимо до цього стовпця наступає деякий набір символів. Це внутрішнє кодування даних цього проекту. Для більшої місткості, і оптимізації швидкості роботи моєї бази даних, мною було прийнято рішення про впровадження внутрішнього кодування деяких сто-

впців бази даних. Зроблено це було методом об’єднання строк, і їх подальшого розділення через задані дільники. Таким чином одна клітина таблиці TABLE_RATION, містить у собі дані про усю вжиту протягом дня їжу, її кількість та те скільки усього було вжито різної їжі(для більш зручного декодування). Наприклад, якщо у клітині міститься наступне – « 3#1:100;2:150;1:100; », це дешифрується як 3 різні страви:

- страва з id 1, у кількості 100 грам;
- страва з id 2, у кількості 150 грам;
- страва з id 1, у кількості 100 грам;

Таку систему кодування можливо розширити для збору, наприклад, інформації про час прийняття їжі, для подальшого об’єднання її по групам прийому їжі і аналітики кількості засвоєних жирів, білків та вуглеводів.

4.3 Написання коду Java

Розглянемо клас MainMenu, а саме набір методів який забезпечує роботу бази даних.

4.3.1 Метод DeshifrEDAOPS

Цей метод призначений для декодування кількості білків, жирів, вуглеводів(БЖВ), та розрахунку кількості калорій у їжі.

Кодування даних їжі представлено записом її БЖВ через «;» .

```
public static float[] DeshifrEDAOPS(Integer gram, Integer
idEDU){
```

У цій строчці ми можемо побачити що цей клас приймає дві вхідних характеристики integer – gram та idEDU, а на виході віддає масив типу Float.

```
float [] KalJiruBelkiUglev={0,0,0,0};
String [] JiruBelkiUglev={};
```

Потрібні нам у майбутньому масиви.

```
SQLiteDatabase database1 =
dataBaseHelper.getWritableDatabase();
final Cursor cursor1 =
database1.query(DataBaseHelper.TABLE_EDA, null, null, null,
null, null, null);
cursor1.moveToFirst();
```

Підключення до класу dataBaseHelper та отримання права на запис до бд. Створення курсору який буте пересуватися по таблиці і передвигання його на першу позицію. Весь подальший код знаходится у циклі do – while, де умова while cursor1.moveToNext().

```
do{
    int idIndex =
cursor1.getColumnIndex(DataBaseHelper.EDA_ID);
    int OPSIndex =
cursor1.getColumnIndex(DataBaseHelper.EDA_OPS);
```

Ми отримуємо індекс стовпців id та ops.

```
if (idEDU==cursor1.getInt(idIndex)){
   String EDAOPS = cursor1.getString(OPSIndex);
```

Якщо вхідний id у методі збігся з з id їжі у базі даних то ми отримуємо характеристики цієї їжі.

```
String delimeter = ";";
String Jiru="";
 String Belki="";
 String Uglev="";
```

Декілька потрібних нам стрінгів.

```
JiruBelkiUglev = EDAOPS.split(delimeter);
Jiru = JiruBelkiUglev[0];
Belki = JiruBelkiUglev[1];
Uglev = JiruBelkiUglev[2];
```

За допомогою методу split ми ділимо отриману характеристику на БЖВ, за допомогою delimeter, який виконує роль роздільного знаку, та записуємо отриманий результат у масив JiruBelkiUglev. Присвоюємо кожній характеристиці віддільну змінну.

```
        Float Persent = ((float)(gram)/100);
Float OUTJiru = (Float.valueOf(Jiru))*Persent;
Float OUTBelki = Float.valueOf(Belki)*Persent;
Float OUTUglev = Float.valueOf(Uglev)*Persent;
```

Вираховуємо процент кількості грам. Так як у бд зберігається БЖД на 100 грамів їжі, при діленні на це число вхідної кількості грамів ми отримуємо множник, при помножені на який ми отримуємо кількість БЖД для заданої користувачем кількості грамів

```
Float Kal = (OUTJiru*9)+(OUTBelki*4)+(OUTUglev*4);
             KalJiruBelkiUglev[0]=Kal;
             KalJiruBelkiUglev[1]=OUTJiru;
             KalJiruBelkiUglev[2]=OUTBelki;
             KalJiruBelkiUglev[3]=OUTUglev;
          }
     } while (cursor1.moveToNext());
return KalJiruBelkiUglev;}
```

Розрахунок кількості калорій – це помноження кількості жирів на 9, білків та вуглеводів на 4, та їх сумування. Усі отримані дані, додаються до масиву, який буде повертатися після виклику цього методу.

4.3.2 Метод DeshifrMAINOPS

```
public float[] DeshifrMAINOPS() {
        float []mass={0,0,0,0};
    Integer A = 0;
    //float[] mass = new float[0];
    try {
```

```
        SQLiteDatabase database1 =
dataBaseHelper.getWritableDatabase();
        final Cursor cursorRAW = database1.rawQuery("SELECT
" + DataBaseHelper.RATION_DATE + "," +
DataBaseHelper.RATION_OPS + " FROM " +
DataBaseHelper.TABLE_RATION, null);
        cursorRAW.moveToLast();
```

Є два типу sql запитів для отримання даних необхідних курсору – Query та rawQuery. Перший, це приведений до деякого виду запит, в якому кожний з параметрів, які пишуться через коми, відповідає за складення самого SQL-запиту. На відміну від Query, rawQuery це запит, який представлений чистою мовою SQL-запитів.

```
int DATEIndex2 = cursorRAW.getColumnIndex(RATION_DATE);
int OPSIndex1 = cursorRAW.getColumnIndex(RATION_OPS);
String StrDate = cursorRAW.getString(DATEIndex2);
A = DateCheck(StrDate);
```

Після отримання даних з бази ми використовуємо метод DateCheck, який повертає індекс в залежності від того, чи є сьогоднішній день в базі даних.

```
List<Float> myList = new ArrayList<>();
myList.add(1f);myList.add(2f);myList.add(3f);myList.add(4f);
if (A == 1) {

    try{
    String RATIONOPS = cursorRAW.getString(OPSIndex1);
    String delimeter0 = "#";
    String delimeter1 = ";";
    String delimeter2 = ":";
    Float SumofKkal = 0f;     Float SumofJiru = 0f;
    Float SumofBelki = 0f;    Float SumofUglev = 0f;
    Float GetKkal = 0f;    Float GetJiru = 0f;
    Float GetBelki = 0f;    Float GetUglev = 0f;
```

Ми створюємо List для додавання безлічі елементів до нього. Масив не підходить, бо після додавання елементу до нього, він отримує свій розмір, який є незмінним. Перші чотири позиції заповнюються для подальшого їх використання. Якщо метод DateCheck повертає одиницю, то сьогоднішній день є у базі даних. Для декодування інформації, використовується цілий набір дільників.

```
 String[] j = RATIONOPS.split(delimeter0);
if (Integer.parseInt(j[0])==0){}else {
String[] Prod = j[1].split(delimeter1);
for (int i = 0; i < (Integer.parseInt(j[0])); i++) {
    String[] idkol = Prod[i].split(delimeter2);
    mass[0] = Float.valueOf(idkol[1]);
    mass[1] = Float.valueOf(idkol[0]) ;
float[] GettomainEdaOps = DeshifrE-
DAOPS((Integer.parseInt(idkol[1])), (Integ-
er.parseInt(idkol[0])));
```

За допомогою split розділяється RATIONOPS який ми отримали з бази даних. Різні частини, відділені спеціальним символом – дільником «#». Усі отримані частини додаються у масив j, де нульовий елемент масиву – це кількість їжі яку додав користувач, а перший елемент, це закодована інформація щодо цієї їжі. Після перевірки на наявність доданої користувачем їжі, ми використовуємо наступний дільник на перший елемент масиву j – «;». Він відділяє кожну додану страву і додає до масиву Prod. Кожний елемент отриманого масиву розділяється через дільник «:», у циклі, який бере кількість повторень з нульового елементу масиву j. Отримані дані, а саме id їжі, та її кількість, передаються методу DeshifrEDAOPS, який був описаний раніше.

```
GetKkal = GettomainEdaOps[0];
    GetJiru = GettomainEdaOps[1];
    GetBelki = GettomainEdaOps[2];
    GetUglev = GettomainEdaOps[3];
    myList.add(GetKkal);     myList.add(GetJiru);
    myList.add(GetBelki);     myList.add(GetUglev);
    SumofKkal = SumofKkal + GetKkal;
    SumofJiru = SumofJiru + GetJiru;
    SumofBelki = SumofBelki + GetBelki;
    SumofUglev = SumofUglev + GetUglev;}
```

Отримані дані додаються до myList, в якому вже заняті перші чотири місця. Також отримані дані додаються до суми цих даних. На цьому закінчується процес декодировки.

```
 mass = new float[myList.size()];
for (int i = 0; i < myList.size(); i++) mass[i] = myL-
ist.get(i);
mass[0] =  SumofKkal ;
mass[1] =  SumofJiru ;
```

```
mass[2] =  SumofBelki;
mass[3] =  SumofUglev;
}}  catch (ArrayIndexOutOfBoundsException e) {
Toast.makeText(this, "Ошибка в дешифровке",
Toast.LENGTH_SHORT).show();            }}} catch (CursorIn-
dexOutOfBoundsException e) {
        if (A != 1 && A != 2) {
            Toast.makeText(MainMenu.this, "Nichego to tut i
net", Toast.LENGTH_SHORT).show();} else {
            Toast.makeText(MainMenu.this, " oshibka",
Toast.LENGTH_SHORT).show();}
    return mass;}
```

Отриманий List дає свій розмір новому масиву і передає йому усі свої елементи. На перші чотири позиції додаються суми калорій та БЖВ. Повертає метод саме цей масив.

4.3.3 Метод ShifrMAINOPS

Цей метод кодує отриману інформацію від користувача. А саме вибрану користувачем їжу, та її кількість. При введені інформації користувач може бачити скільки містить у собі та чи інша кількість вибраного продукту.

```
public void ShifrMAINOPS(int id,int grammget) {
    Integer A = 0;
    //float[] mass = new float[0];
    try {
        SQLiteDatabase database1 = dataBaseHel-
per.getWritableDatabase();
        final Cursor cursorRAW = database1.rawQuery("SELECT
" + DataBaseHelper.RATION_DATE + "," + DataBaseHel-
per.RATION_OPS + " FROM " + DataBaseHelper.TABLE_RATION,
null);
        cursorRAW.moveToLast();
        int DATEIndex2 = cursor-
RAW.getColumnIndex(RATION_DATE);
        int OPSIndex1 = cursor-
RAW.getColumnIndex(RATION_OPS);
        String StrDate = cursorRAW.getString(DATEIndex2);
        A = DateCheck(StrDate);
```

Метод ShifrMAINOPS отримує на вхід id їжі, та її кількість. За допомогою методу rawQuery робиться SQL-запит для отримання стовпців з датою та даними з таблиці TABLE_RATION. Після створення курсору він переноситься на

останній рядок, для отримання останніх даних. Достаємо дані і переводимо в формат String. За допомогою методу DateCheck, дізнаємося чи зроблений останній запис до таблиці сьогодні.

```
  if (A == 1) {

    try{
        String RATIONOPS = cursorRAW.getString(OPSIndex1);
        String delimeter0 = "#";

        String MAINOPS;
        //   Integer range = 4;
        String[] j = RATIONOPS.split(delimeter0);
       if(Integer.parseInt(j[0])==0) {
          j[0]=String.valueOf(Integer.parseInt(j[0])+1);
//0->1 ; 1->2
          String Zero=String.valueOf(id)+":"
            +String.valueOf(grammget)+";";
          MAINOPS=j[0]+"#"+Zero;
```

Якщо останній запис був зроблений не сьогодні, то виконується метод OnNewDay, який перевіряє ще раз число, і якщо сьогоднішнього запису немає, то додає новий за даними виду «0#». Метод OnNewDay, зазвичай, виконується в OnCreate класу MainMenu. Таким чином якщо користувач користується додатком в період з 11:59 по 0:01, дані будуть додані до нового дня. Якщо запис вже є, тоді дані діляться за допомогою вже знайомих дільників. Після ділення за допомогою «#», нульовий елемент масиву j переводиться в Integer, і якщо це перший введений користувачем продукт, то до нього додається одиниця. Після цього отримані на вході методу дані кодуються, в вигляді - id їжі+«:»+кількість їжі у грамах+«;». Так як усі дані – стоки, їх сумування призводить до об'єднання строк. Після цього ми сумуємо отриманий нульовий елемент, решітку та отриманий код.

```
}else {
    j[0] = String.valueOf(Integer.parseInt(j[0]) + 1);//
//0->1 ; 1->2
    j[1] = j[1] + String.valueOf(id) + ":"
            + String.valueOf(grammget) + ";";
    MAINOPS = j[0] + "#" + j[1];
}
```

Якщо це вже не перша їжа додана сьогодні користувачем, тоді після ділення за допомогою «#», нульовий елемент масиву j переводиться в Integer, і до нього додається одиниця. Після цього отримані на вході методу дані кодуються і додаються до вже існуючого коду, в вигляді – перший елемент масиву j+id їжі+«:»+кількість їжі у грамах+«;». Так як усі дані – стоки, їх сумування призводить до об'єднання строк. Після цього ми сумуємо отриманий нульовий елемент, решітку та перший елемент масиву j.

```
Date date = Calendar.getInstance().getTime();
 SimpleDateFormat sdf = new SimpleDateFormat("dd-MM-yyyy");
String dateString = sdf.format(date);
 DateFormat formatter;
 Date date2 = null;
 formatter = new SimpleDateFormat("dd-MM-yyyy");
 try {
     date2 = formatter.parse(dateString);
 } catch (ParseException e) {
     e.printStackTrace();
 }
```

Отримуємо дату і перетворюємо її у вигляд дата-місяць-рік (12-02-2019).

```
  ContentValues contentValues = new ContentValues();
    contentValues.put(DataBaseHelper.RATION_DAYLYNORM,1);
    contentValues.put(DataBaseHelper.RATION_KKALSUM,0);
    contentValues.put(DataBaseHelper.RATION_OPS, MAINOPS);
 String[]datemass={dateString};
 database1.update(DataBaseHelper.TABLE_RATION,
 contentValues,DataBaseHelper.RATION_DATE+"=?",datemass);
     dataBaseHelper.close();
```

За допомогою contentValues ми додаємо усі необхідні для нового запису у таблиці дані, у відповідні стовпці. За допомогою методу update у таблиці TABLE_RATION знаходиться строчка з сьогоднішнім числом і в цій строчці усі дані змінюються на додані до contentValues. Цей метод нічого не повертає.

ВИСНОВКИ

У дипломній роботі було розглянуто питання, щодо розробки android-додатку для розрахунку калорій. Для успішної реалізації був проведений аналіз предметної області, з якого випливає, що ця тема є дуже поширеною на ринку android-додатків, але більшість з них мають досить складний інтерфейс. Було проведено аналіз існуючих аналогів, після чого були сформульовані вимоги до розробляємої системи, її зовнішньому вигляду, функціональності та практичності.

Було проведено проектування бази даних, розроблена архітектура системи і призначений для користувача інтерфейс, створена програмна реалізація. Ці етапи детально описані в роботі.

Перевагами розробленого android-додатку є:

- простий, зручний в навігації, інтуїтивно зрозумілий користувачу інтерфейс;
- можливість швидкого розрахунку калорійності їжі, та її насиченість білками, жирами та вуглеводами;
- можливість розрахунку «Ідеальної ваги» та денної норми калорій;
- можливість розрахунку поживності усієї їжі вжитої протягом дня ;
- використання бази даних для зберігання інформації і вживання системи управління базами даних SQLite для маніпулювання даними;

Для вдалого впровадження додатку і тривалої подальшої експлуатації важливо правильно організована робота по забезпеченню збільшення додатку інноваційними технологіями, яка приведе до збільшення користувачів та людей, зацікавлених у вкладанні коштів, для розвитку даного android-додатку. Даний додаток може бути розміщений у Play Market.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Раціональне харчування: закони та принципи. URL: https://pidruchniki.com/85492/bzhd/ratsionalne_harchuvannya_zakoni_printsip (дата звернення 12.02.2019).
2. Android-додаток «Калькулятор калорий ХиКи». URL: https://play.google.com/store/apps/details?id=ru.hikisoft.calories (дата звернення 26.03.2019).
3. Android-додаток «Худеем вместе. Дневник калорий»». URL: https://play.google.com/store/apps/details?id=ru.hudeem.adg (дата звернення 26.03.2019).
4. О.М. Ткаченко, В.А. Каплун. Об’єктно-орієнтоване програмування мовою Java/ О.М. Ткаченко, В.А. Каплун: Вінниця ВНТУ. 37 с.
5. П. Дейтел, Х. Дейтел, А. Уолд. Android для разработчиков. 3-е издание, 2016 / П. Дейтел, Х. Дейтел, А. Уолд: Питер. 14 с.
6. Сайт «Освой программирование играючи. Сайт Александра Климова» URL: http://developer.alexanderklimov.ru/android (дата звернення 28.03.2019).