ДИПЛОМНА РОБОТА

ДР.ПІс – 10.00.000 ПЗ

Група ПІс-2015

Курліковський С. В.

2019

ПВНЗ Університет Короля Данила

Кафедра Інформаційних технологій та програмної інженерії

УДК 004

ДИПЛОМНА РОБОТА

Тема ***Розробка програми мобільного застосунку з надання фітнес послуг засобами Android Java***

Напрям підготовки ***6.050103 – «Програмна інженерія»***
(код і назва спеціальності)

ПОЯСНЮВАЛЬНА ЗАПИСКА

ДР.ПІс – 10.00.000 ПЗ
(позначення)

Студент

Курліковський С. В.
(підпис) (дата) (розшифрування підпису)

Керівник проекту
к.т.н. *Ващишак С. П.*
(посада) (підпис) (дата) (розшифрування підпису)

Нормоконтроль
к.т.н *Мануляк І. З.*
(посада) (підпис) (дата) (розшифрування підпису)

Допускається до захисту
Завідувач кафедри
д.т.н.,доц. *Мельничук С. І.*
(посада) (підпис) (дата) (розшифрування підпису)

2019

ПВНЗ УНІВЕРСИТЕТ КОРОЛЯ ДАНИЛА

Факультет *Інформаційних технологій*
Кафедра *Інформаційних технологій та програмної інженерії*
Напрям підготовки *6.050103 – «Програмна інженерія»*

ЗАТВЕРДЖУЮ:
Завідувач кафедри ІТПІ
_____ *С.І. Мельничук*
"____" _____ 2019 р.

ЗАВДАННЯ
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ)

Студенту *Курліковському Сергію Володимировичу*

1. Тема проекту (роботи) *Розробка програми мобільного застосунку з надання фітнес послуг засобами Android Java*

Затверджена наказом ректора Університету Короля Данила від *15.11.2018 р. № 20/4*
2. Термін здачі студентом закінченого проекту (роботи) *10.06.2019 р.*
3. Вихідні дані до проекту (роботи) *Java, SQLite, XML*

4. Зміст пояснювальної записки (перелік питань, що їх належить розробити) *1. Аналіз особливостей застосування систем індивідуальних вправ та додатків для їх реалізації. 2. Обгрунтування технологій програмування та розробка структури додатку. 3. Розробка програмного забезпечення застосунку з наданя фітнес-послуг.*

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) *Постановка завдання, вибір мобільної ОС, вибір середовища розробки та мови програмування, структура проекту, створення профілю, створення вправ і записів, метод ІМТ, особливості додатку.*

6. Консультанти з проекту (роботи), із зазначенням розділів проекту, що стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 30.10.2018

Керівник Ващишак С. П.

Завдання прийняв до виконання Курліковський С. В.

КАЛЕНДАРНИЙ ПЛАН

Пор №	Назва етапів дипломного проекту (роботи)	Термін виконання етапів проекту (роботи)	Примітка
1.	Аналіз особливостей застосування систем індивідуальних вправ та додатків для їх реалізації	21.11.2018	
2.	Обгрунтування технологій програмування та розробка структури додатку	03.12.2018	
3.	Розробка програмного забезпечення застосунку з наданя фітнес-послуг	28.01.2019	
4.	Оформлення пояснювальної записки	16.04.2019	
5.	Оформлення графічного матеріалу та підготовка до захисту роботи	10.06.2019	

Студент-дипломник *Курліковський С. В.*
(підпис) (розшифровка підпису)

Керівник проекту *Ващишак С. П.*
(підпис) (розшифровка підпису)

АНОТАЦІЯ

Дипломна робота присвячена розробці мобільного застосунку з надання фітнес послуг. В даній роботі реалізовано конструктор вправ, базу даних, профіль користувача до якого прив’язані записи зроблених вправ та дані змін організму.

Розвиток фітнесу у світі та в Україні невпинно зростає, через його соціальну роль та зв’язок зі здоровим способом життя. Щоб охопити більше коло людей, які хочуть стати здоровішими, потрібно розробляти додаток на найпопулярній платформі Android. Тому головним завданням даної роботи є розроблення конструктора для створення власних вправ та історію записів цих вправ, реалізувати ІМТ для відслідковування прогресу та базу данних, куди будуть зберігатись всі дані.

SUMARRY

Thesis is devoted to developing a mobile application for the provision of fitness services. In this work, a constructor of exercises, a database, a profile of the user, to which the records of the exercises and the changes in the body are attached, are implemented.

The development of fitness in the world and in Ukraine is steadily growing, due to its social role and the connection with a healthy lifestyle. To reach more people who want to become healthier, you need to develop an application on the most popular Android platform. Therefore, the main task of this work is to develop a designer to create their own exercises and history records of these exercises, implement BMI for tracking progress and the database, where all data will be stored.

РЕФЕРАТ

Розрахунково-пояснювальна записка: 107 сторінок, 31 рисунок, 2 таблиці, 4 додатки.

Об’єктом дослідження є процес тренувань за певним спортивним комплексом вправ під певний вік, які допоможуть стати здоровішим і привести тіло до омріяної форми.

Дипломна робота присвячена розробці мобільного застосунку з надання фітнес послуг. **Метою роботи** є розробка додатку, який буде допомагати користувачам покращити самопочуття і здоров’я та привести їх фізичну форму до омріяного результату.

Перший розділ складається з аналізу та пояснення вибору операційної системи і середовище розробки, які застосовувались в дипломному проекті.

Другий розділ містить обгрунтування вибору мови програмування та структуру проекту.

Третій розділ містить опис розробленого додатку, сумісність між версіями android та результати тестування мобільного застосунку.

JAVA, SQLITE, XML, UML, КОРИСТУВАЧ

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ І СКОРОЧЕНЬ

ОС - Операційна система

ОП - Оперативна пам'ять

API - Application programming interface

UI - User interface

GUI - Graphical user interface

IDE - Integrated Development Environment

SDK - Software development kit

ADT - Android development tools

CDT - C development tools

JDT - Java development tools

IDE - Integrated Development Environment

ВСТУП

Актуальність теми. Для людини прагнення досягти максимально можливого комфорту в житті завжди було природною потребою. З розвитком технологій і появою мобільних телефонів та інтернету, вже мало хто може уявити своє життя без можливості перебувати в онлайн-режимі. Саме потреби користувачів і диктують свої правила в створенні нових програмних продуктів. Якщо раніше перші програмні забезпечення для мобільних телефонів - це були вбудовані додатки, необхідні для виконання певних функцій самого пристрою, наприклад, телефонна книга, можливість писати, редагувати смс-повідомлення, то зараз, з появою нових гаджетів, все більшою популярністю користується розробка мобільних додатків для Android і iOS (iPhone). Мобільні девайси, стали невід'ємною частиною життя. Це комунікації між людьми, перегляд інформації, блокнот, прослуховування улюбленої музики, і ще безліч функцій. Мобільний смартфон це копія комп'ютера яку завжди можна мати при собі.

Поява мобільного Інтернету неймовірно полегшила життя багатьом користувачам. Як правило, звичними супутниками подорожей, поїздок стали телефони, планшети та інші пристрої, через які можна просто підключитися до мережі. Це настільки стало звичайною справою, що багато хто навіть не замислюються про те, що все це стало можливо, завдяки спеціальним програмам, без наявності яких навряд чи можна було б досягти необхідної ефективності.

Розвиток фітнесу у світі та в Україні невпинно зростає, через його соціальну роль та зв’язок зі здоровим способом життя. Сьогодні у світовій практиці накопичено чималий досвід застосування різних видів і методик оздоровчого фітнесу, в яких відбиваються результати наукових досліджень проблем рухової активності, і які реалізуються переважно через розробку різноманітних методик і фітнес-програм. Вони спираються на різні види рухової активності – аеробної чи силової спрямованості, оздоровчу гімнастику, фітнес у воді (аквафітнес), рекреативні види рухової активності, засоби психоемоційної

регуляції тощо. Ці методики можуть бути засновані як на одному виді рухової активності (аеробіка, плавання, оздоровчий біг), так і на поєднанні декількох видів (наприклад, оздоровче плавання і біг), причому ті й інші передбачають здоровий спосіб життя (загартування, масаж тощо) як складову оздоровчого процесу. Найбільш розповсюджені та ефективні з точки зору оздоровлення та рекреації методики фітнесу засновані на різних видах рухової активності аеробної спрямованості. Вони включають насамперед вправи, енергозабезпечення яких здійснюється за рахунок використання кисню.

Розробка тієї чи іншої фітнес-програми здійснюється з урахуванням особливостей контингенту тих, на кого вона розрахована (стать, вік, стан здоров'я тощо). Деякі інтегративні методики можуть бути орієнтовані на спеціальні групи населення – дітей, осіб похилого віку, осіб із ризиком певних захворювань і т. п. Розмаїття методик оздоровчого фітнесу свідчить про значний накопичений досвід у цій галузі; воно також пояснюється намаганням задовольнити оздоровчо-рекреаційні інтереси широкого загалу.

Об’єкт дослідження є процес тренувань за певним спортивним комплексом вправ під певний вік, які допоможуть стати здоровішим і привести тіло до омріяної форми.

Задачі досліджень. Відповідно до теми в роботі поставлені такі задачі:

- провести аналіз існуючих аналогів мобільних застосунків з надання фітнес послуг;
- вибір операційної системи, середовища розробки та мови програмування;
- розробити зручний інтерфейс і дизайн, по якому користувачі будуть інтуїтивно пересуватись;
- покращити алгоритм «Індекс Маси Тіла»;

провести повне тестування готового мобільного застосунку.

Методи досліджень. При вирішенні поставленого завдання для розробки якого були використані теорія інформації, математична статистика, а також технології: Java, SQLite, XML в середовищі розробки Android Studio. Проект

розроблений з розрахунком на подальше вдосконалення і додавання нових функціональних можливостей. Тому інтеграція нових функцій займатиме мінімум часу і зусиль.

Результати досліджень. Результатом дипломної роботи є розроблений спортивний мобільний застосунок з надання фітнес послуг, для якого розроблені конструктор створення власних вправ, покращений алгоритм індекса маси тіла, історія записів та бази даних в якій інформація зберігається.

1 АНАЛІЗ ОСОБЛИВОСТЕЙ ЗАСТОСУВАННЯ СИСТЕМ ІНДИВІДУАЛЬНИХ ВПРАВ ТА ДОДАТКІВ ДЛЯ ЇХ РЕАЛІЗАЦІЇ

1.1 Актуальність мобільних пристроїв

Чи варто говорити про те, скільки часу люди проводять зі своїм смартфоном? Практично кожен користувач смартфона заходить в інтернет зразу після пробудження рано і безпосередньо перед сном. Більшість контента поступає саме з екранів смартфона або планшета. В Україні більший відсоток інтернет-трафіку приходиться на мобільні пристрої [1, 2].

І це не дивно, тому що смартфон - це пристрії, який завжди під рукою і не займає багато місця, він дає можливість досить просто получити доступ до потрібної інформації. Смартфони настільки сильно укоренились в наше життя, що ми надовго з ними не розлучаємся. Мобільний смартфон це копія комп'ютера яку завжди можна мати при собі. І їх популярність навіть вища ніж у комп’ютерів, як показано на рис. 1.1.

Мобільні застосунки надають додаткові можливості для самостійного поглиблення в вивченні любої галузі знань як науковцям, викладачам, студентам, так і всім, хто зацікавлений. Основними перевагами використання мобільних застосунків у самоосвітніх цілях є навчання в зручний для користувача час та у зручному місці; отримання необхідної інформації незалежно від часу й місця знаходження; навчання може здійснюватися одночасно з професійною діяльністю або з навчанням за іншим напрямом; можливість почати навчання з будь-яких питань залежно від рівня підготовки. Варто відмітити один з найвагоміших аспектів, який свідчить на користь використання мобільних застосунків у сфері самоосвіти – це забезпечення соціальної рівності: рівні можливості одержання доступу до освітніх та наукових ресурсів незалежно від місця проживання, стану здоров’я та соціального статусу.

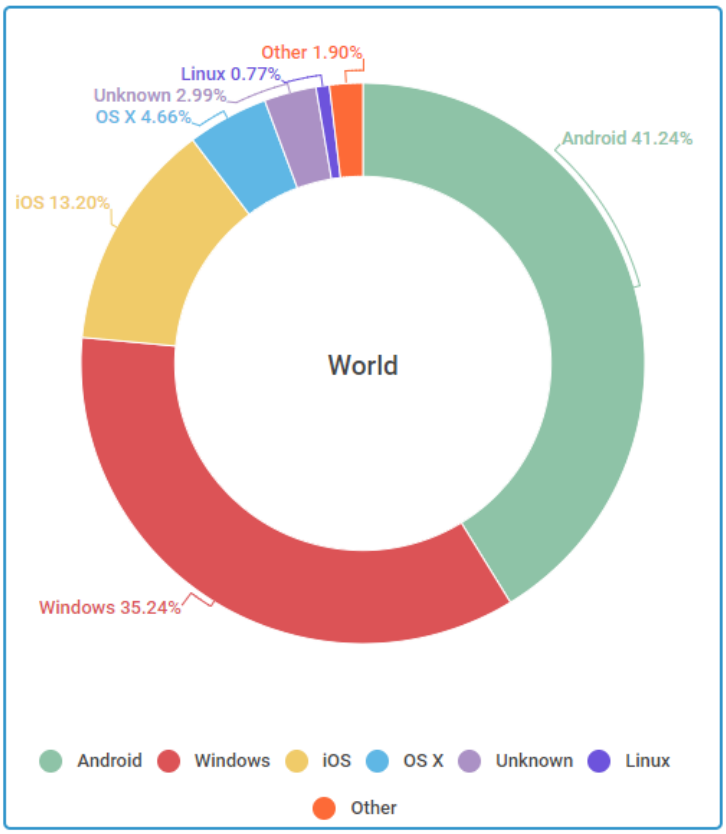


Рисунок 1.1 – Діаграма співвідношення ОС на ринку

Підвівши підсумок вище сказаному можна сказати, що смартфон – це самий популярний цифровий комунікативний пристрій. Отже розробка саме мобільного додатка, являється правильним вибором.

1.2 Аналіз предметної області та додатків - аналогів

Популярність фітнесу на сьогоднішній день незаперечна. Багато чоловіків і жінок за допомогою цього виду спорту вирішують масу проблем зі здоров'ям. Регулярні тренування – це необхідний запас бадьорості і сил, який так необхідний організму. Перевага фітнесу в тому, що він доступний людям абсолютно будь-якого віку, тут немає яких-небудь особливих вимог до статурі або ваговій категорії. Сучасна аеробіка – це цілий комплекс різних елементів, що

поєднують в собі гімнастику і танці. Регулярні тренування – це шанс повністю оздоровити організм і забути про багатьох хворобах [3].

Перерахуємо найбільш популярні з них:

- pump – в цьому стилі основний елемент тренування – це присідання, нахили і жими разом з невеликою штангою. Основне навантаження припадає на ноги, спину і живіт. Таким чином, досягається максимальна ефективність при позбавленні від жиру на проблемних зонах;
- fank більшою мірою відноситься до танцювального напрямку. У тренуванні використовуються елементи молодіжних танців. За основу беруться такі популярні напрямки, як брейк або хіп-хоп. Регулярні заняття Fank-аеробікою дозволяють позбавитися від зайвої ваги, розвинути пластику і гнучкість тіла;
- fittball – оригінальна аеробіка, в якій основний елемент - надувний м'яч. На таких тренажерах найкраще розтягуватися і зміцнювати м'язи ніг. Регулярні тренування дозволяють людині поліпшити поставу і добре розвинути гнучкість;
- step-аеробіка особливу популярність набула в останні роки. Особливість даного напрямку – упор на підйоми і спуски під час тренування, максимальне навантаження м'язів ніг. Основний «помічник» – спеціальна степ-платформа, яка може мати різну висоту, залежно від рівня підготовки спортсмена. Тренування проводяться під ритмічні мотиви під керівництвом професійного тренера [4];
- cycle – сучасний вид аеробіки, який з'явився зовсім нещодавно. Тут робота над тілом не обходиться без застосування велосипедів та імітації різного виду їзди. Якісні тренування дозволяють добре розвинути серцево-судинну систему, вигнати зайві калорії з організму і навантажити м'язи ніг;
- аквааеробіка – це вид фітнесу, який проводиться виключно у воді. Зміст вправ – розвиток м'язів за рахунок подолання опору водної маси під час виконання різних рухів. Перевага таких вправ – мінімум травматизму за

рахунок того, що вага людини у воді суттєво зменшується. При цьому аквааеробіка може проводитися на мілині або глибині. Такі тренування рекомендуються повним людям і вагітним [5];

- тайбо – це «бойовий» вид фітнесу, в основі якого лежать різні елементи східних єдиноборств в суміші з силовими вправами. Під час тренування виконуються удари руками і ногами, здійснюються різні рухи в бік і вниз. Загалом, задіяні всі м'язові групи, що дозволяє говорити про максимальної ефективності тренування;
- flex – аеробіка, в якій застосовуються елементи йоги. При цьому всі рухи здійснюються не під динамічну, а під спокійну, розслаблюючу музику. Найчастіше запалюються арома-свічки, створюється затишна атмосфера комфорту. З допомогою таких занять можна добре розслабити всі м'язи тіла, розтягнути їх і розвинути неймовірну гнучкість.

При аналізі ринку на предмет наявності подібних додатків були знайдені такі додатки, як:

- Fitness - Fit Woman 2019 (рис. 1.2). Додаток, який орієнтований тільки для жіночої статі;

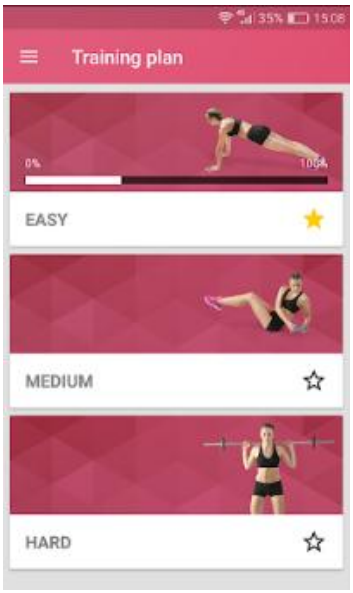


Рисунок 1.2 – Головна сторінка додатку Fitness-Fit Woman 2019

- Total Fitness - Gym & Workouts (рис. 1.3). Фітнес-керівництво для любителів цього виду спорту;

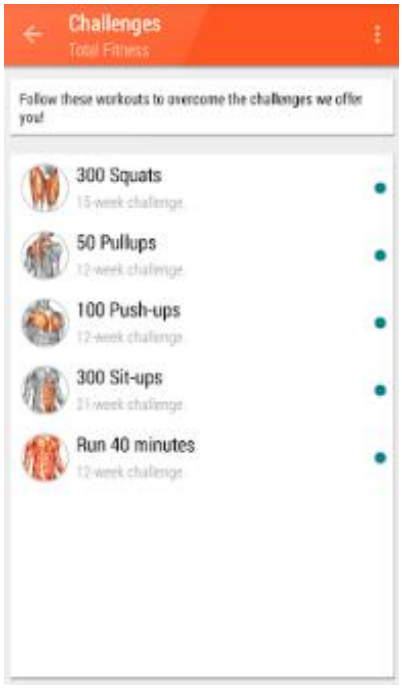


Рисунок 1.3 – Головна сторінка додатку Total Fitness-Gym & Workouts

- Fitness Tracker (рис. 1.4). Відслідковувати прогрес тренувань.

Рисунок 1.4 – Головна сторінка додатку Fitness Tracker

Основні недоліки додатків-аналогів наведено в таблиці 1.1.

Таблиця 1.1 – Основні недоліки додатків-аналогів

аналоги / недоліки	Fitness Tracker	Total Fitness-Gym & Workouts	Fitness-Fit Woman 2019
Погано оптимізований	√		√
Статичний список вправ	√	√	√
Не зручний інтерфейс	√		

1.3 Вибір операційної системи

На даний момент існує багато різноманітних мобільних операційних систем [6]. Але всі вони мають різну популярність. Якщо додаток не кроссплатформовий, то розробка для свідомо вузького кола користувачів може заздалегідь приректи його на провал. Адже чим більше користувачів опробує продукт, тим більше залишиться зацікавленим в ньому. Тому одна із головних задач – це вибір ОС, під яку буде розроблятись додаток.

На рис. 1.5 представлена діаграма, що показує частки, які займають різні ОС на ринку смартфонів.

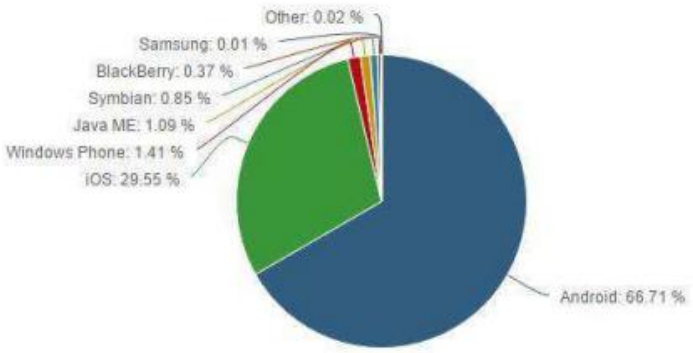


Рисунок 1.5 – Діаграма співвідношення ОС на ринку смартфонів

Згідно статистики від компанії NetMarketShare, особистому досвіду і даних діаграм представлених сайтом [7], були виділені три найбільш популярні ОС, які необхідно розглянути (в порядку збільшення популярності):

- Windows Phone
- iOS
- Android

Загальна інформація про світові продажі смартфонів по типах ОС наведено в таблиці 1.2.

Таблиця 1.2 – Світові продажі смартфонів по типах ОС, (3,4-й квартал 2018 року)

ОС	Продаж в 2018, кв. III-IV, тис. в шт.	% Ринку
Android	143 312	71, 53
iOS	32 741	16, 34
Windows Phone	6 506	3, 25
Bada/Linux	2 701	1, 35
Symbian	5 917	2, 96
BlackBarry	4 477	2, 23
Інші	4 693	2, 34
Загалом	200 347	100

1.3.1 Аналіз операційної системи Windows Phone

Windows Phone – мобільна ОС, розроблена американською компанією Microsoft. Вона встановлена на 1,41% смартфонів. Через невелику частину ринку і неясні перспективи число додатків для Windows Phone помітно поступається кількістю додатків для Android і iOS. Однак, магазин додатків Windows Phone Store може задовольнити практично любі потреби, оскільки кількість додатків в ньому перевищує 300 тисяч.

Проте, враховуючи щорічне падіння популярності, розробка додатку під дану ОС не є привабливою перспективою, оскільки охоплення користувачів нею буде вкрай малим.

1.3.2 Аналіз операційної системи iOS

iOS – операційна система для смартфонів, планшетів та сенсорних програвачів, розроблена американською компанією Apple. Частка iOS на ринку мобільних ОС (29,55%) в точності відображає частку ринку iPhone і iPad (мобільні пристрої від компанії Apple), в той час, як Android і Windows Phone використовуються різними виробниками смартфонів. Кількість додатків для iOS в магазині додатків App Store перевищує мільйон [8, 9].

Дана ОС є дуже привабливою для розробки додатків, оскільки має велику аудиторію користувачів. Однак, не дивлячись на це, існує ряд недоліків:

- кількість користувачів iOS все ж менша, ніж користувачів Android;
- недешевий аккаунт розробника для завантаження додатку в App Store;
- відсутність необхідних інструментів для розробки.

1.3.3 Аналіз операційної системи Android

Android – ОС для смартфонів і багатьох інших пристроїв. Спочатку розроблялась каліфорнійською компанією Android Inc., яку пізніше купив американський пошуковий гігант Google [10, 11]. Частка Android на ринку ОС становить 66,71%. Кількість додатків Android в магазині додатків Google Play перевищує 1,5 млн.

Ця ОС має найбільшу аудиторію користувачів, а отже, як було сказано вище, вона приверне більший відсоток користувачів додатку. Аккаунт розробника для публікації додатку в Play Маркет також платний, проте коштує

значно менше ніж в App Store і платіж здійснюється одноразово (в App Store платіж щорічний).

Проаналізувавши найбільш популярні ОС було прийнято рішення, що розробка буде виконуватись під пристрої з ОС Android, так як вона найкраще відповідає заявленим вимогам.

1.4 Вибір середовища розробки

Найбільш популярними середовищами розробки під операційну систему Android є:

- Eclipse;
- Microsoft Xamarin;
- Android Studio.

Eclipse – модульне інтегроване середовище розробки програмного забезпечення. Розробляється і підтримується Eclipse Foundation і включає проекти, такі як платформа Eclipse, набір інструментів для програмістів на мові Java, системи контролю версій, конструктори GUI тощо. Написаний в основному на Java, може бути використаний для розробки застосувань на Java і, за допомогою різних плагінів, на інших мовах програмування, включаючи Ada, C, C++, COBOL, Fortran, Perl, PHP, Python, Ruby (включно з каркасом Ruby on Rails), Scala, Clojure та Scheme. Середовища розробки зокрема включають Eclipse ADT (Ada Development Toolkit) для Ada, Eclipse CDT для C/C++, Eclipse JDT для Java, Eclipse PDT для PHP [12].

Початок коду йде від IBM VisualAge, він був розрахований на розробників Java, тих, що складали Java Development Tools (JDT). Але користувачі могли розширяти можливості, встановлюючи написані для програмного каркасу Eclipse плагіни, такі як інструменти розробки під інші мови програмування, і могли писати і вносити свої власні плагіни і модулі.

Випущений на умовах Eclipse Public License, Eclipse є вільним програмним

забезпеченням. Він став одним з перших IDE під GNU Classpath і без проблем працює під IcedTea.

Все ж, дане середовище не орієнтована конкретно на розробку мобільних додатків і, тому можливі різні проблеми на етапі розробки, яких можливо уникнути, вибравши інше середовище розробки.

Microsoft Xamarin – це платформа розробки мобільних додатків для створення нативних додатків iOS, Android і Windows, із загального коду C# або .NET, яка дозволяє багаторазово використовувати між платформами від 75% майже до 100% коду. Додатки, написані з допомогою Xamarin і C#, мають повний доступ до інтерфейсів API базової платформи і можливість створювати нативні призначені для користувача інтерфейси, а також компілювати код в машинний, і тому вплив на продуктивність під час виконання буде незначним.

Розробники, які знайомі з C#, .NET та Visual Studio можуть розраховувати на такі ж можливості і продуктивність при роботі з Xamarin для мобільних додатків, включаючи віддалене налагодження на пристроях iOS, Android і Windows, без необхідності вивчати нативні язики програмування, наприклад Objective-C або Java. Однак, багато високопродуктивних додатків з красивими призначеними для користувача інтерфейсами – такі, як NASCAR, Aviva і MixRadio, розроблені з допомогою Xamarin.

Проте, через не високу популярність даного середовища розробки і передбачуваними проблемами з пошуком інформації від неї довелось відмовитись.

Android Studio – це інтегроване середовище розробки (IDE) для роботи з платформою Android, яке було анонсоване 16 травня 2013 року на конференції Google I/O [13]. Це молоде середовище розробки, але вже дуже популярне серед розробників під ОС Android. Воно має ряд позитивних особливостей [14]. Ось деякі з них:

- розширений редактор макетів: WYSIWYG, здатність працювати з UI компонентами за допомогою Drag-and-Drop, функція попереднього перегляду макету на декількох конфігураціях екрану;

- збірка додатків, заснована на Gradle;
- різні види збірок і генерація кількох .apk файлів;
- рефакторинг коду;
- статичний аналізатор коду (Lint), що дозволяє знаходити проблеми продуктивності, несумісності версій тощо;
- вбудований ProGuard і утиліта для підписування додатків;
- шаблони основних макетів і компонентів Android;
- підтримка розробки додатків для Android Wear і Android TV;
- вбудована підтримка Google Cloud Platform, яка включає в себе інтеграцію з сервісами Google Cloud Messaging і App Engine;
- android studio 2.1 підтримує Android N Preview SDK, а це значить, що розробники зможуть почати роботу зі створення програми для нової програмної платформи;
- нова версія Android Studio 2.1 здатна працювати з оновленим компілятором Jack, а також отримала покращену підтримку Java 8 і вдосконалену функцію Instant Run;
- починаючи з Platform-tools 23.1.0 для Linux виключно 64-розрядна;
- в Android Studio 3.0 будуть по стандарту включені інструменти мови Kotlin. засновані на JetBrains IDE.

Також дане середовище розробки, як можна було зрозуміти по його назві, орієнтована на розробку додатків саме під ОС Android, що є безперечним плюсом перед іншими IDE.

1.5 Постановка задачі досліджень

В підсумку, після аналіза існуючих середовищ розробки, було обране саме середовище розробки Android Studio, так як воно ідеально підходить під потреби створеного мною додатка. Крім того, в інтернеті є багато інформації про роботу з ним та процеси його налаштування.

Для реалізації поставленої мети необхідно вирішити наступні задачі:

- виявити недоліки існуючих аналогів;
- вибрати мобільну операційну систему та середовище розробки;
- вибрати мову програмування;
- побудувати структуру додатку;
- розробити додаток;
- описати особливості застосування розробленого додатку.

2 ОБГРУНТУВАННЯ ТЕХНОЛОГІЙ ПРОГРАМУВАННЯ ТА РОЗРОБКА СТРУКТУРИ ДОДАТКУ

В даному розділі буде описано ключову специфіку Android додатків, загальна схема роботи, які версії Android найпопулярніші, вибір технологій за допомогою яких буде реалізовано проект та безпосередньо проектування системи.

2.1 Специфіка розробки додатків на ОС Android

Операційна система Android досить унікальна. Для отримання хорошого результату розробнику додатків може знадобитись знати багато тонкощів і особливостей даної операційної системи. При розробці існують кілька труднощів, які важливо враховувати. Таких як:

- додаток після установки вимагає в два рази, а в деяких випадках навіть в чотири рази більше місця, ніж розмір оригінального програми;
- на вбудованої флеш-карті швидкість роботи з файлами падає в кілька десятків разів при невеликій кількості вільного місця;
- процеси можуть використовувати до 16 Мб, а в деяких випадках і 24 Мб оперативної пам'яті. Між ядрами і додатком лежить свій шар API і на нативному коді - шар бібліотек.

В Android можна запускати необмежену кількість додатків, яку дозволяє оперативна пам'ять в пристрої. Але тільки один з додатків є головним і відображається на екрані. З відкритої програми можна перейти до фонового режиму або запустити нову. Даний процес візуально нагадує вкладки браузера.

Екрани інтерфейсу користувача представлені класом Activity в коді і містяться в процесах. Activity може жити довше процесу [15]. Activity при зупинці може бути запущено знову для збереження всієї потрібної нам

інформації. При цьому використовуються спеціальні механізми опису дій засновані на компоненті Intent. Коли потрібно зробити дзвінок, надіслати листа, показати вікно або виконати дію, викликається Intent.

Також в Android присутні подібні проблеми з сервісами, як в Linux для виконання потрібних дій у фоновому режимі, наприклад, як програвання музики або відео. В таких цілях використовується Content providers [16] (провайдери вмісту) для обміну даними між додатками.

Content providers (Постачальники контенту) – керують доступом до структурованого набору даних. Вони інкапсулюють дані і надають механізми забезпечення їх безпеки. Постачальники контенту є стандартний інтерфейс для об'єднання даних в один процес з кодом, який виконується в іншому процесі.

2.1.1 Загальна схема роботи додатків Android

Додатки для Android використовують вікна в своїй роботі подібно до Windows вікон, хоча в даній системі вікна мають іншу назву - Activity. Як і в Windows, будь-яке вікно має свій життєвий цикл і свої характерні риси. Схема представляє життєвий цикл програми для Android [17] зображена на рис. 2.1.

Activity (операції) – це компонент додатка, який видає екран, з яким користувачі можуть взаємодіяти для виконання будь-яких дій, наприклад набрати номер телефону, зробити фото, відправити лист або переглянути карту. Кожній операції присвоюється вікно для промальовування відповідного для користувача інтерфейсу. Зазвичай вікно відображається на весь екран, проте його розмір може бути менше, і воно може розміщуватись поверх інших вікон.

Зазвичай, програма містить декілька операцій, які слабо пов'язані один з одним. Зазвичай одна з операцій в додатку позначається як «основна», пропонована користувачеві при першому запуску програми. У свою чергу, кожна операція може запустити іншу операцію для виконання різних дій. Кожен раз, коли запускається нова операція, попередня операція зупиняється, однак

система зберігає її в стеку. При запуску нової операції вона поміщається в стек і нова операція відображається для користувача. Стек працює за принципом «останнім увійшов – першим вийшов», тому після того як користувач завершив поточну операцію і натиснув кнопку Назад, поточна операція видаляється з стека (і знищується), і відновлюється попередня операція.

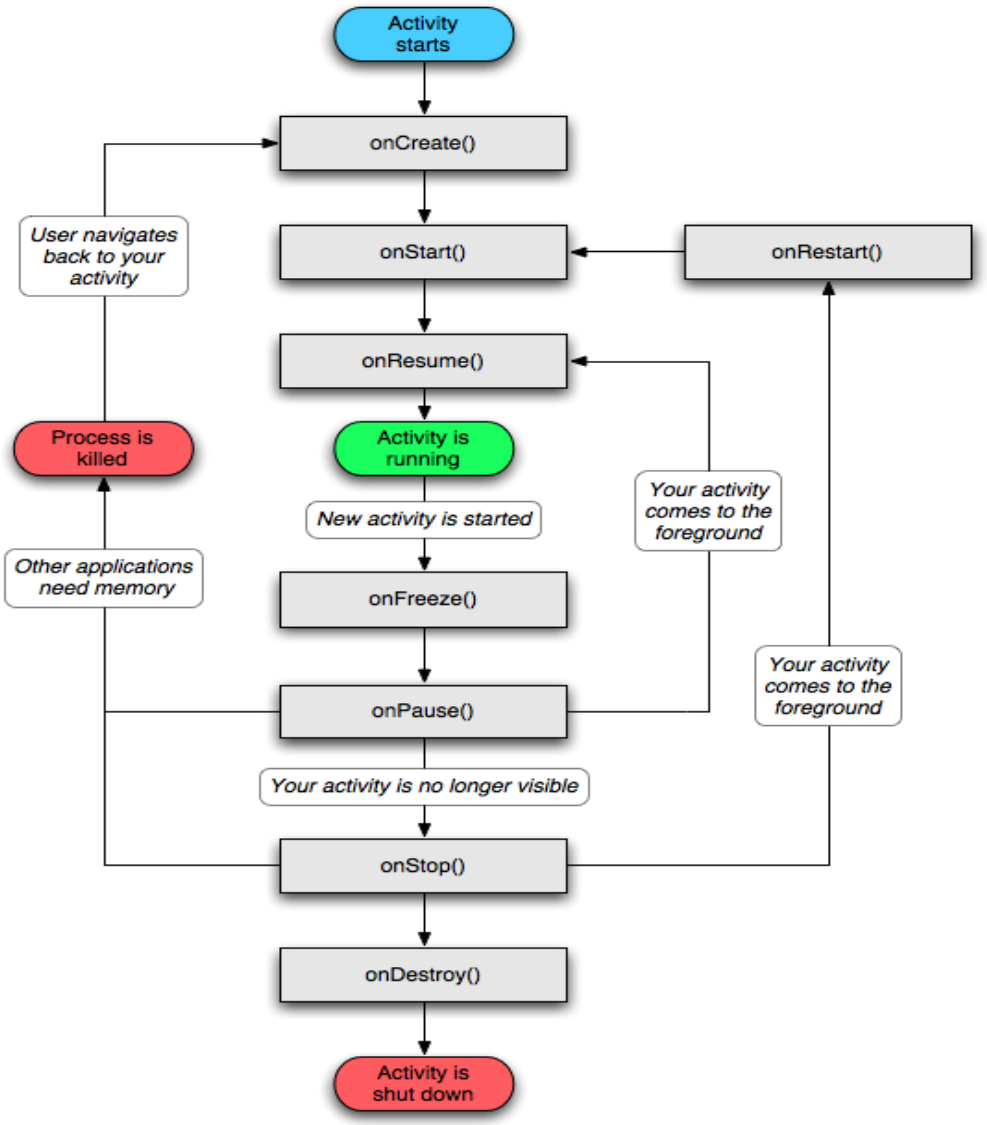


Рисунок 2.1 – Життєвий цикл Activity (операції) в системі

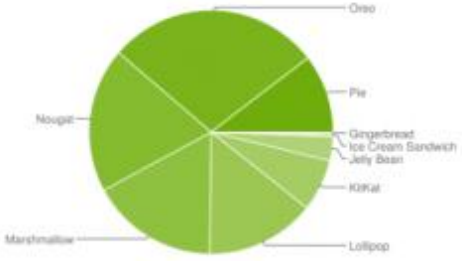
Для того щоб створити нове вікно викликається метод onCreate (), при розробки цей метод перепризначувався і в ньому відбувається ініціювання програми та його елементів. Далі слід метод onStart () і метод onResume (). Які визиваются перед показом онка при його створенні, або його відновлення (при

переході з іншої програми, при відкритті згорнутого додатки і так далі). При згортанні використовується метод onDestroy (), в даному методі зберігаються параметри і дані користувача. Щоб отримати послідовність викликів методів і більш повний опис їх можна знайти на офіційному сайті [18].

2.1.2 Статистика розподілу версій ОС Android

Компанія Google опублікувала статистику розподілу діючих версій операційної системи Android, як показано на рис. 2.2 станом на 7 травня 2019 року. Після публікації статистики Google представили нову операційну систему Android 10 Q, яка поки не отримала назви, а також тим, що останній раз подібну статистику Google публікувала в жовтні минулого року [19].

Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	0.3%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.3%
4.1.x	Jelly Bean	16	1.2%
4.2.x		17	1.5%
4.3		18	0.5%
4.4	KitKat	19	6.9%
5.0	Lollipop	21	3.0%
5.1		22	11.5%
6.0	Marshmallow	23	16.9%
7.0	Nougat	24	11.4%
7.1		25	7.8%
8.0	Oreo	26	12.9%
8.1		27	15.4%
9	Pie	28	10.4%



Data collected during a 7-day period ending on May 7, 2019
Any versions with less than 0.1% distribution are not shown.

Рисунок 2.2 – Статистика розподілу версій ОС Android станом на 7.05.2019 року

Найпоширенішою, на даний час, є ОС Android 8.0-8.1 Oreo з сумарною часткою 28.3%. На другому місці опинилася ОС Android 7.0-7.1 Nougat з 19.2% всіх сумісних пристроїв. Третьою за поширеністю є ОС Android 6.0 Marshmallow з 16.9%, а четвертої Lollipop 5.0-5.1 – 14.5% ринку.

Решта версій розподілилися наступним чином: Android 9.0 Pie - 10.4%, KitKat 4.4 – 6.9%, Jelly Bean 4.1-4.3 – 3.2%, а Ice Cream Sandwich 4.0.3-4.0.4 і Gingerbread 2.3.3-2.3 .7 - по 0.3% відповідно.

Відзначається, що настільки швидкому поширенню ОС Android 9.0 Pie зобов'язана ініціативи Google Project Treble, запущеної в травні 2017 року. Всі нові смартфони з Android 9.0 Pie повністю сумісні з Project Treble, тому в майбутньому, ймовірно, оновлення будуть поширюватися ще швидше.

2.2 Обгрунтування вибору технологій Java і SQLite

Загалом, програми під Android можна створити практично на будь-яких популярних мовах (таких як, «Java, Kotlin, C/C++, Python») – фреймворки і утиліти найдуться під все. Однак професійні Android-розробники, використовують всі можливості операційної системи і мають доступ до найновіших функцій Android, за допомогою Java або Kotlin. Навіть якщо ці мови здаються складними, їх варто вивчити, щоб писати різноманітні, красиві і функціональні додатки для операційної системи Android [20].

Android не підтримує створення нативних додатків з використання Python, але це не означає, що це неможливо. Любителі цієї мови розробили безліч інструментів, що дозволяють скомпілювати код на Python в потрібний стан, а наявність різних бібліотек дозволить будувати навіть нативні інтерфейси з дотриманням Material Design. Найпопулярнішим фреймворком є Kivy, який дозволить вам створити додаток для Play Market на чистому Python [21].

C/C++ низькорівневі мови, які підтримуються Android Studio з використанням Java NDK. Це дозволяє писати нативні додатки, що може стати в

нагоді для створення ігор або інших ресурсоємних програм. Android Studio пропонує підтримку C/C++ через Android NDK (Native Development Kit). Це означає, що код буде запускатись не через Java Virtual Machine, а безпосередньо через девайс, що дасть вам більше контролю над такими елементами системи, як пам'ять, сенсори, жести і тощо, а також дає можливість задіяти максимум ресурсів Android-пристроїв . Це означає, що користуватися вам доведеться бібліотеками, написаними на C або C++. У свою чергу, він складний в налаштуванні та не дуже зручний, тому рекомендується використовувати його для написання тільки тих модулів програми, де необхідно швидко робити складні операції: обробку і рендеринг графіки, відео і складних 3D-моделей [22].

Kotlin - була офіційно представлена в травні 2017 року на Google I/O і позиціонується Google, як друга офіційна мова програмування під Android після Java, тільки трохи проста для розуміння. Знання Java необхідні тут, щоб розуміти принципи роботи Kotlin, загальну структуру мови та її особливості. Багато розробники вважають Kotlin обгорткою над Java і рекомендують вивчати його тільки після того, як ви відчуєте впевненість в своїх знаннях Java. Kotlin сумісний з Java і не викликає зниження продуктивності і збільшення розміру файлів. Відмінність від Java в тому, що він вимагає менше службового, так званого boilerplate-коду, тому більш легкий для читання. Його творцям вдалося уникнути nullpointexceptions, а компіляція більше не переривається через дрібниці на зразок забутого знака «;» [23, 24].

Java - офіційна мова програмування, підтримувана середовищем розробки Android Studio. За даними щорічного опитування ресурсу Stackoverflow [25], в 2018 році Java увійшла до списку п’яти найпопулярніших мов програмування.

На Java посилається більшість офіційної документації Google, а знайти платні і безкоштовні бібліотеки та документацію не важко – їх безліч. Код на Java легко читається і структурується, особливо при дотриманні прийнятих стандартів його оформлення.

При розробці на Java під Android використовуються не тільки Java-класи, що містять код, але також файли маніфесту на мові XML, що надають системі

основну інформацію про програму, і системи автоматичного складання Gradle, Maven або Ant, команди в яких пишуться на мовах Groovy, POM і XML відповідно; за замовчуванням в проектах використовується Gradle, а на початкових етапах розробці на Java правити файли, написані на Groovy, практично не доведеться. Для верстки UI-частини зазвичай також використовується мова XML [26, 27].

Android Studio, в грудні 2014 року визнана Google офіційним середовищем розробки під ОС Android, яка вдосконалюється з року в рік, чим полегшує життя Android-розробникам. Її можливості, як візуальний UI-редактор і автодоповнення коду, допомагають зробити процес розробки більш комфортним. Тим, хто готовий до повного занурення в Android-розробку, Java рекомендується в першу чергу.

Java – об'єктно-орієнтована мова програмування, що розробляється компанією Sun Microsystems з 1991 року і офіційно випущений 23 травня 1995 року. Спочатку нову мову програмування називався Oak і розроблявся для побутової електроніки, але згодом був перейменований в Java і став використовуватися для написання аплетів, додатків і серверного програмного забезпечення [28].

Мова Java зародилася як частина проекту створення передового програмного забезпечення для різних побутових приладів. Реалізація проекту була почата на мові C ++, але незабаром виник ряд проблем, найкращим засобом боротьби з якими була зміна самого інструмента – мови програмування. Стало очевидним, що необхідний платформо-незалежний мова програмування, що дозволяє створювати програми, які не доводилося б компілювати окремо для кожної архітектури і можна було б використовувати на різних процесорах під різними операційними системами.

Мова Java потрібен для створення інтерактивних продуктів для мережі Internet. Фактично, більшість архітектурних рішень, прийнятих при створенні Java, було продиктовано бажанням надати синтаксис, подібний до C і C ++. В Java використовуються практично ідентичні угоди для оголошення змінних,

передачі параметрів, операторів і для управління потоком виконанням коду. В Java додані всі хороші риси C ++.

Три ключові елементи об'єдналися в технології мови Java:

- java надає для широкого використання свої аплети (applets) – невеликі, надійні, динамічні, які не залежать від платформи активних мережевих додатків, що вбудовуються в сторінки Web. Аплети Java можуть налаштовуватися і поширюватися споживачам з такою ж легкістю, як будь-які документи HTML;
- java демонструє силу об'єктно-орієнтованої розробки додатків, поєднуючи простий і знайомий синтаксис з надійним і зручним в роботі середовищем розробки. Це дозволяє широкому колу програмістів швидко створювати нові програми і нові аплети;
- java надає програмісту великій набір класів та об'єктів для ясного абстрагування багатьох системних функцій, використовуваних при роботі з вікнами, мережею і для вводу-виводу. Ключова риса цих класів полягає в тому, що вони забезпечують створення незалежних від використовуваної платформи абстракцій для широкого спектра системних інтерфейсів.

Головною перевагою Java є оновлення і підтримка мови, велика кількість написаних бібліотек, які вирішують найрізноманітніші завдання, а також те, що написані на мові програмування Java додатки можуть бути запущені на всіх пристроях, на яких встановлена віртуальна машина Java. Для створення програмного забезпечення для ОС Android використовується версія Java 1.7. Написані програми компілюються в байт-код, який запускається в віртуальній машині JVM. У Android використовується власна віртуальна машина Dalvik, а починаючи з версії 4.4 - Art, що накладає деякі обмеження кількість методів в додатку і на використання стандартних бібліотек містяться в Java SE.

За замовчуванням в Android використовується SQLite - популярна і проста в освоєнні реляційна база даних. SQLite підтримує типи TEXT (аналог String в

Java), INTEGER (аналог long в Java) і REAL (аналог double в Java). Решта типів слід конвертувати, перш ніж зберігати в базі даних [29].

Бібліотека Android містить абстрактний клас SQLiteOpenHelper, з допомогою якого можна створювати, відкривати і оновлювати бази даних. Це основний клас, з яким здійснюється робота в проекті. при реалізації цього допоміжного класу приховується логіка, на основі якої приймається рішення про створення або оновленні бази даних перед її відкриттям.

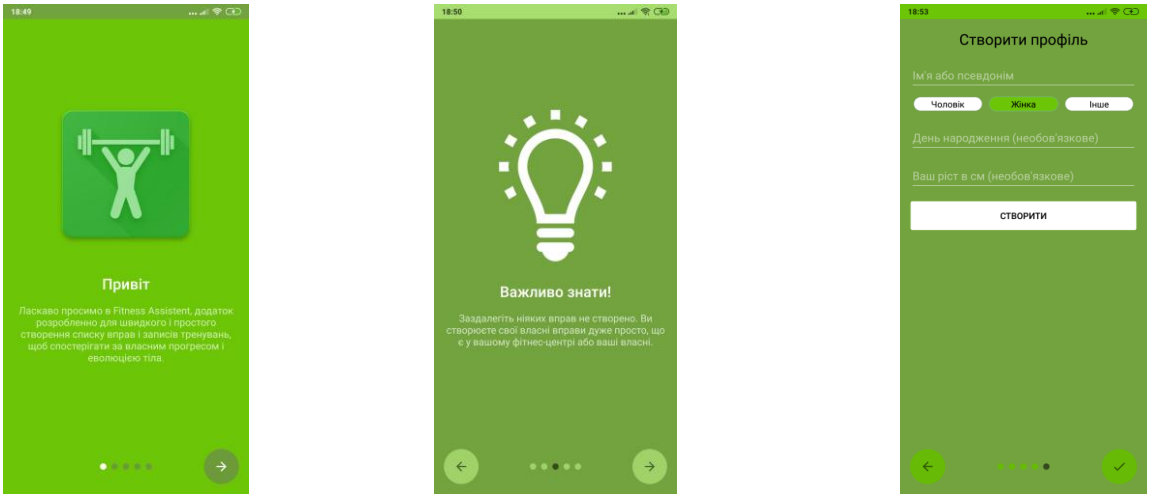
Клас SQLiteOpenHelper містить два абстрактних метода: onCreate () – метод, який викликається при першому створенні бази даних, а також onUpgrade (), який викликається при модифікації бази даних. У додатку створений власний клас DatabaseHelper, успадкований від SQLiteOpenHelper. В цьому класі реалізовані методи onCreate () і onUpgrate (). У них описана в них логіка створення і модифікації бази даних. У фрагменті коду, наведеному в Додатку А, описано створення бази даних за допомогою методу onCreate. При це метод onCreate викликається тільки один раз при створенні бази даних. Запит до бази даних виконується за допомогою виклику SQLiteDatabase :: query(). В результаті виконання запитів повертається об'єкт Cursor, що містить таблицю з результатами запиту. Cursor передбачає послідовну роботу з рядками результату. У кожен момент часу активний один рядок, на яку посилається показник. перебираючи записи послідовно, можна отримати доступ до даних.

2.3 Проектування структури додатку

При проектуванні, на сам перед, була розроблена структура додатку, UML діаграма класів та use-case діаграма в якій прописані передбачені дії користувача при взаємодії з додатком.

2.3.1 Логічна структура додатку

При першому запуску додатку, відкривається вікно вступу, яке починається з вітання користувача і незначного опису про додаток, наостанок користувачу доведеться створити профіль який складається хоча б з його імені (псевдоніма) і також можуть бути вказані дані про дату народження та свій ріст (необов’язкові поля вводу), рис. 2.3.



а б в

Рисунок 2.3 – Представлення (вступ) додатку при першому використанні:
а – вікно привітання, б – інформація про додаток, в – створення профілю

Діаграма , представлена на рис. 2.4, відображає дії користуваччa доступні в додатку при роботі з ним. Від самого початку для користувачів не буде добавлено жодних вправ, для того, щоб не обмежувати їх або захаращувати пам'ять не потрібними для них вправами. Їм буде представлена можливість створити власний список потрібних їм вправ (назву, опис та фото), які поділені на дві категорії (фітнес/бодібілдинг або кардіо). Також можна створити/видалити профіль або редагувати поточний, створення записів тренувань і відслідковування їх. Також буде можливість імпорту чужої бази даних (фітнес центрі в якому займаються) або експорту власної бази даних.

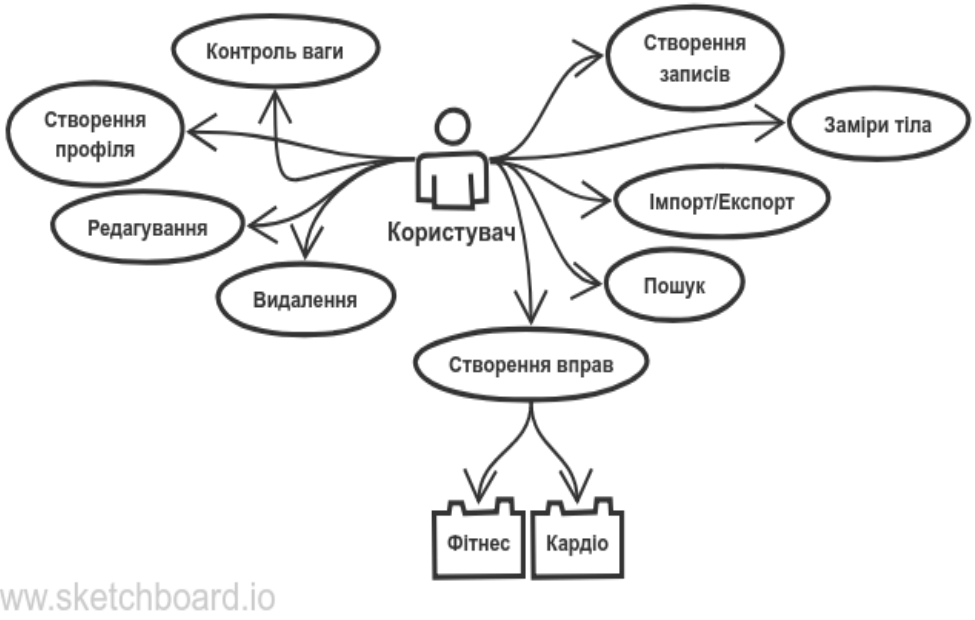


Рисунок 2.4 – Use-case діаграма доступних дій користувача при роботі з базою даних

Потоки даних в даній діаграмі поширюються від бази даних до інших елементів за допомогою SQL-запитів. При завантаженні програми основна активність звертається до бази даних, яка перебуває на диску. Якщо на диску база даних не виявлено, то створюється порожня база даних на локальному диску. Таким чином, користувач може вибрати цікавлять його категорії об'єктів. При виборі категорії в меню завантажується активність «Категорія», яка звертається до бази даних і, прочитавши можливі категорії, відображає їх у списку. Ця активність теж звертається до бази даних (Формуючи запит) і виводить всі підкатегорії. Вихід з цих активностей здійснюється за допомогою кнопки «Назад». При необхідності отримання подальшої інформації завантажується наступна активність, куди передається ID номер обраного об'єкта.

Після створення вправ, користувач переходить до «Тренування», щоб створити запис заняття, перегляд історії та графік занять. В вкладці «Вправи» буде доступна інформація про вправи які були створені ним заздалегідь або

можливість створити вправу вибравши її тип (вправа буде створена по вказаній назві, щоб добавити для неї опис і фото прийдеться перейти до меню «Вправи» де з може редагувати її знайшовши в списку), як показано на рис. 2.5 . В вкладках «Графік» та «Історія» будуть доступні ваші збережені записи (з інформацією про те скільки підходів, повторів ви зробили у вигляді графіка та списку.

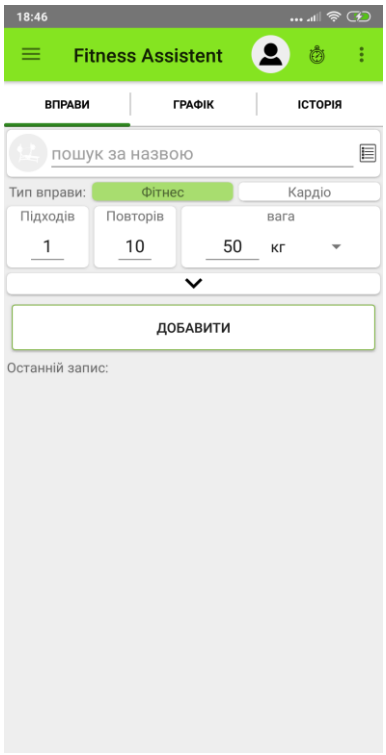


Рисунок 2.5 – Головна активність при запуску додатку

2.3.2 Архітектура додатку

Програмний додаток для ОС Android складається з набору активностей, кожній з яких відповідає вікні керування. кожна активність представлена в проекті класом, реалізований на мові Java, що зберігається в однойменному файлі з розширенням .java. Кожній активності відповідає xml файл-опис. В xml-файлі описано у вигляді xml-коду розташування візуалізуються об'єктів. При запуску активності система Android автоматично розпізнає розмір екрану

мобільного пристрою і призводить виведений контент у відповідність з розміткою, описаної в xml-файлі. Таким чином, одна і та ж активність буде виглядати однаково незалежно від діагоналі використовуваного пристрою. Також, для кожного додатку Android повинен існувати xml-файл, в якому у вигляді xml-коду будуть прописані мінімальні вимоги до системи, а також активність, яка викликається при запуску програми [30].

Додаток працює з вбудовуваної реляційної базою даних SQLite. 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. Таким чином, в якості протоколу обміну використовуються виклики функцій (API) бібліотеки SQLite. Такий підхід зменшує накладні витрати, час відгуку і спрощує програму. SQLite зберігає всю базу даних (включаючи визначення, таблиці, індекси і дані) в єдиному стандартному файлі на тому пристрої, на якому виповнюється програма [31].

2.3.3 Список модулів

Функціонально, додаток складається з наведених нижче модулів (Активностей). Активність є схемою уявлення Android додатку.

Кожен екран для користувача інтерфейсу представлений класом Activity і по суті є окремою формою додатка. Android-додаток здатний складатися з декількох активностей і може перемикатися між ними під час виконання програми.

Схема активностей розробленого мною додатку, представлена нижче на рис. 2.6 за допомогою діаграми класів UML.

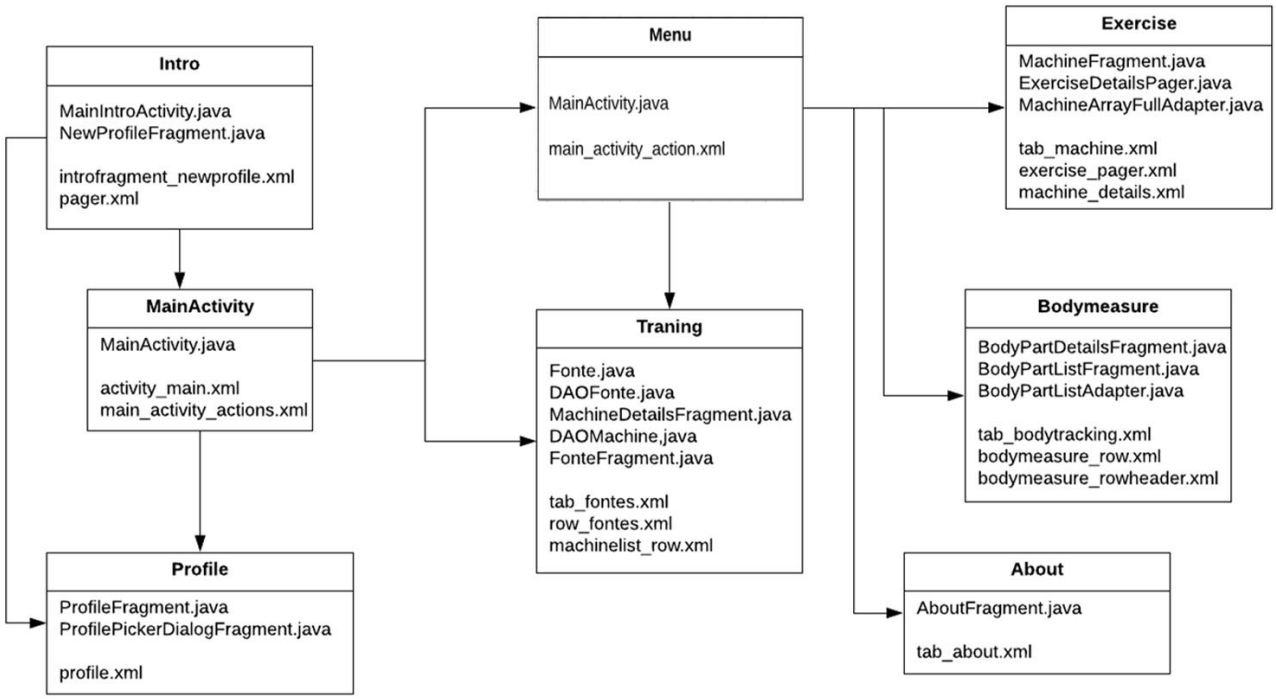


Рисунок 2.6 – Керуючі класи (activity додатку) і зв’язки між ними

Діаграма UML потрібна тоді коли необхідно короткий, лаконічний, але разом з тим зрозумілий і наочний засіб для опису проекту. Як узагальнено, так і більш конкретно на рівні окремих класів. Тут і приходить на допомогу UML.

3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ЗАСТОСУНКУ З НАДАННЯ ФІТНЕС-ПОСЛУГ

В даному розділі розроблені головні Activity додатку і показані методи їх реалізації та проведено тестування додатку.

3.1 Опис функціоналу

Перед тим, як почати користуватись додатком, необхідно створити профіль, як показано на рисунку 3.2. При першому створені профілю він містить такі поля, як:

- ім’я або псевдонім;
- вибір статі;
- день народження;
- ріст.

Профілів може бути декілька, це необхідно, якщо потрібно відгородити вправи і історію записів з двох різних залів, або ж коли смартфоном користуються двоє чи більше людей. Щоб зробити новий профіль потрібно натиснути на піктограму портрету і відкриється діалогове меню, як показано на рисунку 3.1.

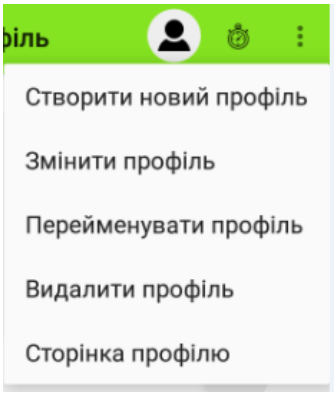


Рисунок 3.1 – Випадаючий список дій над профілем

Нижче приведений фрагмент коду створення профілю який розміщений у файлі introfragment_newprofil.xml, який відповідає створенню профіля вперше. Графічне зображення форми створення профілю показано на рисунку 3.2.

```
<android.support.design.widget.TextInputLayout
  android:id="@+id/signup_input_layout_name3"
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:layout_marginTop="5dp">

<EditText
  android:id="@+id/profileSize"
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:hint="@string/size_cm_optional"
  android:inputType="numberSigned"
  android:singleLine="true"
  android:textColor="@android:color/black"

<Button
  android:id="@+id/create_newprofil"
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:layout_marginTop="@dimen/mi_baseline"
  android:background="@drawable/round_corner_button"
  android:text="@string/create"
  android:textColor="@android:color/black" />
```

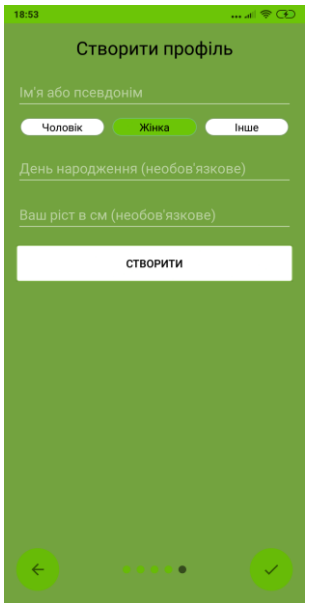


Рисунок 3.2 – Графічне зображення створення профілю в перше

Коли дані введено і користувач натиснув «СТВОРИТИ», дані відправляються у базу даних, і зберігаються в таблиці DBprofil, яка знаходиться у файлі DAOProfile.java (повний код файлу наведено у Додатку Б). Нижче приведений фрагмент коду полів:

```
public static final String TABLE_NAME = "DBprofil";

public static final String KEY = "_id";
public static final String NAME = "name";
public static final String CREATIONDATE = "creationdate";
public static final String SIZE = "size";
public static final String BIRTHDAY = "birthday";
public static final String PHOTO = "photo";
public static final String GENDER = "gender";

public static final String TABLE_CREATE = "CREATE TABLE " +
TABLE_NAME + " (" + KEY + " INTEGER PRIMARY KEY AUTOINCREMENT, " +
CREATIONDATE + " DATE, " + NAME + " TEXT, " + SIZE + " INTEGER, " +
BIRTHDAY + " DATE, " + PHOTO + " TEXT, " + GENDER + " INTEGER);";

public static final String TABLE_DROP = "DROP TABLE IF EXISTS " +
TABLE_NAME + ";";

private Cursor mCursor = null;
```

Якщо користувач не ввів ім’я, тоді задіяний буде Toast (короткочасне повідомлення), який повідомляє користувача, що потрібно ввести хоча б ім’я або псевдонім. Даний приклад і метод реалізації звернення наведено на рис. 3.3

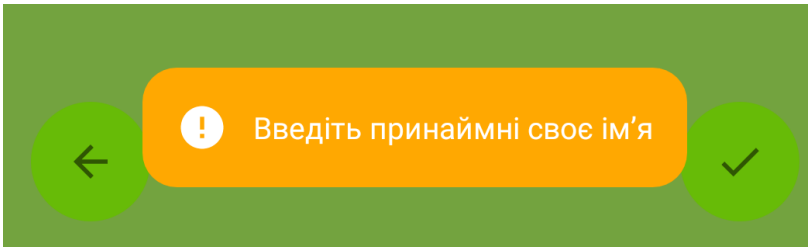


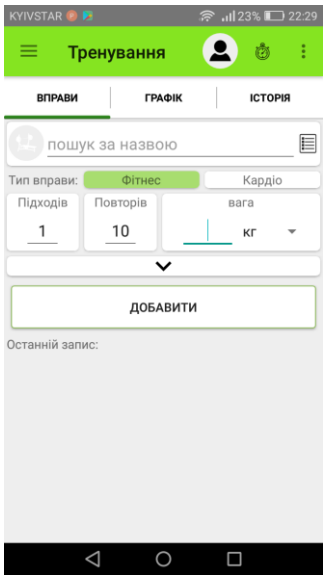
Рисунок 3.3 – Повідомлення користувача про некоректну дію

```
if (mName.getText().toString().isEmpty()) {
    KToast.warningToast(getActivity(),
getResources().getText(R.string.fillNameField).toString(),
Gravity.BOTTOM, KToast.LENGTH_SHORT);
} else {
```

```
    int size = 0;
    try {
        if (!mSize.getText().toString().isEmpty()) {
            size =
Double.valueOf(mSize.getText().toString()).intValue();
        }
    } catch (NumberFormatException e) {
        size = 0;
    }
```

При старті додатку головним activity є блок «Тренування». В даному блоці можна робити записи своїх тренувань, як показано на рис. 3.4 із заздалегідь створеним списком вправ. Щоб створити запис, потрібно в полі пошуку ввести назву вправи, або вибрати її з піктограми списку при натисканні на неї. Фрагмент коду реалізації поля пошуку і піктограми наведені нижче:

```
private OnFocusChangeListener touchRazEdit = new
View.OnFocusChangeListener() {
      @Override
public void onFocusChange(View v, boolean hasFocus) {
   if (hasFocus == true) {
      switch(v.getId()) {
      case R.id.editCardioDate:
                  showDatePicker();
                  break;
                  case R.id.editDuration:
                  showTimePicker();
                  break;
                  case R.id.editSerie:
         distanceEdit.setText("");
         break;
      case R.id.editPoids:
         durationEdit.setText("");
         break;
      case R.id.editMachine:
                  exerciceEdit.setText("");
         break;
      }
   }else if (hasFocus == false) {
      switch(v.getId()) {
      case R.id.editMachine:
         FillRecordTable(exerciceEdit.getText().toString());
         break; }
}
```



Рисунки 3.4 – Операція (activity) «Тренування» при старті додатку та виборі з меню навігації

Графік відображає інформацію про конкретну вправу, показуючи дату її запису, як показано на рисунку 3.5, а також можна задати щоб на графіку відображались записи від певної кількості повторів вправи. Для реалізації графіку була використана бібліотека «com.github.hannesa2:MPAndroidChart:3.0.5» в якій потрібно було переопріділити метод, фрагмент коду:

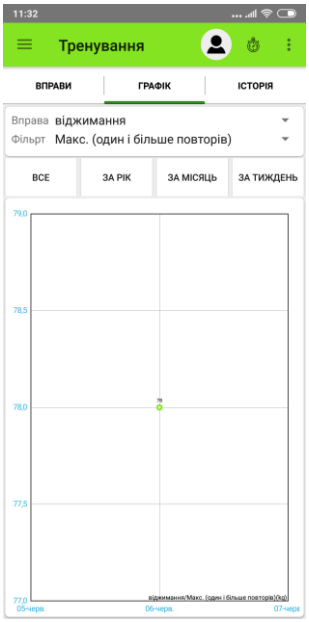


Рисунок 3.5 – Графік записаних вправ

Історія вправ (рис. 3.6), показує записи всіх вправ або конкретно вибрану вправу, також їх можна сортувати по даті запису, список записів містять наступну інформацію: назву, дату, кількість підходів та повторів і вагу. Фрагмент коду:

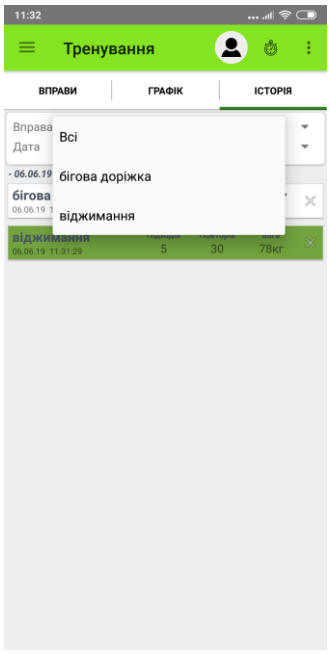


Рисунок 3.6 – Історія записів

Щоб пересуватись по додатку, реалізоване меню навігації (Navigation Drawer Activity), щоб відкрити меню, яке зображено на рис 3.8, потрібно натиснути на піктограму списку рис. 3.7 або натиснувши на лівий кут екрану і потягнути до середини або дальше. Фрагмент коду реалізації меню навігації:

```
private class DrawerItemClickListener implements
ListView.OnItemClickListener {
    public void onItemClick(AdapterView parent, View view, int
position, long id) {
        selectItem(position);
        switch (position) {
            case 0:
                showFragment(PROFILE);
                setTitle(getString(R.string.ProfileLabel));
                break;
            case 1:
                showFragment(FONTESPAGER);
setTitle(getResources().getText(R.string.menu_Workout));
                break;
```

```
            case 2:
                showFragment(MACHINES);
setTitle(getResources().getText(R.string.MachinesLabel));
                break;
            case 3:
                showFragment(WEIGHT);
setTitle(getResources().getText(R.string.weightMenuLabel));
                break;
            case 4:
                showFragment(BODYTRACKING);
setTitle(getResources().getText(R.string.bodytracking));
                break;
            case 5:
                showFragment(SETTINGS);
setTitle(getResources().getText(R.string.SettingLabel));
                break;
            case 6:
                showFragment(ABOUT);
setTitle(getResources().getText(R.string.AboutLabel));
                break;
            default:
                showFragment(FONTESPAGER);
setTitle(getResources().getText(R.string.FonteLabel));
```

Рисунок 3.7 – Піктограма меню навігації

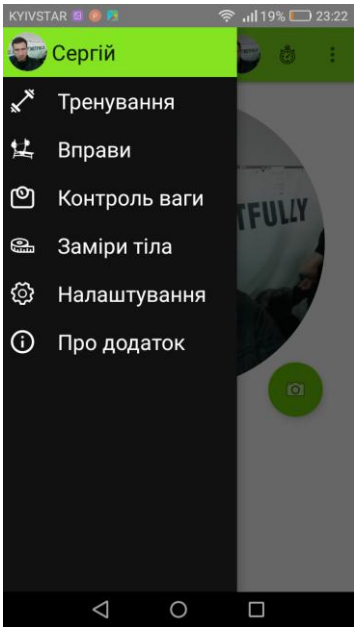


Рисунок 3.8 – Меню навігації (Navigation Drawer Activity)

При створені вправи, потрібно вказати тип вправи (як показано на рис. 3.9), після вибору типу вправи, користувача буде перенаправлено на сторінку

створення вправи, на якій потрібно вказати (назву та опис), як показано на рис. 3.10, фрагмент коду (повний код буде зображено в Додатку В):

```
<EditText
    android:id="@+id/machine_name"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:gravity="center_vertical"
    android:inputType="text"
    android:lines="1"
    android:maxLines="1"
    android:text="@string/exercise_name"
    android:textSize="14sp" />

<TextView
    android:id="@+id/machine_description_textview"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:gravity="center_vertical"
    android:text="@string/description"
    android:textSize="12sp" />
```

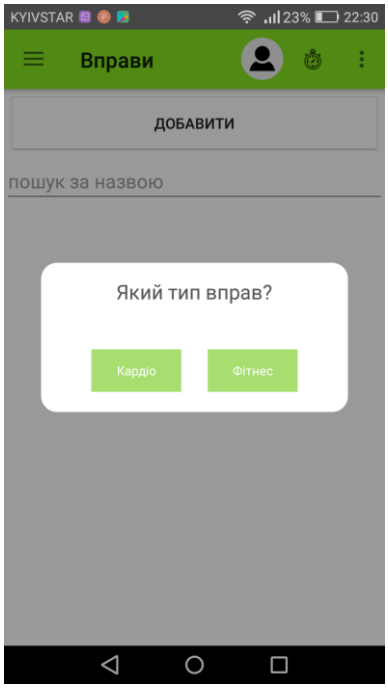


Рисунок 3.9 – Діалогове вікно при створені вправи з вибором її типу

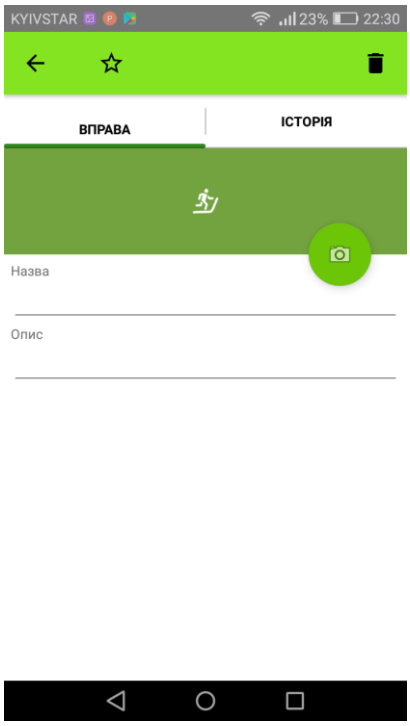


Рисунок 3.10 – Сторінка створення вправи

Щоб видалити вправу потрібно перейти по ній, і клацнути на іконку смітника, після цього з’явиться діалогове вікно з підтвердженням дії, як показано на рис. 3.11, фрагмент коду:

```
private void deleteMachine() {
    AlertDialog.Builder deleteDialogBuilder = new
AlertDialog.Builder(this.getActivity());


deleteDialogBuilder.setTitle(getActivity().getResources().getText(R
.string.global_confirm));

deleteDialogBuilder.setMessage(getActivity().getResources().getText
(R.string.deleteMachine_confirm_text));

deleteDialogBuilder.setPositiveButton(this.getResources().getString
(R.string.global_yes), new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            mDbMachine.delete(machine);
            deleteRecordsAssociatedToMachine();
            getActivity().onBackPressed();
        }
    });
deleteDialogBuilder.setNegativeButton(this.getResources().getString
```

```
(R.string.global_no), new DialogInterface.OnClickListener() {

        @Override
        public void onClick(DialogInterface dialog, int which) {
            dialog.dismiss();
        }
    });
```

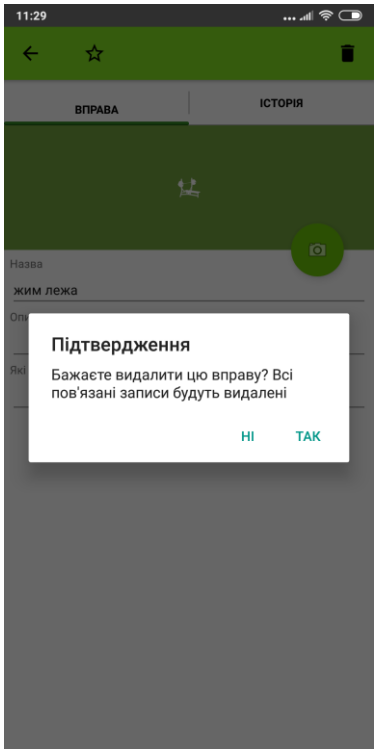


Рисунок 3.11 – Підтвердження дії про видалення вправи

ПункТ меню «Контроль ваги» призначений для внесення даних про зміни в організмі, пов’язані з вагою і тощо. Він зображений на рис. 3.12, фрагмент коду реалізації цього пункту приведений нижче (повний код дивіться в Додатку Г):

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_gravity="center_vertical|center_horizontal"
    android:gravity="center_horizontal"
    android:orientation="vertical">
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="top|center_horizontal"
        android:layout_marginTop="10dp"
```

```
        android:autoSizeMaxTextSize="18sp"
        android:autoSizeMinTextSize="14sp"
        android:autoSizeStepGranularity="1dp"
        android:gravity="center_horizontal"
        android:lines="1"
        android:text="@string/weightLabel"
        android:textSize="18sp" />
<com.easyfitness.utils.EditableInputView.EditableInputViewWithDate
        android:id="@+id/weightInput"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
android:layout_gravity="center_horizontal|center|clip_vertical"
        android:layout_marginTop="5dp"
        android:layout_weight="1"
        android:gravity="center_vertical|center_horizontal"
        android:inputType="numberSigned|numberDecimal"
        android:lines="1"
        android:maxWidth="100dp"
        android:maxLines="1"
        android:text=" - "
        android:textColor="@android:color/black"
        android:textSize="30sp"
        app:iconVisible="false"
        tools:text="70.5" />
    <ImageButton
        android:id="@+id/weightDetailsButton"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginLeft="5dp"
        android:layout_marginRight="5dp"
        android:adjustViewBounds="false"
        android:background="@color/background"
        android:paddingBottom="5dp"
        android:paddingTop="5dp"
        app:srcCompat="@drawable/ic_baseline_list_alt_24px" />
</LinearLayout>
```

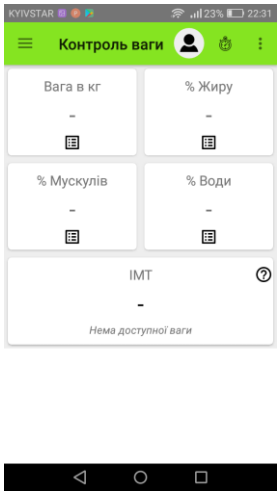


Рисунок 3.12 – Операція (activity) про внесення даних зв’язані з вагою

Щоб дізнатись, як вираховується ІМТ, потрібно в його блоці клацнути на знак питання, і відкриється діалогове вікно з формулою, як показано на рис. 3.13. Метод реалізації ІМТ:

```
private String getImcText(float imc) {
    if (imc<18.5) {
        return getString(R.string.underweight);
    } else if (imc < 25) {
        return getString(R.string.normal);
    } else if (imc < 30) {
        return getString(R.string.overweight);
    } else {
        return getString(R.string.obese);
    }
}
```

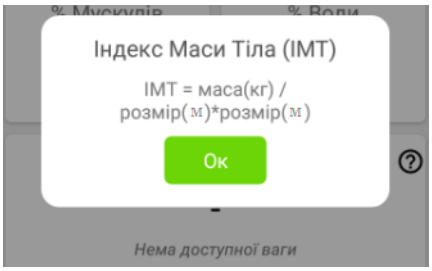


Рисунок 3.13 – Формула ІТМ

В пункті «Заміри тіла», можна ввести запити про зміни об’ємів певних частин тіла, таких як: біцепс; грудни; талія; стегна; ікри. Рисунок 3.14 данного пункта і фрагмент коду реалізації (повний код дивіться в Додатку Д):

```
private static int getBodyLogoID(int pBodyID) {
    switch(pBodyID){
        case ABDOMINAUX: return R.drawable.ic_chest;
        case ADDUCTEURS: return R.drawable.ic_leg;
        case BICEPS: return R.drawable.ic_arm;
        case TRICEPS: return R.drawable.ic_arm;
        case DELTOIDS: return R.drawable.ic_chest;
        case MOLLETS: return R.drawable.ic_leg;
        case PECTORAUX: return R.drawable.ic_chest_measure;
        case DORSEAUX: return R.drawable.ic_chest;
        case QUADRICEPS: return R.drawable.ic_leg;
        case ISCHIOJAMBIERS: return R.drawable.ic_leg;
        case LEFTARM: return R.drawable.ic_arm_measure;
        case RIGHTARM: return R.drawable.ic_arm_measure;
        case LEFTTHIGH: return R.drawable.ic_tight_measure;
```

```
        case RIGHTTHIGH: return R.drawable.ic_tight_measure;
        case LEFTCALVES: return R.drawable.ic_calve_measure;
        case RIGHTCALVES: return R.drawable.ic_calve_measure;
        case WAIST: return R.drawable.ic_waist_measure;
        case NECK: return R.drawable.ic_neck;
        case BEHIND: return R.drawable.ic_buttock_measure;
    return 0;
}
```

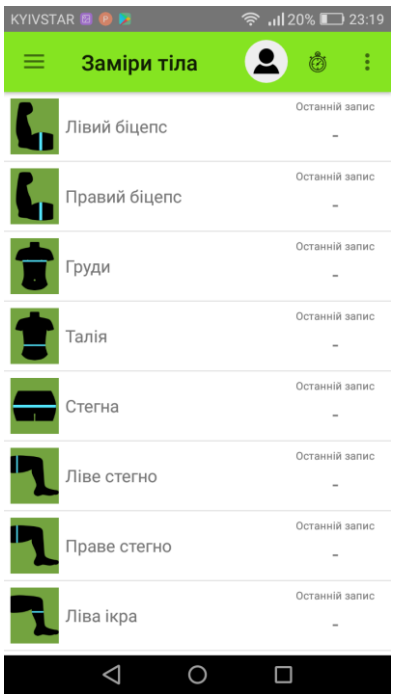


Рисунок 3.14 – Меню «Заміри тіла»

При натисканні на будь-яку частину для вимірювання тіла, відкриється activity де можна вписати результати заміру, дата відвантажується автоматично (з дати, яка встановлена на телефоні або якщо включений інтернет, то з нього) і графік, який по датах виводить записи з заміром, як показано на рис. 3.15, фрагмент коду:

```
<LinearLayout
    android:id="@+id/layoutdown"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_below="@id/layoutup"
    android:layout_toLeftOf="@id/buttonAddWeight"
    android:layout_toStartOf="@id/buttonAddWeight"
    android:orientation="horizontal">
```

```
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="center_vertical"
    android:layout_weight="40"
    android:text="@string/measureLabel" />
<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_weight="10"
    android:orientation="horizontal">
<EditText
    android:id="@+id/editWeight"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_weight="10"
    android:hint="@string/MesureHint"
    android:inputType="numberDecimal"></EditText>
```

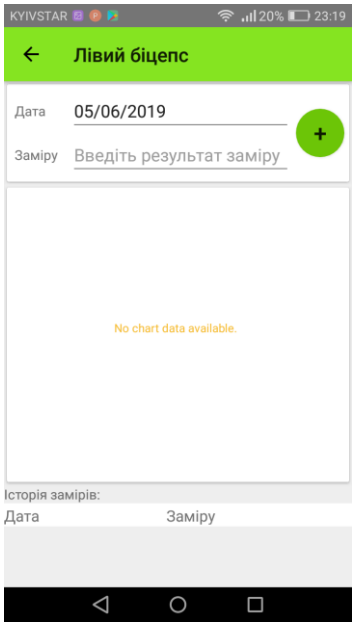


Рисунок 3.15 – Внесення результату вимірювань

В пункті меню «Налаштування» можна вибрати одиницю вимірювання ваги та mp3 плеєр (при натиску на checkbox, внизу екрана появиться TabBar, який показано на рис 3.17), меню «Налаштування» зображено на рис. 3.16 та весь код реалізації:

```
<?xml version="1.0" encoding="utf-8"?>
<PreferenceScreen
xmlns:android="http://schemas.android.com/apk/res/android">
```

```
    <PreferenceCategory
android:title="@string/pref_bodybuilding_setting">
        <android.support.v7.preference.ListPreference
            android:title="@string/pref_preferredUnitTitle"
            android:summary=""
            android:key="defaultUnit"
            android:defaultValue="0"
            android:entries="@array/fonte_units"
            android:entryValues="@array/fonte_unitsEnum" />
    </PreferenceCategory>
    <PreferenceCategory android:title="@string/pref_ui_setting">
        <CheckBoxPreference
            android:defaultValue="false"
            android:key="prefShowMP3"
            android:summary="@string/pref_show_mp3_player_summary"

android:title="@string/pref_show_mp3_player"></CheckBoxPreference>
    </PreferenceCategory>
</PreferenceScreen>
```

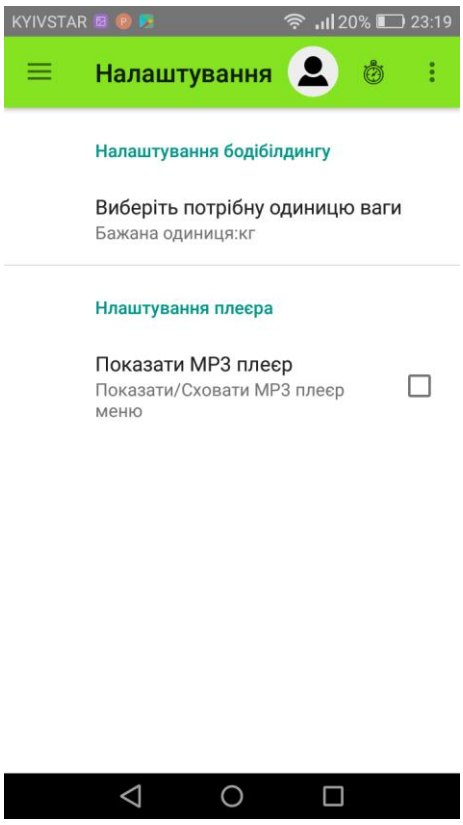


Рисунок 3.16 – меню «Налаштування»

Рисунок 3.17 – вспливаюче меню mp3 плеєра

Останній пункт меню «Про додаток» в ньому міститься інформація про версію додатку, як зв'язатись з розробником, історія обновлень (в якому буде вказуватись, які зміни були добавлені) та використані бібліотеки, як показано на рис. 3.18. Фрагмент коду:

```
<!-- App Name -->
<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_weight="50"
        android:text="@string/app_name" />
    <TextView
        android:id="@+id/app_version_textview"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_weight="50"
        android:text="@string/app_version" />
</LinearLayout>
<!-- Database Version -->
<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_weight="50"
        android:text="@string/database_version_label" />
    <TextView
        android:id="@+id/database_version"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_weight="50"
        android:text="1" />
</LinearLayout>
<!-- Author Name -->
<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_weight="50"
        android:text="@string/author_label" />
```

```
    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_weight="50"
        android:text="@string/author" />
</LinearLayout>
```

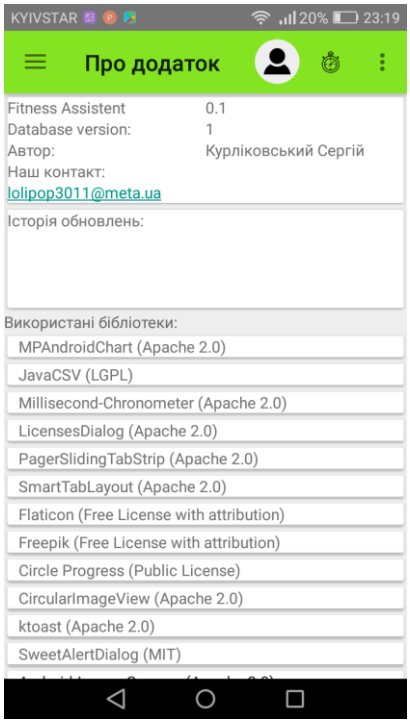


Рисунок 3.18 – Пунк меню «Про додаток»

В додатку є реалізований секундомір (рис. 3.19) викликається він при натисканні на піктограму годинника в TabBar, activity секундоміра з'являється поверх інших вікон. Фрагмент коду (повний код дивіться в додатку Е):

```
<gr.antoniom.chronometer.Chronometer
     android:id="@+id/chronoValue"
     android:layout_width="fill_parent"
     android:layout_height="wrap_content"
     android:gravity="center"
     android:keepScreenOn="true"
     android:text="00:00"
     android:textSize="@dimen/abc_text_size_display_3_material" />
<LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:background="#20BB30"
    android:orientation="horizontal" >
```

```
    <Button
        android:id="@+id/btn_startstop"
        android:layout_width="70dp"
        android:layout_height="30dp"
        android:layout_margin="10dp"
        android:background="@android:color/white"
        android:clickable="true"
        android:text="@string/start"
        android:textColor="#3FAA22"
        android:textStyle="bold" />
```

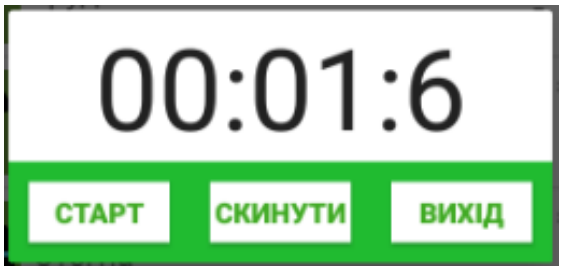


Рисунок 3.19 – Секундомір

Присутня сторінка профіля, на якій можна змінити ім’я, дату народження, ріст та фото (рис. 3.20). Фрагмент коду сторінки профілю (повний код в додатку Є):

```
<LinearLayout
    android:id="@+id/mainProfile"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"
    android:layout_marginBottom="50dp"
    android:focusable="true"
    android:focusableInTouchMode="true"
    android:orientation="vertical"
    android:paddingBottom="0dp"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp">
    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content">
        <com.mikhaellopez.circularimageview.CircularImageView
            android:id="@+id/photo"
            android:layout_width="300dp"
            android:layout_height="300dp"
            android:layout_centerHorizontal="true"
            android:layout_centerVertical="true"
            android:layout_marginBottom="10dp"
```

```
            android:layout_marginLeft="10dp"
            android:layout_marginRight="10dp"
            android:layout_marginTop="20dp"
            android:hapticFeedbackEnabled="false"
            android:isScrollContainer="false"
            android:maxHeight="200dp"
            android:minHeight="60dp"
            android:scaleType="centerCrop"
            android:src="@drawable/ic_profile_white"
            android:textAlignment="center"
            app:civ_border_color="#EEEEEE"
            app:civ_border_width="0dp"
            app:civ_shadow="false"
            app:civ_shadow_color="#8BC34A"
            app:civ_shadow_radius="0" />
    </RelativeLayout>
```

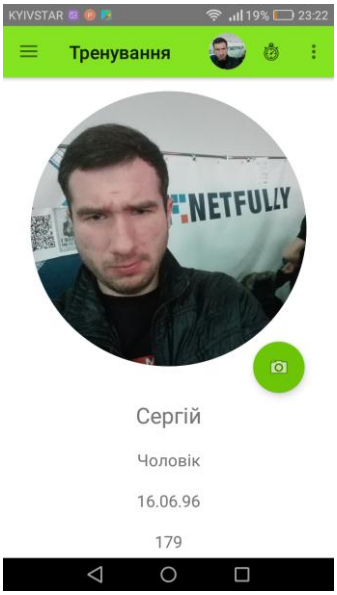


Рисунок 3.20 – Сторінка профілю

3.2 Встановлення, сумісність та тестування додатку

Додаток знаходиться на етапі бета-тестування. Встановлення відбувається шляхом скачування APK-файлу, який необхідно завантажити в пам'ять пристрою і використовуючи файловий менеджер, відкрити цей файл для встановлення. По завершенні етапу бета-тестування, будуть виявлені помилки програми та враховані побажання користувачів. В результаті буде випущена перша стабільна

версія програми, яка буде завантажена в магазин додатків Play Маркет, в подальшому встановлення буде відбуватись звідти.

Додаток підтримує мінімальну версію Android 5.0 (API 21) та всі подальші версії, максимальну на даний момент Android 9 (API 28). Додаток добре оптимізований для всіх версій.

Тестування сервісу відбувалося в два етапи. Перший етап полягав в тестуванні додатку на різних версіях Android (6.0-Marshmallow-API 23 та 8.1-Oreo-API 27), які мають різну роздільну здатність екранів. Необхідність такого тестування полягає в тому, щоб виявити і виправити помилки, допущені під час розробки. Другий етап тестування необхідний для опрацювання всіх можливих сценаріїв дій користувача, для того, щоб переконатися, що все працює правильно і ніяких помилок в проектуванні не допущено.

ВИСНОВКИ

При виконанні дипломної роботи були виявлені недоліки додатків-аналогів, головними з яких є статичні вправи, де немає можливостей додавати свій список занять. Також не було історії пророблених вправ (занять). Тому ці всі нюанси було включено в розроблений додаток для покращення його роботи.

При виборі операційної системи було проаналізовано ринок і зроблено висновок, що Android найбільш прогресивна ОС і нею користується більший відсоток людей. Середовищем розробки було обрано Android Studio, тому що на неї є детальні описи і велика документація. Java була вибрана мовою розробки, тому що Google зробили її офіційною мовою під Android.

Побудова структури помогля осягнути приблизний масштаб проекту та його логічні складові.

Було розроблено, метод ІМТ (який дозволяє визначити кількість жирової тканини в організмі та допомагає оцінити ризик виникнення хвороб, пов’язаних із надмірною вагою) конструктор вправ, пошук за допомогою якого можна швидко знайти бажану вправу, створення записів про заміри певних частин тіла, графік та історію записів тренувань.

Додаток розроблений під версію Android 5.0 (API 21), яка охопить понад 90% ринку користувачів ОС Android.

За результатами виконаних теоретичних та практичних досліджень розроблено мобільний додаток під назвою «Fitness Assistant».

ПЕРЕЛІК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Актуальність мобільних пристроїв. URL: https://analitic.ub.ua/24446-rozrobka-mobilnih-dodatkiv-aktualnist-pitannya.html (Дата звернення 21.11.2018)
2. Мобільні застосунки як засіб комунікації в суспільстві знань: освітній аспект. URL: http://nbuviap.gov.ua/index.php?option=com_content&view=article&id=3552:mobilni-zastosunki-yak-zasib-komunikatsiji-v-suspilstvi-znan-osvitnij-aspekt&catid=81&Itemid=415 (Дата звернення 22.11.2018)
3. Вплив спорту на здоров'я людини. URL: http://sport-kosa.ru/zdorove/regulyarnye-trenirovki-eto-chast-zdorovogo-obraza-zhizni.html
4. Степ-аеробіка – уроки для початківців в домашніх умовах. URL: https://www.kleo.ru/items/zdorovie/step_aerobika.shtml
5. Аквааеробіка: поняття, користь, показання. URL: http://plavaem.info/akvaaerobika.php
6. Співвідношення мобільних ОС на ринку: URL: http://w7phone.ru/rynok-mobilnyx-os-statistika-za-sentyabr-2017-141927/
7. Market Share Statistics for Internet Technologies. URL: https://netmarketshare.com/
8. Тереза Нейл. Мобильная разработка. Галерея шаблонов: монографія. Санкт-Петербург: Питер, 2012 р. 208 с.
9. Вандад Нахавандипур, iOS. Примеры программирования: монографія. Санкт-Петербург: Питер, 2014 р. 830 с.
10. Голощапов А. Л., Google Android. Программирование для мобильных устройств: монографія. Санкт-Петербург: БХВ-Петербург, 2013 р. 832 с.
11. Android. Матеріал з Вікіпедії – вільної енциклопедії. URL: https://uk.wikipedia.org/wiki/Android

12. Eclipse. The Platform for Open, innovation and collaboration. URL: https://www.eclipse.org/
13. Особливості android studio. URL: http://developer.alexanderklimov.ru/android/studio/androidstudio.php
14. Android Studio. Матеріал з Вікіпедії – вільної енциклопедії. URL: https://uk.wikipedia.org/wiki/Android_Studio
15. Activity. URL: https://developer.android.com/reference/android/app/Activity
16. Постачальники контенту (Content Provider). URL: https://developer.android.com/guide/topics/providers/content-providers?hl=RU
17. Життєвий цикл додатка. URL: https://metanit.com/java/android/2.1.php
18. Життєвий цикл активності Android – для чого всі ці методи? URL: http://qaru.site/questions/19641/android-activity-life-cycle-what-are-all-these-methods-for
19. Свіжа статистика версій Android: Pie встановлений на 10% пристроїв. URL: https://rozetked.me/news/5973-svezhaya-statistika-versiy-android-pie-ustanovlen-na-10-ustroystv
20. На чому пишуть програми під Android. URL: https://livetyping.com/ru/blog/na-chem-pishut-prilozhenija-pod-android
21. Python для Android: Як почати робити Кросплатформені додатки з Kivy. URL: https://medium.com/nuances-of-programming/python
22. Створення нативних Android-додатків з використанням компілятора Intel C ++ Compiler в Android Studio. Матеріал з habr. URL: https://habr.com/ru/company/intel/blog/260003/
23. Пишемо на Kotlin під Android. URL: Матеріал з habr. URL: https://habr.com/ru/company/JetBrains/blog/231525/
24. Java vs. Kotlin для Android. DOU: https://dou.ua/lenta/articles/java-vs-kotlin/
25. Stackoverflow – сайт питань і відповідей для програмістів. URL: https://ru.stackoverflow.com/
26. Хантер Д., Рафтер Дж., Фаусетт Дж. XML. Базовый курс: монографія. Москва: Диалектика-Вильямс, 2018 р. 1344 с.

27. Кох Дж., Дэвидсон К. XML. Огромные возможности и легкость изучения монографія. Москва: НТ Пресс, 2007 р. 256 с.
28. Брюс Эккель. Философия Java. 4-е издание, полное : монографія. Санкт-Петербург: Питер, 2019 р. 1168 с.
29. Бен Форта. SQL за 10 минут. Издательство: монографія. Москва: Диалектика Вильямс, 2014 р. 288 с.
30. Поняття зворотніх викликів життєвого циклу. URL: http://mikrotik.kpi.ua/index.php/courses-list/android/52-android-starting-an-activity
31. SQLite. Матеріал з Вікіпедії – вільної енциклопедії. URL: https://uk.wikipedia.org/wiki/SQLite

Додаток А

Файл DAOBase.java

```
import android.content.Context;
import android.database.sqlite.SQLiteDatabase;

public class DAOBase {

   private SQLiteDatabase database;
   private DatabaseHelper dbHelper;

   public DAOBase(Context context) {
      dbHelper = DatabaseHelper.getInstance(context);
   }

   public SQLiteDatabase open() {
      return database = dbHelper.getWritableDatabase();
   }

   public SQLiteDatabase getWritableDatabase() {
      return database = dbHelper.getWritableDatabase();
   }

   public SQLiteDatabase getReadableDatabase() {
      return database = dbHelper.getReadableDatabase();
   }

   public void close() {
      dbHelper.close();
   }
}
```

Файл DatabaseHelper.java

```
import android.app.Activity;
import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteException;
import android.database.sqlite.SQLiteOpenHelper;

import com.easyfitness.DAO.bodymeasures.BodyPart;
import com.easyfitness.DAO.bodymeasures.DAOBodyMeasure;

import java.io.File;
```

```
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.util.TimeZone;

public class DatabaseHelper extends SQLiteOpenHelper {

   private static DatabaseHelper sInstance;

    public static final int DATABASE_VERSION = 1;
   public static final String OLD09_DATABASE_NAME =
"easyfitness";
   public static final String DATABASE_NAME = "easyfitness.db";
   private Context mContext = null;

   public DatabaseHelper(Context context) {
      super(context, DATABASE_NAME, null, DATABASE_VERSION);
      mContext = context;
   }

   @Override
   public void onCreate(SQLiteDatabase db) {
        db.execSQL(DAORecord.TABLE_CREATE);
db.execSQL(DAOProfil.TABLE_CREATE);
      db.execSQL(DAOWeight.TABLE_CREATE);
      db.execSQL(DAOMachine.TABLE_CREATE);
      db.execSQL(DAOBodyMeasure.TABLE_CREATE);
   }

   @Override
    public void onUpgrade(
            final SQLiteDatabase db, final int oldVersion,
            final int newVersion)
        {
            int upgradeTo = oldVersion + 1;
            while (upgradeTo <= newVersion)
            {
                switch (upgradeTo)
                {
                   case 1:
                      db.execSQL(DAOCardio.TABLE_CREATE);
                    break;
                    case 2:
                      db.execSQL(DAOCardio.TABLE_CREATE);
                        break;
                    case 3:
                       db.execSQL(DAOCardio.TABLE_DROP);
                       db.execSQL(DAOCardio.TABLE_CREATE);
                       break;
                    case 4:
                       db.execSQL("ALTER TABLE "+
```

```
DAOFonte.TABLE_NAME + " ADD COLUMN " + DAOFonte.NOTES + "
TEXT");
                        db.execSQL("ALTER TABLE "+
DAOFonte.TABLE_NAME + " ADD COLUMN " + DAOFonte.UNIT + " INTEGER
DEFAULT 0");
                        break;
                     case 5:
                        db.execSQL(DAOMachine.TABLE_CREATE_5);
                        db.execSQL("ALTER TABLE "+
DAOFonte.TABLE_NAME + " ADD COLUMN " + DAOFonte.MACHINE_KEY + "
INTEGER");
                        break;
                     case 6:
                        if (!isFieldExist(db,
DAOMachine.TABLE_NAME, DAOMachine.BODYPARTS))
: Probleme d'upgrade
                           db.execSQL("ALTER TABLE "+
DAOMachine.TABLE_NAME + " ADD COLUMN " + DAOMachine.BODYPARTS +
" TEXT");
                        break;
                     case 7:
db.execSQL("ALTER TABLE "+ DAOMachine.TABLE_NAME + " ADD COLUMN
" + DAOMachine.PICTURE + " TEXT");
                        break;
                case 8:
                   db.execSQL("ALTER TABLE " +
DAOFonte.TABLE_NAME + " ADD COLUMN " + DAOFonte.TIME + " TEXT");
                   break;
                case 9:
                   db.execSQL(DAOBodyMeasure.TABLE_CREATE);
                   break;
                case 10:
                   db.execSQL("ALTER TABLE " +
DAOMachine.TABLE_NAME + " ADD COLUMN " + DAOMachine.FAVORITES +
" INTEGER");
                   break;
                case 11:
                   db.execSQL("ALTER TABLE " +
DAOFonte.TABLE_NAME + " RENAME TO tmp_table_name");
                   db.execSQL(DAOFonte.TABLE_CREATE);
                   db.execSQL("INSERT INTO " +
DAOFonte.TABLE_NAME + " SELECT * FROM tmp_table_name");
                   break;
                case 12:
                     db.execSQL("DROP TABLE IF EXISTS
tmp_table_name");
                   break;
                case 13:
                   db.execSQL("ALTER TABLE " +
DAOProfil.TABLE_NAME + " ADD COLUMN " + DAOProfil.SIZE + "
INTEGER");
                   db.execSQL("ALTER TABLE " +
DAOProfil.TABLE_NAME + " ADD COLUMN " + DAOProfil.BIRTHDAY + "
```

```
DATE");
                    break;
                      case 14:
                          db.execSQL("ALTER TABLE " +
DAOProfil.TABLE_NAME + " ADD COLUMN " + DAOProfil.PHOTO + "
TEXT");
                     break;
                      case 15:

                          db.execSQL("ALTER TABLE " +
DAORecord.TABLE_NAME + " ADD COLUMN " + DAORecord.DISTANCE + "
REAL");
                          db.execSQL("ALTER TABLE " +
DAORecord.TABLE_NAME + " ADD COLUMN " + DAORecord.DURATION + "
INTEGER");
                          db.execSQL("ALTER TABLE " +
DAORecord.TABLE_NAME + " ADD COLUMN " + DAORecord.TYPE + "
INTEGER DEFAULT " + DAOMachine.TYPE_FONTE);
                          break;
                      case 16:
                          db.execSQL("ALTER TABLE " +
DAOBodyMeasure.TABLE_NAME + " ADD COLUMN " + DAOBodyMeasure.UNIT
+ " INTEGER");
                          migrateWeightTable(db, mContext);
                          break;
                      case 17:
                          db.execSQL("ALTER TABLE " +
DAOProfil.TABLE_NAME + " ADD COLUMN " + DAOProfil.GENDER + "
INTEGER");
                      break;
              }
                  upgradeTo++;
              }
          }


    @Override
     public void onDowngrade(
              final SQLiteDatabase db, final int oldVersion,
              final int newVersion)
          {
              int upgradeTo = oldVersion - 1;
              while (upgradeTo >= newVersion)
              {
                  switch (upgradeTo)
                  {
                      case 2:
                          break;
                      case 3:
DAOFonte.TABLE_NAME + " DROP COLUMN " + DAOFonte.NOTES);
DAOFonte.TABLE_NAME + " DROP COLUMN " + DAOFonte.UNIT);
                         break;
                      case 4:
```

```
db.execSQL("ALTER TABLE "+ DAOFonte.TABLE_NAME + " DROP COLUMN "
+ DAOFonte.MACHINE_KEY );
                        break;
                    case 5:
db.execSQL("ALTER TABLE "+ DAOMachine.TABLE_NAME + " DROP COLUMN
" + DAOMachine.BODYPARTS );
                        break;
                }
                upgradeTo--;
            }
        }

   public static DatabaseHelper getInstance(Context context) {
            if (sInstance == null) {
         sInstance = new
DatabaseHelper(context.getApplicationContext());
      }
      return sInstance;
   }

      public boolean isFieldExist(SQLiteDatabase db, String
tableName, String fieldName)
   {
       boolean isExist = true;
       Cursor res;

      try {
         res = db.rawQuery("SELECT " + fieldName + " FROM " +
tableName,null);
            res.close();
      } catch (SQLiteException e) {
         isExist = false;
      }

        return isExist;
    }

    public boolean tableExists(SQLiteDatabase db, String
tableName) {
        boolean isExist = true;
        Cursor res;

        try {
            res = db.rawQuery("SELECT * FROM " + tableName,
null);
            res.close();
        } catch (SQLiteException e) {
            isExist = false;
        }
        return isExist;
    }

   public static void renameOldDatabase(Activity activity)
```

```
   {
       File oldDatabaseFile =
activity.getDatabasePath(OLD09_DATABASE_NAME);
       if ( oldDatabaseFile.exists() ) {
          File newDatabaseFile = new
File(oldDatabaseFile.getParentFile(), DATABASE_NAME);
          oldDatabaseFile.renameTo(newDatabaseFile);
       }
   }

    private void migrateWeightTable(SQLiteDatabase db, Context
context) {
        List<ProfileWeight> valueList = new
ArrayList<ProfileWeight>();
        String selectQuery = "SELECT * FROM " +
DAOWeight.TABLE_NAME;
        ;
        SQLiteDatabase db = this.getWritableDatabase();
        Cursor mCursor = null;
        mCursor = db.rawQuery(selectQuery, null);

        if (mCursor.moveToFirst()) {
            do {
                ContentValues value = new ContentValues();

                value.put(DAOBodyMeasure.DATE,
mCursor.getString(mCursor.getColumnIndex(DAOWeight.DATE)));
                value.put(DAOBodyMeasure.BODYPART_KEY,
BodyPart.WEIGHT);
                value.put(DAOBodyMeasure.MEASURE,
mCursor.getFloat(mCursor.getColumnIndex(DAOWeight.POIDS)));
                value.put(DAOBodyMeasure.PROFIL_KEY,
mCursor.getLong(mCursor.getColumnIndex(DAOWeight.PROFIL_KEY)));

                db.insert(DAOBodyMeasure.TABLE_NAME, null,
value);

            } while (mCursor.moveToNext());
            mCursor.close();
            db.close();
        }

    }
}
```

Додаток Б

Файл DAOProfil.java

```
import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;

import com.easyfitness.DAO.bodymeasures.BodyMeasure;
import com.easyfitness.DAO.bodymeasures.DAOBodyMeasure;
import com.easyfitness.utils.DateConverter;

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

public class DAOProfil extends DAOBase {
    public static final String TABLE_NAME = "EFprofil";

    public static final String KEY = "_id";
    public static final String NAME = "name";
    public static final String CREATIONDATE = "creationdate";
    public static final String SIZE = "size";
    public static final String BIRTHDAY = "birthday";
    public static final String PHOTO = "photo";
    public static final String GENDER = "gender";

    public static final String TABLE_CREATE = "CREATE TABLE " +
TABLE_NAME + " (" + KEY + " INTEGER PRIMARY KEY AUTOINCREMENT, "
+ CREATIONDATE + " DATE, " + NAME + " TEXT, " + SIZE + "
INTEGER, " + BIRTHDAY + " DATE, " + PHOTO + " TEXT, " + GENDER +
" INTEGER);";

    public static final String TABLE_DROP = "DROP TABLE IF
EXISTS " + TABLE_NAME + ";";

    private Cursor mCursor = null;

    public DAOProfil(Context context) {
        super(context);
    }

    public void addProfil(Profile m) {
        Profile check = getProfil(m.getName());
        if (check != null) return;

        SQLiteDatabase db = this.getWritableDatabase();
```

```
        ContentValues value = new ContentValues();

        value.put(DAOProfil.CREATIONDATE,
DateConverter.dateToDBDateStr(new Date()));
        value.put(DAOProfil.NAME, m.getName());
        value.put(DAOProfil.BIRTHDAY,
DateConverter.dateToDBDateStr(m.getBirthday()));
        value.put(DAOProfil.SIZE, m.getSize());
        value.put(DAOProfil.PHOTO, m.getPhoto());
        value.put(DAOProfil.GENDER, m.getGender());

        db.insert(DAOProfil.TABLE_NAME, null, value);

        close();
    }


    public void addProfil(String pName) {
        Profile check = getProfil(pName);
        if (check != null) return;

        SQLiteDatabase db = this.getWritableDatabase();

        ContentValues value = new ContentValues();

        value.put(DAOProfil.CREATIONDATE,
DateConverter.dateToDBDateStr(new Date()));
        value.put(DAOProfil.NAME, pName);
        value.put(DAOProfil.BIRTHDAY,
DateConverter.dateToDBDateStr(m.getBirthday()));
        value.put(DAOProfil.SIZE, 0);
        db.insert(DAOProfil.TABLE_NAME, null, value);

        close();
    }

    public Profile getProfil(long id) {
        SQLiteDatabase db = this.getReadableDatabase();
        if (mCursor != null) mCursor.close();
        mCursor = null;
        mCursor = db.query(TABLE_NAME,
                new String[]{KEY, CREATIONDATE, NAME, SIZE,
BIRTHDAY, PHOTO, GENDER},
                KEY + "=?",
                new String[]{String.valueOf(id)},
                null, null, null, null);
        if (mCursor != null && mCursor.getCount() > 0) {
            mCursor.moveToFirst();

            Profile value = new
Profile(mCursor.getLong(mCursor.getColumnIndex(DAOProfil.KEY)),
```

```
DateConverter.DBDateStrToDate(mCursor.getString(mCursor.getColum
nIndex(DAOProfil.CREATIONDATE))),

mCursor.getString(mCursor.getColumnIndex(DAOProfil.NAME)),

mCursor.getInt(mCursor.getColumnIndex(DAOProfil.SIZE)),

mCursor.getString(mCursor.getColumnIndex(DAOProfil.BIRTHDAY))

!= null ?
DateConverter.DBDateStrToDate(mCursor.getString(mCursor.getColum
nIndex(DAOProfil.BIRTHDAY))) : new Date(0),

mCursor.getString(mCursor.getColumnIndex(DAOProfil.PHOTO)),

mCursor.getInt(mCursor.getColumnIndex(DAOProfil.GENDER))
            );
            mCursor.close();
            close();
            return value;
        } else {
            mCursor.close();
            close();
            return null;
        }

    }


    public Profile getProfil(String name) {
        SQLiteDatabase db = this.getReadableDatabase();
        if (mCursor != null) mCursor.close();
        mCursor = null;
        mCursor = db.query(TABLE_NAME,
                new String[]{KEY, CREATIONDATE, NAME, SIZE,
BIRTHDAY, PHOTO, GENDER},
                NAME + "=?",
                new String[]{name},
                null, null, null, null);
        if (mCursor != null && mCursor.getCount() > 0) {
            mCursor.moveToFirst();

            Profile value = new
Profile(mCursor.getLong(mCursor.getColumnIndex(DAOProfil.KEY)),

DateConverter.DBDateStrToDate(mCursor.getString(mCursor.getColum
nIndex(DAOProfil.CREATIONDATE))),

mCursor.getString(mCursor.getColumnIndex(DAOProfil.NAME)),

mCursor.getInt(mCursor.getColumnIndex(DAOProfil.SIZE)),

mCursor.getString(mCursor.getColumnIndex(DAOProfil.BIRTHDAY)) !=
```

```
null ?
DateConverter.DBDateStrToDate(mCursor.getString(mCursor.getColum
nIndex(DAOProfil.BIRTHDAY))) : new Date(0),

mCursor.getString(mCursor.getColumnIndex(DAOProfil.PHOTO)),

mCursor.getInt(mCursor.getColumnIndex(DAOProfil.GENDER))
            );

            mCursor.close();
            close();
            return value;
        } else {
            close();
            return null;
        }

    }

    public List<Profile> getProfilsList(String pRequest) {
        List<Profile> valueList = new ArrayList<Profile>();

        SQLiteDatabase db = this.getReadableDatabase();
        mCursor = null;
        mCursor = db.rawQuery(pRequest, null);

        if (mCursor.moveToFirst()) {
            do {
                Profile value = new
Profile(mCursor.getLong(mCursor.getColumnIndex(DAOProfil.KEY)),

DateConverter.DBDateStrToDate(mCursor.getString(mCursor.getColum
nIndex(DAOProfil.CREATIONDATE))),

mCursor.getString(mCursor.getColumnIndex(DAOProfil.NAME)),

mCursor.getInt(mCursor.getColumnIndex(DAOProfil.SIZE)),

mCursor.getString(mCursor.getColumnIndex(DAOProfil.BIRTHDAY)) !=
null ?
DateConverter.DBDateStrToDate(mCursor.getString(mCursor.getColum
nIndex(DAOProfil.BIRTHDAY))) : new Date(0),

mCursor.getString(mCursor.getColumnIndex(DAOProfil.PHOTO)),

mCursor.getInt(mCursor.getColumnIndex(DAOProfil.GENDER))
                );

                valueList.add(value);
            } while (mCursor.moveToNext());
        }

        close();
```

```
        return valueList;
    }

    public Cursor GetCursor() {
        return mCursor;
    }

    public List<Profile> getAllProfils() {

        String selectQuery = "SELECT  * FROM " + TABLE_NAME + "
ORDER BY " + KEY + " DESC";

        return getProfilsList(selectQuery);
    }


    public List<Profile> getTop10Profils() {
        String selectQuery = "SELECT TOP 10 * FROM " +
TABLE_NAME + " ORDER BY " + KEY + " DESC";

        return getProfilsList(selectQuery);
    }

    public String[] getAllProfil() {

        SQLiteDatabase db = this.getReadableDatabase();
        mCursor = null;

        String selectQuery = "SELECT DISTINCT  " + NAME + " FROM
" + TABLE_NAME + " ORDER BY " + NAME + " ASC";
        mCursor = db.rawQuery(selectQuery, null);

        int size = mCursor.getCount();

        String[] valueList = new String[size];

        if (mCursor.moveToFirst()) {
            int i = 0;
            do {
                String value = mCursor.getString(0);
                valueList[i] = value;
                i++;
            } while (mCursor.moveToNext());
        }

        close();

        return valueList;
    }

    public Profile getLastProfil() {

        SQLiteDatabase db = this.getReadableDatabase();
```

```
        mCursor = null;

        String selectQuery = "SELECT MAX(" + KEY + ") FROM " +
TABLE_NAME;
        mCursor = db.rawQuery(selectQuery, null);
        mCursor.moveToFirst();
        long value = Long.parseLong(mCursor.getString(0));

        Profile prof = this.getProfil(value);
        mCursor.close();
        close();

        return prof;
    }
    public int updateProfile(Profile m) {
        SQLiteDatabase db = this.getWritableDatabase();

        ContentValues value = new ContentValues();
        value.put(DAOProfil.CREATIONDATE,
DateConverter.dateToDBDateStr(m.getCreationDate()));
        value.put(DAOProfil.NAME, m.getName());
        value.put(DAOProfil.BIRTHDAY,
DateConverter.dateToDBDateStr(m.getBirthday()));
        value.put(DAOProfil.SIZE, m.getSize());
        value.put(DAOProfil.PHOTO, m.getPhoto());
        value.put(DAOProfil.GENDER, m.getGender());
        return db.update(TABLE_NAME, value, KEY + " = ?",
                new String[]{String.valueOf(m.getId())});
    }
    public void deleteProfil(Profile m) {
        deleteProfil(m.getId());
    }
    public void deleteProfil(long id) {
        open();
        DAOWeight mWeightDb;
        mWeightDb = new DAOWeight(null);
List<ProfileWeight> valueList =
mWeightDb.getWeightList(getProfil(id));
        for (int i = 0; i < valueList.size(); i++) {
            mWeightDb.deleteMeasure(valueList.get(i).getId());
        }
        DAOBodyMeasure mBodyDb;
        mBodyDb = new DAOBodyMeasure(null);
List<BodyMeasure> bodyMeasuresList =
mBodyDb.getBodyMeasuresList(getProfil(id));
        for (int i = 0; i < bodyMeasuresList.size(); i++) {

mBodyDb.deleteMeasure(bodyMeasuresList.get(i).getId());
        }
        SQLiteDatabase db = this.getWritableDatabase();
        db.delete(TABLE_NAME, KEY + " = ?",
                new String[]{String.valueOf(id)});
```

```
        close();
    }

    public int getCount() {
        String countQuery = "SELECT  * FROM " + TABLE_NAME;
        open();
        SQLiteDatabase db = this.getReadableDatabase();
        Cursor cursor = db.rawQuery(countQuery, null);

        int value = cursor.getCount();
        cursor.close();
        close();
        return value;
    }

    public void populate() {
        Date date = new Date();
        Date dateBirthday = DateConverter.getNewDate();
        Profile m = new Profile(0, date, "Champignon", 120,
dateBirthday, null, 0);
        this.addProfil(m);
        m = new Profile(0, date, "Musclor", 150, dateBirthday,
null, 0);
        this.addProfil(m);
    }
}
```

Додаток В

Файл machine_details.xml

```
<android.support.design.widget.CoordinatorLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/tab_machine_details"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <android.support.v4.widget.NestedScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:paddingBottom="0dp"
        android:paddingLeft="0dp"
        android:paddingRight="0dp"
        android:paddingTop="0dp"

app:layout_behavior="@string/appbar_scrolling_view_behavior">

        <LinearLayout
            android:id="@+id/machine_photo_layout"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:focusable="true"
            android:focusableInTouchMode="true"
            android:orientation="vertical"
            android:paddingBottom="0dp"
            android:paddingLeft="0dp"
            android:paddingRight="0dp"
            android:paddingTop="0dp">

            <ImageView
                android:id="@+id/machine_photo"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:adjustViewBounds="true"
                android:background="@color/background_even"
                android:cropToPadding="false"
                android:maxHeight="200dp"
                android:minHeight="60dp"
                android:scaleType="centerCrop"
                android:src="@drawable/ic_machine" />


            <LinearLayout
                android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:orientation="vertical"
```

```
                android:padding="6dip">

                <LinearLayout

android:id="@+id/exerciseTypeSelectionLayout"
                    android:layout_width="match_parent"
                    android:layout_height="match_parent"
                    android:baselineAligned="false"
                    android:orientation="horizontal"
                    android:visibility="visible">

                    <android.support.v7.widget.CardView
                        style="@style/CardViewStyle"
                        android:layout_width="0dp"
                        android:layout_height="wrap_content"
                        android:layout_weight="40">

                        <TextView

android:id="@+id/bodyBuildingSelection"
                            android:layout_width="fill_parent"
                            android:layout_height="wrap_content"

android:background="@color/background_odd"
                            android:gravity="center"
                            android:text="@string/FonteLabel" />

                    </android.support.v7.widget.CardView>

                    <android.support.v7.widget.CardView
                        style="@style/CardViewStyle"
                        android:layout_width="0dp"
                        android:layout_height="wrap_content"
                        android:layout_weight="40">

                        <TextView
                            android:id="@+id/cardioSelection"
                            android:layout_width="fill_parent"
                            android:layout_height="wrap_content"
                            android:gravity="center"
                            android:text="@string/CardioLabel"
/>

                    </android.support.v7.widget.CardView>

                </LinearLayout>

                <TextView
                    android:id="@+id/machine_name_textview"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:gravity="center_vertical"
                    android:text="@string/name"
```

```
                    android:textSize="12sp" />

            </LinearLayout>
        </LinearLayout>


    </android.support.v4.widget.NestedScrollView>

    <android.support.design.widget.FloatingActionButton
        android:id="@+id/actionCamera"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="32dp"
        android:clickable="true"
        android:src="@android:drawable/ic_menu_camera"
        app:backgroundTint="@color/launcher_background"
        app:layout_anchor="@id/machine_photo"
        app:layout_anchorGravity="bottom|right|end" />
</android.support.design.widget.CoordinatorLayout>
```

Файл MachineDetailsFragment.java

```
import android.app.Activity;
import android.app.AlertDialog;
import android.content.Context;
import android.content.DialogInterface;
import.android.content.DialogInterface.OnMultiChoiceClickListene
r;
import android.content.Intent;
import android.graphics.BitmapFactory;
import android.net.Uri;
import android.os.Bundle;
import android.os.Environment;
import android.support.design.widget.FloatingActionButton;
import android.support.media.ExifInterface;
import android.support.v4.app.Fragment;
import android.text.Editable;
import android.text.TextWatcher;
import android.util.Log;
import android.view.Gravity;
import android.view.LayoutInflater;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.View.OnFocusChangeListener;
import android.view.View.OnLongClickListener;
import android.view.ViewGroup;
import android.view.ViewTreeObserver;
import android.widget.EditText;
import android.widget.ImageView;
```

```
import android.widget.LinearLayout;
import android.widget.Spinner;
import android.widget.TextView;

import com.easyfitness.BtnClickListener;
import com.easyfitness.DAO.DAOMachine;
import com.easyfitness.DAO.DAORecord;
import com.easyfitness.DAO.Machine;
import com.easyfitness.R;
import com.easyfitness.utils.ImageUtil;
import com.easyfitness.utils.RealPathUtil;
import com.onurkaganaldemir.ktoastlib.KToast;
import com.theartofdev.edmodo.cropper.CropImage;

import java.io.File;
import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;

import cn.pedant.SweetAlert.SweetAlertDialog;

public class MachineDetailsFragment extends Fragment {
   public final int
MY_PERMISSIONS_REQUEST_WRITE_EXTERNAL_STORAGE = 101;

   Spinner typeList = null;   EditText musclesList = null;
   EditText machineName = null;
   EditText machineDescription = null;
   ImageView machinePhoto = null;
   FloatingActionButton machineAction = null;

    LinearLayout machinePhotoLayout = null;

    LinearLayout exerciseTypeSelectorLayout = null;
    TextView bodybuildingSelector = null;
    TextView cardioSelector = null;
    int selectedType = DAOMachine.TYPE_FONTE;

   String machineNameArg = null;
   long machineIdArg = 0;
   long machineProfilIdArg = 0;

   boolean isImageFitToScreen = false;
   ExerciseDetailsPager pager =null;

   ArrayList<Integer> selectMuscleList=new ArrayList();

   // http://labs.makemachine.net/2010/03/android-multi-
selection-dialogs/
   protected CharSequence[] _muscles = { "Biceps", "Triceps",
"Epaules", "Pectoraux", "Dorseaux", "Quadriceps", "Adducteurs",
"Uranus", "Neptune", "Neptune" };
```

```
   protected boolean[] _selections =  new boolean[
_muscles.length ];
   DAOMachine mDbMachine = null;
   DAORecord mDbRecord = null;
   Machine mMachine;

   View fragmentView = null;

   ImageUtil imgUtil = null;
   private boolean toBeSaved;


   public static MachineDetailsFragment newInstance(long
machineId, long machineProfile) {
      MachineDetailsFragment f = new MachineDetailsFragment();
      Bundle args = new Bundle();
      args.putLong("machineID", machineId);
      args.putLong("machineProfile", machineProfile);
      f.setArguments(args);

      return f;
   }

   @Override
   public View onCreateView(LayoutInflater inflater, ViewGroup
container,
         Bundle savedInstanceState) {

            View view =
inflater.inflate(R.layout.machine_details, container, false);
      fragmentView = view;


      mDbMachine = new DAOMachine(view.getContext());
      mDbRecord = new DAORecord(view.getContext());

      machineName = view.findViewById(R.id.machine_name);
      machineDescription =
view.findViewById(R.id.machine_description);
      musclesList = view.findViewById(R.id.machine_muscles);
      machinePhoto = view.findViewById(R.id.machine_photo);

        machinePhotoLayout =
view.findViewById(R.id.machine_photo_layout);
        bodybuildingSelector =
view.findViewById(R.id.bodyBuildingSelection);
        cardioSelector =
view.findViewById(R.id.cardioSelection);
        exerciseTypeSelectorLayout =
view.findViewById(R.id.exerciseTypeSelectionLayout);

      machineAction = view.findViewById(R.id.actionCamera);
```

```
        imgUtil = new ImageUtil(machinePhoto);

      buildMusclesTable();

        Bundle args = this.getArguments();

        machineIdArg = args.getLong("machineID");
        machineProfilIdArg = args.getLong("machineProfile");


      musclesList.setOnClickListener(onClickMusclesList);
      musclesList.setOnFocusChangeListener(onFocusMachineList);

machinePhoto.setOnLongClickListener(onLongClickMachinePhoto);
      machinePhoto.setOnClickListener(new View.OnClickListener()
{
            @Override
            public void onClick(View v) {
                if(isImageFitToScreen) {
                    isImageFitToScreen=false;
                    machinePhoto.setLayoutParams(new
LinearLayout.LayoutParams(LinearLayout.LayoutParams.MATCH_PARENT
, LinearLayout.LayoutParams.WRAP_CONTENT));
                    machinePhoto.setAdjustViewBounds(true);

machinePhoto.setMaxHeight((int)(getView().getHeight()*0.2));

machinePhoto.setScaleType(ImageView.ScaleType.CENTER_CROP);
                }else{
                   if (mCurrentPhotoPath != null &&
!mCurrentPhotoPath.isEmpty()) {
                        File f = new File(mCurrentPhotoPath);
                        if(f.exists()) {

                         isImageFitToScreen=true;


                       BitmapFactory.Options bmOptions = new
BitmapFactory.Options();
                       bmOptions.inJustDecodeBounds = true;

BitmapFactory.decodeFile(mCurrentPhotoPath, bmOptions);
                       float photoW = bmOptions.outWidth;
                       float photoH = bmOptions.outHeight;

                                                          int
scaleFactor = (int) (photoW (machinePhoto.getWidth()));
Math.min(photoW/targetW,
photoH/targetH);machinePhoto.setLayoutParams(new
LinearLayout.LayoutParams(LinearLayout.LayoutParams.MATCH_PARENT
, LinearLayout.LayoutParams.WRAP_CONTENT));

machinePhoto.setAdjustViewBounds(true);
```

```
machinePhoto.setMaxHeight((int)(photoH/scaleFactor));

machinePhoto.setScaleType(ImageView.ScaleType.CENTER_INSIDE);
                        }
                    }
                }
            }
        });
      machineAction.setOnClickListener(onClickMachinePhoto);

        mMachine = mDbMachine.getMachine(machineIdArg);
        machineNameArg = mMachine.getName();

      if (machineNameArg.equals("")) {requestForSave();}

      machineName.setText(machineNameArg);
        machineDescription.setText(mMachine.getDescription());

musclesList.setText(this.getInputFromDBString(mMachine.getBodyPa
rts()));
        mCurrentPhotoPath = mMachine.getPicture();
        exerciseTypeSelectorLayout.setVisibility(View.GONE);

        if (mMachine.getType() == DAOMachine.TYPE_CARDIO) {

cardioSelector.setBackgroundColor(getResources().getColor(R.colo
r.background_odd));
            bodybuildingSelector.setVisibility(View.GONE);

bodybuildingSelector.setBackgroundColor(getResources().getColor(
R.color.background));
            selectedType = mMachine.getType();

view.findViewById(R.id.machine_muscles).setVisibility(View.GONE)
;

view.findViewById(R.id.machine_muscles_textview).setVisibility(V
iew.GONE);
        } else {

cardioSelector.setBackgroundColor(getResources().getColor(R.colo
r.background));
            cardioSelector.setVisibility(View.GONE);

bodybuildingSelector.setBackgroundColor(getResources().getColor(
R.color.background_odd));
            selectedType = mMachine.getType();
        }

       view.getViewTreeObserver().addOnGlobalLayoutListener(new
ViewTreeObserver.OnGlobalLayoutListener() {
```

```
            @Override
            public void onGlobalLayout() {

if(android.os.Build.VERSION.SDK_INT >=
android.os.Build.VERSION_CODES.JELLY_BEAN) {

fragmentView.getViewTreeObserver().removeOnGlobalLayoutListener(
this);
                }
                else {

fragmentView.getViewTreeObserver().removeGlobalOnLayoutListener(
this);
                }

              if (mCurrentPhotoPath != null &&
!mCurrentPhotoPath.isEmpty()) {
                  ImageUtil.setPic(machinePhoto,
mCurrentPhotoPath);
               } else {
                     if (mMachine.getType() ==
DAOMachine.TYPE_FONTE) {

imgUtil.getView().setImageDrawable(getActivity().getResources().
getDrawable(R.drawable.ic_machine));
                     } else {

imgUtil.getView().setImageDrawable(getActivity().getResources().
getDrawable(R.drawable.ic_running));
                     }

machinePhoto.setScaleType(ImageView.ScaleType.CENTER_INSIDE);
                }

machinePhoto.setMaxHeight((int)(getView().getHeight()*0.2));
}
        });

       machineName.addTextChangedListener(watcher);
       machineDescription.addTextChangedListener(watcher);
       musclesList.addTextChangedListener(watcher);

         imgUtil.setOnDeleteImageListener(new
ImageUtil.OnDeleteImageListener() {
             @Override
             public void onDeleteImage(ImageUtil imgUtil) {

imgUtil.getView().setImageDrawable(getActivity().getResources().
getDrawable(R.drawable.ic_machine));
                 mCurrentPhotoPath = null;
                 requestForSave();
             }
         });
```

```
        if ( getParentFragment() instanceof ExerciseDetailsPager
) {
            pager = (ExerciseDetailsPager)getParentFragment();
        };

        return view;
    }

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        //setHasOptionsMenu(true);
    }

    @Override
    public void onStart() {
        super.onStart();
    }

    boolean isCreateMuscleDialogActive = false;

    private BtnClickListener itemClickDeleteRecord = new
BtnClickListener() {
        @Override
        public void onBtnClick(long id) {
            showDeleteDialog(id);
        }
    };

    private void showDeleteDialog(final long idToDelete) {

        new SweetAlertDialog(getContext(),
SweetAlertDialog.WARNING_TYPE)

.setTitleText(getString(R.string.DeleteRecordDialog))

.setContentText(getResources().getText(R.string.areyousure).toSt
ring())

.setCancelText(getResources().getText(R.string.global_no).toStri
ng())

.setConfirmText(getResources().getText(R.string.global_yes).toSt
ring())
            .showCancelButton(true)
            .setConfirmClickListener(new
SweetAlertDialog.OnSweetClickListener() {
                @Override
                public void onClick(SweetAlertDialog sDialog) {
                    mDbRecord.deleteRecord(idToDelete);
                    KToast.infoToast(getActivity(),
getResources().getText(R.string.removedid).toString(),
```

```
Gravity.BOTTOM, KToast.LENGTH_LONG);
                   sDialog.dismissWithAnimation();
               }
            })
            .show();
   }


   private boolean CreateMuscleDialog()
   {
        if (isCreateMuscleDialogActive)
            return true;
      isCreateMuscleDialogActive = true;

      AlertDialog.Builder newProfilBuilder = new
AlertDialog.Builder(this.getActivity());


newProfilBuilder.setTitle(this.getResources().getString(R.string
.selectMuscles));
         newProfilBuilder.setMultiChoiceItems( _muscles,
_selections, new OnMultiChoiceClickListener() {

               @Override
               public void onClick(DialogInterface arg0, int
arg1, boolean arg2) {
                    if(arg2)
                    {

selectMuscleList.add(arg1);
                    }
                    else if (selectMuscleList.contains(arg1))
                    {
selectMuscleList.remove(Integer.valueOf(arg1));
                    }
               }
              });


newProfilBuilder.setPositiveButton(getResources().getString(R.st
ring.global_ok), new DialogInterface.OnClickListener() {
             public void onClick(DialogInterface dialog, int
whichButton) {
               String msg="";
               int i = 0;
               boolean firstSelection = true;
               ( selectMuscleList.size() > 0 ) { // Si on a au
moins selectionne un muscle
                    for (i = 0; i < _selections.length; i++) {
                       if (_selections[i] && firstSelection)  {
msg=_muscles[i].toString(); firstSelection = false;}
                    else if (_selections[i]&& !firstSelection)
{ msg=msg+";" +_muscles[i]; }
```

```
                    }
                }
                 setMuscleText(msg);
                 isCreateMuscleDialogActive = false;
            }
        });


newProfilBuilder.setNegativeButton(getResources().getString(R.st
ring.global_cancel), new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int
whichButton) {
                isCreateMuscleDialogActive = false;
            }
        });

        newProfilBuilder.show();

        return true;
    }

    listView.getItemAtPosition(position);

    private boolean CreatePhotoSourceDialog() {
        if (imgUtil==null)
            imgUtil = new ImageUtil();

        return imgUtil.CreatePhotoSourceDialog(this);
    }

    private OnClickListener onClickMusclesList = new
View.OnClickListener() {
        @Override
        public void onClick(View v) {
            CreateMuscleDialog();
        }
    };

    private OnLongClickListener onLongClickMachinePhoto = new
View.OnLongClickListener() {
        @Override
        public boolean onLongClick(View v) {
            return CreatePhotoSourceDialog();
        }
    };

    private OnClickListener onClickMachinePhoto = new
View.OnClickListener() {
        @Override
        public void onClick(View v) {
            CreatePhotoSourceDialog();
        }
    };
```

```
   String mCurrentPhotoPath = null;

   public void onActivityResult(int requestCode, int resultCode,
Intent data) {
      super.onActivityResult(requestCode, resultCode, data);

      switch (requestCode) {
         case ImageUtil.REQUEST_TAKE_PHOTO:
            if (resultCode == Activity.RESULT_OK) {
               mCurrentPhotoPath = imgUtil.getFilePath();
               imgUtil.setPic(machinePhoto, mCurrentPhotoPath);
               imgUtil.saveThumb(mCurrentPhotoPath);
               imgUtil.galleryAddPic(this, mCurrentPhotoPath);
               requestForSave();
            }
            break;
         case ImageUtil.REQUEST_PICK_GALERY_PHOTO:
            if (resultCode == Activity.RESULT_OK) {
               String realPath;
               realPath =
RealPathUtil.getRealPath(this.getContext(), data.getData());

               imgUtil.setPic(machinePhoto, realPath);
               imgUtil.saveThumb(realPath);
               mCurrentPhotoPath = realPath;
               requestForSave();
            }
            break;
            case CropImage.CROP_IMAGE_ACTIVITY_REQUEST_CODE:
                CropImage.ActivityResult result =
CropImage.getActivityResult(data);
                if (resultCode == Activity.RESULT_OK) {
                    Uri resultUri = result.getUri();
                    String realPath;
                    realPath =
RealPathUtil.getRealPath(this.getContext(), resultUri);

File SourceFile = new File(realPath);

                    File storageDir = null;
                    String timeStamp = new
SimpleDateFormat("yyyyMMdd_HHmmss").format(new Date());
                    String imageFileName = "JPEG_" + timeStamp +
".jpg";
                    String state =
Environment.getExternalStorageState();
                    if
(!Environment.MEDIA_MOUNTED.equals(state)) {
                        return;
                    } else {
                        //We use the FastNFitness directory for
saving our .csv file.
```

```
                        storageDir =
Environment.getExternalStoragePublicDirectory("/FastnFitness/Cam
era/");
                        if (!storageDir.exists()) {
                            storageDir.mkdirs();
                        }
                    }
                    File DestinationFile = null;

                    try {
                        DestinationFile =
imgUtil.moveFile(SourceFile, storageDir);
                        Log.v("Moving", "\n" +
                        "Переміщення файлу успішне.");
                        realPath = DestinationFile.getPath();
                    } catch (IOException e) {
                        e.printStackTrace();
                        Log.v("Moving", "\n" +
                        "Помилка переміщення файлу.");
                    }

                    imgUtil.setPic(machinePhoto, realPath);
                    imgUtil.saveThumb(realPath);
                    mCurrentPhotoPath = realPath;
                    requestForSave();
                } else if (resultCode ==
CropImage.CROP_IMAGE_ACTIVITY_RESULT_ERROR_CODE) {
                    Exception error = result.getError();
                }
                break;
        }
    }

   private OnFocusChangeListener onFocusMachineList = new
View.OnFocusChangeListener() {

        @Override
        public void onFocusChange(View arg0, boolean arg1) {
              if (arg1) {
              CreateMuscleDialog();
           }
        }
    };



     private OnClickListener clickExerciseTypeSelector = new
View.OnClickListener() {
         @Override
         public void onClick(View v) {
             switch (v.getId()) {
                 case R.id.cardioSelection:
```

```
cardioSelector.setBackgroundColor(getResources().getColor(R.colo
r.background_odd));

bodybuildingSelector.setBackgroundColor(getResources().getColor(
R.color.background));
                    selectedType = DAOMachine.TYPE_CARDIO;
                    break;
                case R.id.bodyBuildingSelection:
                default:

cardioSelector.setBackgroundColor(getResources().getColor(R.colo
r.background));

bodybuildingSelector.setBackgroundColor(getResources().getColor(
R.color.background_odd));
                    selectedType = DAOMachine.TYPE_FONTE;
                    break;
            }
        }
    };

   public void setMuscleText(String t) {
      musclesList.setText(t);
   }

   public MachineDetailsFragment getThis() {
      return this;
   }

   public TextWatcher watcher = new TextWatcher() {
         @Override
         public void onTextChanged(CharSequence s, int start,
           int before, int count) {
            requestForSave();
         }

         @Override
         public void beforeTextChanged(CharSequence s, int
start, int count,
               int after) {
         }

         @Override
         public void afterTextChanged(Editable s) {
         }
   };

   private void requestForSave() {
      toBeSaved = true;
      if (pager!=null) pager.requestForSave();
   }
```

```
   public void buildMusclesTable()    {
      _muscles[0]=
getActivity().getResources().getString(R.string.biceps);
      _muscles[1]=
getActivity().getResources().getString(R.string.triceps);
      _muscles[2]=
getActivity().getResources().getString(R.string.pectoraux);
      _muscles[3]=
getActivity().getResources().getString(R.string.dorseaux);
      _muscles[4]=
getActivity().getResources().getString(R.string.abdominaux);
      _muscles[5]=
getActivity().getResources().getString(R.string.quadriceps);
      _muscles[6]=
getActivity().getResources().getString(R.string.ischio_jambiers)
;
      _muscles[7]=
getActivity().getResources().getString(R.string.adducteurs);
      _muscles[8]=
getActivity().getResources().getString(R.string.mollets);
      _muscles[9]=
getActivity().getResources().getString(R.string.deltoids);
   }


   public String getMuscleNameFromId(int id){
      String ret = "";
      try {
         ret = _muscles[id].toString();
      } catch (Exception e) {
         e.printStackTrace();
      }
      return ret;
   }


   public int getMuscleIdFromName(String pName){
      for (int i=0; i<_muscles.length; i++) {
         if (_muscles[i].toString().equals(pName)) return i;
      }
      return -1;
   }


   public String getDBStringFromInput(String pInput){
      String[] data = pInput.split(";");
      String output = "";

      if (pInput.isEmpty()) return "";

      int i=0;
      if (data.length > 0 ) {
         output = String.valueOf(getMuscleIdFromName(data[i]));
```

```
         for (i=1; i < data.length; i++) {
            output = output  + ";"+
getMuscleIdFromName(data[i]);
         }
      }

      return output;
   }


      public String getInputFromDBString(String pDBString){
      String[] data = pDBString.split(";");
      String output = "";

      int i=0;

      try {
         if (data.length > 0 ) {
            if (data[0].isEmpty()) return "";

            if ( ! data[i].equals("-1") ) {
               output =
getMuscleNameFromId(Integer.valueOf(data[i]));
               _selections[Integer.valueOf(data[i])]=true;
               for (i=1; i < data.length; i++) {
                  if ( ! data[i].equals("-1") ) {
                     output = output + ";" +
getMuscleNameFromId(Integer.valueOf(data[i]));
                     _selections[Integer.valueOf(data[i])]=true;
                  }
               }
            }
         }
      } catch (NumberFormatException e) {
         output="";
         e.printStackTrace();
      }

      return output;
   }

   public int getCameraPhotoOrientation(Context context, Uri
imageUri, String imagePath){
       int rotate = 0;
       try {
           context.getContentResolver().notifyChange(imageUri,
null);
           File imageFile = new File(imagePath);

           ExifInterface exif = new
ExifInterface(imageFile.getAbsolutePath());
           int orientation =
exif.getAttributeInt(ExifInterface.TAG_ORIENTATION,
```

```
ExifInterface.ORIENTATION_NORMAL);

            switch (orientation) {
            case ExifInterface.ORIENTATION_ROTATE_270:
                rotate = 270;
                break;
            case ExifInterface.ORIENTATION_ROTATE_180:
                rotate = 180;
                break;
            case ExifInterface.ORIENTATION_ROTATE_90:
                rotate = 90;
                break;
            }

            Log.i("RotateImage", "Exif orientation: " +
orientation);
            Log.i("RotateImage", "Rotate value: " + rotate);
        } catch (Exception e) {
            e.printStackTrace();
        }
        return rotate;
    }

    public boolean toBeSaved() {
        return toBeSaved;
    }

    public void machineSaved() {
        toBeSaved = false;
    }

    public Machine getMachine() {
        Machine m=mMachine;
        m.setName(machineName.getText().toString());
        m.setDescription(machineDescription.getText().toString());

m.setBodyParts(getDBStringFromInput(this.musclesList.getText().t
oString()));
        m.setPicture(mCurrentPhotoPath);
        m.setFavorite(false);
        m.setType(selectedType);
        return m;
    }
}
```

Додаток Г

Файл tab_weight.xml

```
<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">

    <LinearLayout
        android:id="@+id/tab_weight"
        android:name="tab_weight"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="#EEEEEE"
        android:gravity="center"
        android:orientation="vertical"
        android:paddingBottom="0dp"
        android:paddingLeft="0dp"
        android:paddingRight="0dp"
        android:paddingTop="0dp">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="start"
            android:gravity="center"
            android:orientation="horizontal">

            <android.support.v7.widget.CardView
                android:layout_width="0dp"
                android:layout_height="wrap_content"

android:layout_gravity="center_vertical|center_horizontal"
                android:layout_margin="5dp"
                android:layout_weight="1"
                android:minHeight="70dp"
                android:minWidth="70dp"
                app:cardCornerRadius="5dp">

                <LinearLayout
                    android:layout_width="match_parent"
                    android:layout_height="match_parent"

android:layout_gravity="center_vertical|center_horizontal"
                    android:gravity="center_horizontal"
                    android:orientation="vertical">
```

```
                    <TextView
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"

android:layout_gravity="top|center_horizontal"
                        android:layout_marginTop="10dp"
                        android:autoSizeMaxTextSize="18sp"
                        android:autoSizeMinTextSize="14sp"
                        android:autoSizeStepGranularity="1dp"
                        android:gravity="center_horizontal"
                        android:lines="1"
                        android:text="@string/weightLabel"
                        android:textSize="18sp" />
                </LinearLayout>
            </android.support.v7.widget.CardView>

            <android.support.v7.widget.CardView
                android:layout_width="0dp"
                android:layout_height="wrap_content"

android:layout_gravity="center_vertical|center_horizontal"
                android:layout_margin="5dp"
                android:layout_weight="1"
                android:minHeight="70dp"
                android:minWidth="70dp"
                app:cardCornerRadius="5dp">

                <LinearLayout
                    android:layout_width="match_parent"
                    android:layout_height="match_parent"

android:layout_gravity="center_vertical|center_horizontal"
                    android:gravity="center_horizontal"
                    android:orientation="vertical">

                    <TextView
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"


<com.easyfitness.utils.EditableInputView.EditableInputViewWithDa
te
                        android:id="@+id/fatInput"
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"

android:layout_gravity="center_horizontal|center|clip_vertical"
                        android:layout_marginTop="5dp"

android:gravity="center_vertical|center_horizontal"

android:inputType="number|numberSigned|numberDecimal"
                        android:lines="1"
```

```
                        android:maxLines="1"
                        android:maxWidth="100dp"
                        android:textColor="@android:color/black"
                        android:textSize="30sp"
                        tools:text="50.9" />

                    <ImageButton
                        android:id="@+id/fatDetailsButton"
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"
                        android:layout_marginLeft="5dp"
                        android:layout_marginRight="5dp"
                        android:adjustViewBounds="false"
                        android:background="@color/background"
                        android:paddingBottom="5dp"
                        android:paddingTop="5dp"

app:srcCompat="@drawable/ic_baseline_list_alt_24px" />
                </LinearLayout>


            </android.support.v7.widget.CardView>

        </LinearLayout>

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="start"
            android:gravity="center"
            android:orientation="horizontal">

            <android.support.v7.widget.CardView
                android:layout_width="0dp"
                android:layout_height="wrap_content"

android:layout_gravity="center_vertical|center_horizontal"
                android:layout_margin="5dp"
                android:layout_weight="1"
                android:minHeight="70dp"
                android:minWidth="70dp"

<com.easyfitness.utils.EditableInputView.EditableInputViewWithDa
te
                        android:id="@+id/musclesInput"
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"

android:layout_gravity="center_horizontal|center|clip_vertical"
                        android:layout_marginTop="5dp"

android:gravity="center_vertical|center_horizontal"
```

```
android:inputType="numberSigned|numberDecimal"
                        android:lines="1"
                        android:maxLines="1"
                        android:maxWidth="100dp"
                        android:textColor="@android:color/black"
                        android:textSize="30sp"
                        tools:text="20.2" />

                    <ImageButton
                        android:id="@+id/musclesDetailsButton"
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"
                        android:layout_marginLeft="5dp"
                        android:layout_marginRight="5dp"
                        android:adjustViewBounds="false"
                        android:background="@color/background"
                        android:paddingBottom="5dp"
                        android:paddingTop="5dp"

app:srcCompat="@drawable/ic_baseline_list_alt_24px" />
                </LinearLayout>


            </android.support.v7.widget.CardView>

            <android.support.v7.widget.CardView
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_gravity="center"
                android:layout_margin="5dp"
                android:layout_weight="1"
                android:minHeight="70dp"
                android:minWidth="70dp"
                app:cardCornerRadius="5dp">


                <LinearLayout
                    android:layout_width="match_parent"
                    android:layout_height="match_parent"

android:layout_gravity="center_vertical|center_horizontal"
                    android:gravity="center_horizontal"
                    android:orientation="vertical">

                    <TextView
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"

android:layout_gravity="top|center_horizontal"
                        android:layout_marginTop="10dp"
                        android:autoSizeMaxTextSize="18sp"
                        android:autoSizeMinTextSize="14sp"
                        android:autoSizeStepGranularity="1dp"
```

```
                        android:gravity="center_horizontal"
                        android:lines="1"
                        android:text="@string/waterLabel"
                        android:textSize="18sp" />


<com.easyfitness.utils.EditableInputView.EditableInputViewWithDa
te
                        android:id="@+id/waterInput"
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"

android:layout_gravity="center_horizontal|center"
                        android:layout_marginTop="5dp"

android:gravity="center_vertical|center_horizontal"

android:inputType="numberSigned|numberDecimal"
                        android:lines="1"
                        android:maxLines="1"
                        android:maxWidth="100dp"
                        android:textColor="@android:color/black"
                        android:textSize="30sp"
                        tools:text="20.5" />



            </android.support.v7.widget.CardView>

        </LinearLayout>

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="start"
            android:gravity="center"
            android:orientation="horizontal">

            <android.support.v7.widget.CardView
                android:layout_width="0dp"
                android:layout_height="wrap_content"

android:layout_gravity="center_vertical|center_horizontal"
                android:layout_margin="5dp"
                android:layout_weight="50"
                android:minHeight="70dp"
                android:minWidth="70dp"
                app:cardCornerRadius="5dp">


                <LinearLayout

                    android:layout_width="match_parent"
```

```
                    android:layout_height="match_parent"

android:layout_gravity="center_vertical|center_horizontal"
                    android:gravity="center_horizontal"
                    android:orientation="vertical">

                    <RelativeLayout
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"

android:layout_gravity="center_vertical|center_horizontal"
                        android:layout_marginTop="10dp"
                        android:gravity="center_horizontal">


                        <ImageButton
                            android:id="@+id/imcHelp"
                            android:layout_width="25sp"
                            android:layout_height="25sp"

android:layout_alignParentBottom="false"

android:layout_alignParentEnd="false"

android:layout_alignParentRight="true"
                            android:layout_centerInParent="true"
                            android:layout_centerVertical="true"
                            android:adjustViewBounds="false"

android:background="@color/background"
                            android:cropToPadding="false"

app:srcCompat="@drawable/ic_baseline_help_outline_24px" />

                    </RelativeLayout>

                    <TextView
                        android:id="@+id/imcValue"
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"

android:layout_gravity="center_vertical|center_horizontal|center
"
                        android:layout_marginBottom="5dp"
                        android:layout_marginTop="5dp"

android:gravity="center_vertical|center_horizontal"
                        android:maxLines="1"
                        android:maxWidth="100dp"
                        android:textColor="@android:color/black"
                        android:textSize="30sp"
                        tools:text="20" />
```

```
                    <TextView
                        android:id="@+id/imcViewText"
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"
                        android:layout_marginBottom="10dp"
                        android:gravity="center"
                        android:textSize="14sp"
                        android:textStyle="italic"
                        tools:text="normal" />
                </LinearLayout>

                <LinearLayout

                    android:layout_width="match_parent"
                    android:layout_height="match_parent"

android:layout_gravity="center_vertical|center_horizontal"
                    android:gravity="center_horizontal"
                    android:orientation="vertical">

                    <RelativeLayout
                        android:id="@+id/rfmTitle"
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"

android:layout_gravity="center_vertical|center_horizontal"
                        android:layout_marginTop="10dp"
                        android:gravity="center_horizontal">

                        <ImageButton
                            android:id="@+id/rfmHelp"
                            android:layout_width="25sp"
                            android:layout_height="25sp"

android:layout_alignParentBottom="false"

android:layout_alignParentEnd="false"

android:layout_alignParentRight="true"
                            android:layout_centerInParent="true"
                            android:layout_centerVertical="true"
                            android:adjustViewBounds="false"

android:background="@color/background"
                            android:cropToPadding="false"

app:srcCompat="@drawable/ic_baseline_help_outline_24px" />

                    </RelativeLayout>

                    <TextView
                        android:id="@+id/rfmValue"
                        android:layout_width="match_parent"
```

```
                        android:layout_height="wrap_content"

android:layout_gravity="center_vertical|center_horizontal|center
"
                        android:layout_marginBottom="5dp"
                        android:layout_marginTop="5dp"

android:gravity="center_vertical|center_horizontal"
                        android:maxLines="1"
                        android:maxWidth="100dp"
                        android:textColor="@android:color/black"
                        android:textSize="30sp"
                        tools:text="20" />

                    <TextView
                        android:id="@+id/rfmViewText"
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"
                        android:layout_marginBottom="10dp"
                        android:gravity="center"
                        android:textSize="14sp"
                        android:textStyle="italic"
                        tools:text="normal" />


                </LinearLayout>


            </android.support.v7.widget.CardView>

        </LinearLayout>

    </LinearLayout>
</ScrollView>
```

Файл WeightFragment.java

```
import android.app.Activity;
import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.support.v4.app.FragmentTransaction;
import android.view.Gravity;
import android.view.LayoutInflater;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.ImageButton;
import android.widget.Spinner;
import android.widget.TextView;
```

```
import com.easyfitness.DAO.DAOProfil;
import com.easyfitness.DAO.DAOWeight;
import com.easyfitness.DAO.Profile;
import com.easyfitness.DAO.bodymeasures.BodyMeasure;
import com.easyfitness.DAO.bodymeasures.BodyPart;
import com.easyfitness.DAO.bodymeasures.DAOBodyMeasure;
import com.easyfitness.bodymeasures.BodyPartDetailsFragment;
import
com.easyfitness.utils.EditableInputView.EditableInputView;
import
com.easyfitness.utils.EditableInputView.EditableInputViewWithDat
e;
import com.onurkaganaldemir.ktoastlib.KToast;

import cn.pedant.SweetAlert.SweetAlertDialog;


public class WeightFragment extends Fragment {
    private EditableInputViewWithDate weightEdit = null;
    private EditableInputView fatEdit = null;
    private EditableInputView musclesEdit = null;
    private EditableInputView waterEdit = null;

    private ImageButton weightDetailsButton = null;
    private ImageButton fatDetailsButton = null;
    private ImageButton musclesDetailsButton = null;
    private ImageButton waterDetailsButton = null;

    private TextView imcText = null;
    private TextView imcRank = null;
    private TextView rfmText = null;
    private TextView rfmRank = null;
    private ImageButton rfmHelpButton = null;
    private ImageButton imcHelpButton = null;

    MainActivity mActivity = null;
    private DAOWeight mWeightDb = null;
    private DAOBodyMeasure mDbBodyMeasure = null;
    private DAOProfil mDb = null;

    public static WeightFragment newInstance(String name, int
id) {
        WeightFragment f = new WeightFragment();

        Bundle args = new Bundle();
        args.putString("name", name);
        args.putInt("id", id);
        f.setArguments(args);

        return f;
    }
```

```
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup
container,
                             Bundle savedInstanceState) {

        View view = inflater.inflate(R.layout.tab_weight,
container, false);

        weightEdit = view.findViewById(R.id.weightInput);
        fatEdit = view.findViewById(R.id.fatInput);
        musclesEdit = view.findViewById(R.id.musclesInput);
        waterEdit = view.findViewById(R.id.waterInput);
        weightDetailsButton =
view.findViewById(R.id.weightDetailsButton);
        fatDetailsButton =
view.findViewById(R.id.fatDetailsButton);
        musclesDetailsButton =
view.findViewById(R.id.musclesDetailsButton);
        waterDetailsButton =
view.findViewById(R.id.waterDetailsButton);
        imcText = view.findViewById(R.id.imcValue);
        imcRank = view.findViewById(R.id.imcViewText);
        rfmText = view.findViewById(R.id.rfmValue);
        rfmRank = view.findViewById(R.id.rfmViewText);

        rfmHelpButton = view.findViewById(R.id.rfmHelp);
        imcHelpButton = view.findViewById(R.id.imcHelp);


        weightEdit.setOnTextChangeListener(itemOnTextChange);
        fatEdit.setOnTextChangeListener(itemOnTextChange);
        musclesEdit.setOnTextChangeListener(itemOnTextChange);
        waterEdit.setOnTextChangeListener(itemOnTextChange);
        imcHelpButton.setOnClickListener(showHelp);
        rfmHelpButton.setOnClickListener(showHelp);

weightDetailsButton.setOnClickListener(showDetailsFragment);

fatDetailsButton.setOnClickListener(showDetailsFragment);

musclesDetailsButton.setOnClickListener(showDetailsFragment);

waterDetailsButton.setOnClickListener(showDetailsFragment);

        mWeightDb = new DAOWeight(view.getContext());
        mDbBodyMeasure = new DAOBodyMeasure(view.getContext());

        return view;
    }

    @Override
    public void onStart() {
        super.onStart();
```

```
        refreshData();
    }

    @Override
    public void onAttach(Activity activity) {
        super.onAttach(activity);
        this.mActivity = (MainActivity) activity;
    }

    private AdapterView.OnClickListener showDetailsFragment =
new AdapterView.OnClickListener() {
        @Override
        public void onClick(View v) {
            int bodyPartID = BodyPart.WEIGHT;
            switch (v.getId()) {
                case R.id.weightDetailsButton:
                    bodyPartID = BodyPart.WEIGHT;
                    break;
                case R.id.fatDetailsButton:
                    bodyPartID = BodyPart.FAT;
                    break;
                case R.id.musclesDetailsButton:
                    bodyPartID = BodyPart.MUSCLES;
                    break;
                case R.id.waterDetailsButton:
                    bodyPartID = BodyPart.WATER;
                    break;
            }

    public String getName() {
        return getArguments().getString("name");
    }

    private float calculateImc(float weight, int size) {
        float imc = 0;

        if (size==0) return 0;

        imc = (float)(weight / (size / 100.0 * size / 100.0));

        return imc;
    }

    private String getImcText(float imc) {
        if (imc<18.5) {
            return getString(R.string.underweight);
        } else if (imc < 25) {
            return getString(R.string.normal);
        } else if (imc < 30) {
            return getString(R.string.overweight);
        } else {
            return getString(R.string.obese);
```

```
        }
    }

    private float calculateRfm(float waistCirc, int sex, int
size) {
        float rfm = 0;

        if (waistCirc == 0) return 0;

        return 0;
    }


    private String getRfmText(float rfm) {
        if (rfm < 18.5) {
            return "underweight";
        } else if (rfm < 25) {
            return "normal";
        } else if (rfm < 30) {
            return "overweight";
        } else {
            return "obese";
        }
    }

    private void refreshData() {
        View fragmentView = getView();
        if (fragmentView != null) {
            if (getProfil() != null) {
                BodyMeasure lastWeightValue = null;
                BodyMeasure lastWaterValue = null;
                BodyMeasure lastFatValue = null;
                BodyMeasure lastMusclesValue = null;

                if (getProfil()!=null) {
                    lastWeightValue =
mDbBodyMeasure.getLastBodyMeasures(BodyPart.WEIGHT,
getProfil());
                    lastWaterValue =
mDbBodyMeasure.getLastBodyMeasures(BodyPart.WATER, getProfil());
                    lastFatValue =
mDbBodyMeasure.getLastBodyMeasures(BodyPart.FAT, getProfil());
                    lastMusclesValue =
mDbBodyMeasure.getLastBodyMeasures(BodyPart.MUSCLES,
getProfil());
                }

                if (lastWeightValue != null) {

weightEdit.setText(String.valueOf(lastWeightValue.getBodyMeasure
()));
                    int size = getProfil().getSize();
                    if (size == 0) {
```

```
                        imcText.setText("-");

imcRank.setText(R.string.no_size_available);
                    } else {
                        float imcValue =
calculateImc(lastWeightValue.getBodyMeasure(), size);
                        imcText.setText(String.format("%.1f",
imcValue));
                        imcRank.setText(getImcText(imcValue));
                    }
                } else {
                    weightEdit.setText("-");
                    imcText.setText("-");

imcRank.setText(R.string.no_weight_available);
                }

                if (lastWaterValue!=null)

waterEdit.setText(String.valueOf(lastWaterValue.getBodyMeasure()
));
                else
                    waterEdit.setText("-");

                if (lastFatValue!=null)

fatEdit.setText(String.valueOf(lastFatValue.getBodyMeasure()));
                else
                    fatEdit.setText("-");

                if (lastMusclesValue != null)

musclesEdit.setText(String.valueOf(lastMusclesValue.getBodyMeasu
re()));
                else
                    musclesEdit.setText("-");
            }
        }
    }

    private BtnClickListener itemClickDeleteRecord = new
BtnClickListener() {
        @Override
        public void onBtnClick(long id) {
            showDeleteDialog(id);
        }
    };

    private Spinner.OnItemSelectedListener
itemOnItemSelectedChange = new Spinner.OnItemSelectedListener()
{

        @Override
```

```
        public void onItemSelected(AdapterView<?> parent, View
view, int position, long id) {
            refreshData();
        }

        @Override
        public void onNothingSelected(AdapterView<?> parent) {

        }
    };


    private EditableInputView.OnTextChangedListener
itemOnTextChange = new EditableInputView.OnTextChangedListener()
{

        @Override
        public void onTextChanged(EditableInputView view) {
            EditableInputViewWithDate v =
(EditableInputViewWithDate) view;
            try {
                switch (view.getId()) {
                    case R.id.weightInput:
                       float weightValue =
Float.parseFloat(v.getText());

mDbBodyMeasure.addBodyMeasure(v.getDate(), BodyPart.WEIGHT,
weightValue, getProfil().getId());
                        break;
                    case R.id.fatInput:
                        float fatValue =
Float.parseFloat(v.getText());

mDbBodyMeasure.addBodyMeasure(v.getDate(), BodyPart.FAT,
fatValue, getProfil().getId());
                        break;
                    case R.id.musclesInput:
                        float musclesValue =
Float.parseFloat(v.getText());

mDbBodyMeasure.addBodyMeasure(v.getDate(), BodyPart.MUSCLES,
musclesValue, getProfil().getId());
                        break;
                    case R.id.waterInput:
                        float waterValue =
Float.parseFloat(v.getText());

mDbBodyMeasure.addBodyMeasure(v.getDate(), BodyPart.WATER,
waterValue, getProfil().getId());
                        break;
                }
            } catch (NumberFormatException e) {
```

```
            }

            refreshData();
        }
    };

    private OnClickListener showHelp = new OnClickListener() {

        @Override
        public void onClick(View v) {
            switch (v.getId()) {
                case R.id.imcHelp:
                    new SweetAlertDialog(getContext(),
SweetAlertDialog.NORMAL_TYPE)

.setTitleText(R.string.BMI_dialog_title)

.setContentText(getString(R.string.BMI_formula))

.setConfirmText(getResources().getText(R.string.global_ok).toStr
ing())
                            .showCancelButton(true)
                            .show();
                    break;
                case R.id.rfmHelp:
                    new SweetAlertDialog(getContext(),
SweetAlertDialog.NORMAL_TYPE)

.setTitleText(R.string.RFM_dialog_title)

.setContentText(getString(R.string.RFM_female_formula) +

getString(R.string.RFM_male_formula))

.setConfirmText(getResources().getText(R.string.global_ok).toStr
ing())
                            .showCancelButton(true)
                            .show();
                    break;
            }
        }
    };

    private void showDeleteDialog(final long idToDelete) {

        new SweetAlertDialog(getContext(),
SweetAlertDialog.WARNING_TYPE)

.setTitleText(getString(R.string.DeleteRecordDialog))

.setContentText(getResources().getText(R.string.areyousure).toSt
ring())
```

```
.setCancelText(getResources().getText(R.string.global_no).toStri
ng())

.setConfirmText(getResources().getText(R.string.global_yes).toSt
ring())
                .showCancelButton(true)
                .setConfirmClickListener(new
SweetAlertDialog.OnSweetClickListener() {
                    @Override
                    public void onClick(SweetAlertDialog
sDialog) {

mDbBodyMeasure.deleteMeasure(idToDelete);
                        refreshData();
                        KToast.infoToast(getActivity(),
getResources().getText(R.string.removedid).toString(),
Gravity.BOTTOM, KToast.LENGTH_LONG);
                        sDialog.dismissWithAnimation();
                    }
                })
                .show();
    }

    private Profile getProfil() {
        return ((MainActivity)
getActivity()).getCurrentProfil();
    }

    public Fragment getFragment() {
        return this;
    }

    @Override
    public void onHiddenChanged(boolean hidden) {
        if (!hidden) refreshData();
    }
}
```

БІБЛІОГРАФІЧНА ДОВІДКА

Тема дипломної роботи: Розробка програми мобільного застосунку з надання фітнес послуг засобами Android Java.

Обсяг пояснювальної записки:. ____61___аркуші

Перелік графічних матеріалів:

- таблиць__2_____
- рисунків_31_____
- додатків__4 на 46_____аркушах

Дата закінчення дипломного проекту: “10” червня 2019 р.

Студент – дипломник _____
(підпис) (розшифровка підпису)