**ТЕХНОЛОГИЧНО УЧИЛИЩЕ “ЕЛЕКТРОННИ СИСТЕМИ”**

**към ТЕХНИЧЕСКИ УНИВЕРСИТЕТ - СОФИЯ**

**ДИПЛОМНА РАБОТА**

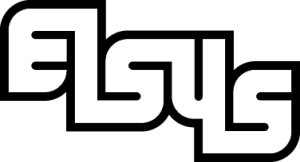
Тема: Android фитнес приложение

Дипломант: Научен ръководител:

*Ивайло Георгиев Иван Игнатов*

СОФИЯ

2020

**TЕХНОЛОГИЧНО УЧИЛИЩЕ “ЕЛЕКТРОННИ СИСТЕМИ”**

**към ТЕХНИЧЕСКИ УНИВЕРСИТЕТ - СОФИЯ**

Дата на заданието: 15.11.2019 г. Утвърждавам:..............................

Дата на предаване: 15.02.2020 г. /проф. д-р инж. Т. Василева/

**ЗАДАНИЕ**

**за дипломна работа**

на ученика Ивайло Николаев Георгиев 12 А клас

1.Тема: Мобилно фитнес приложение

2.Изисквания:

2.1 Подръжка на различни акаунти

2.2 Упражнения с характеристики и история със серии и повторения

2.3 Крачкометър

2.4 База данни с храни

2.5 Задаване на желани резултати и изчисляване на нужните за постигането им калории и макроси, както и предложения за тренировки

2.6 Списък с приятели и feed от техните дейности

2.7 Записване на тегло, упражнения и храни

2.8 Сканиране на баркод на стока

3.Съдържание 3.1 Обзор

3.2 Същинска част

3.3 Приложение

Дипломант :...........................................

Ръководител:..........................................

/ Иван Игнатов/

Директор:................................................

/ доц. д-р инж. Ст. Стефанова /

# Увод

Съвременното ни ежедневие носи със себе си забързан начин на живот, в който се налага да лавираме между семейство, работа, задължения, както и да усвояваме нови умения. Макар част от този проблем, технологичния прогрес ни предоставя база, с която да изградим решения за него. Тук ще засегна създаването на фитнес приложение, като необходим помощник в спортните начинания, а от там и в управлението на времето и усилията ни. Развивайки дипломната си работа се стремя не само към да улесняване на всекидневието (включително това на спортистите), но и мотивирам онези, които желаят промяна във начин си на живот, в здравословна

Хората, често правят първите си стъпки с голяма доза несигурност, навлизайки в сферата на спортната дейност. Веднъж, превърнали спорта в неотменна част от ежедневието си, той набира инерция, заемайки все повече от времето и енергията ни.

Така се зароди идеята ми да създам мобилно – фитнес приложение, което да се превърне в незаменим асистент в следенето на десетките показатели, които всеки спортист, бил той аматьор или професионално занимаващ се – следи. Мобилното ми приложение ще включва в себе си функции, като:

* Крачкомер;
* Фитнес упражнения – с всички техни движения и параметри;
* Изчисляване на прием на калории в зависимост от зададена от потребителя цел.

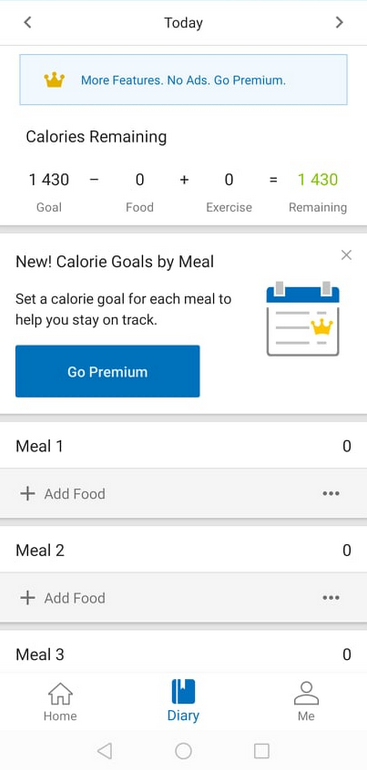
# ПЪРВА ГЛАВА

# Обзор на съществуващите решения. Преглед на технологии подходящи за изпълнение на дипломната работа.

## 1.1. Съществуващи приложения

### 1.1.1. MyFitnessPal

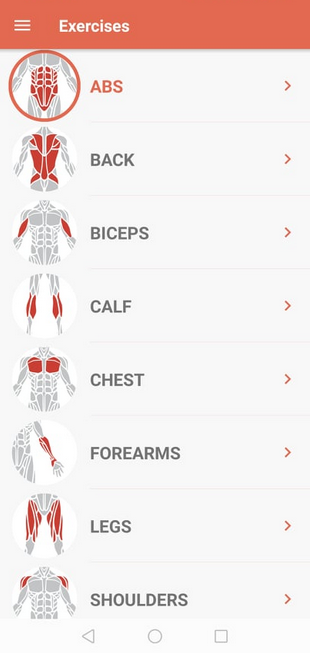
MyFitnessPal е фитнес приложение, с което потребителите да проследяват приема на дневното количество калории (Фигура 1.1). Разработена е както декстоп версия за Windows, така и мобилни версия за Android и IOS. Приложението предоставя интуитивен потребителски интерфейс (UI), позволяващ бързото му усвояване от разнородни възрастови групи - млади и възрастни. Приложението е подходящо и за тези, които се интересуват от спазването на хранителен режим. Конкурентното предимство на MyFitnessPal са функциите му за изчисляване и следене на прием и разход на калории, но не се изчерпват до там, той включва и крачкометър, рецепти за диетични хранения (които разработчиците, периодично допълват, редактират или премахват), списък с приятели – заемка от социалните мрежи, както и набор от статии посветени на здравословния начин на живот. Като сериозен пропуск в приложението намирам липсата на опция за създаване на личен режим/програма за тренировки, както и информационно табло(dashboard), което да визуализира прогреса, както и допълнителна информация по привлекателен за потребителя начин (чрез различни динамични фигури и скали).



*Фигура 1.1*

### 1.1.2. Fitness & Bodybuilding

Fitness & Bodybuilding е приложение от групата на MyFitnessPal, но насочено към техниката на упражненията и създадено, по-скоро за бодибилдинг средите. (Фигура 1.2). Ето защо приложението е разработено за Аndroid и IOS, и липса десктоп версия – тъй като се очаква, че потребителя ще го използва в самата зала. По същата причина не са развити функции, като калориен калкулатори или съвети за хранителен режим. Особено преимущество на работата с това приложение е улеснения интерфейс, подходящ и за потребители без сериозен опит във фитнес среда. Приложението предлага и набор от препоръчителни тренировки според нивото на ползващия го, като в същото време е напълно безплатна (Premium версия не съществува). Освен изчисленията на калории, очевидна е и липсата на крачкометър, и възможността за създаване на индивидуализирани упражнения.



*Фигура 1.2*

## 1.2. Технологии и средства за развой

### 1.2.1. Android Studio

Android Studio(Фигура 1.3) е официалнатa среда за разработка (IDE - Integrated Development Enviroment) за операционната система на Google – Android, изградена на основата на JetBrains' IntelliJ IDEA софтуера, проектирано специално за разработка на Android приложения. Android Studio включва както основните инструменти за работа под IntelliJ, така и строго специфични елементи и функции, необходими в развоя на Android приложения, сред тях са:

* Gradle базирана изграждаща система;
* Емулатор включващ в себе си множество инструменти и функционции, работещ изключително бързо.

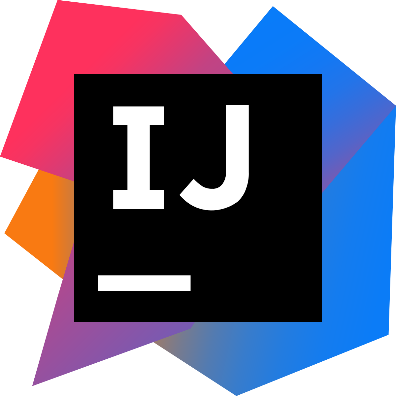
Android Studio разполага с версии за множество Операционни системи версиите на Windows след в. 7, macOS, както и Linux дистрибуции (базирани основно на Debian и Ubuntu). Успешно замества други среди за разработка, като например [Eclipse Android Development Tools](https://en.wikipedia.org/wiki/Eclipse_(software)#_blank) (ADТ), тъй като предлага богат набор от инструменти, засягащи фронт енд и бек енд. Android Studio, както подсказва името му е официалната среда за разработка на Android приложения.



*Фигура 1.3*

### 1.2.2. IntelliJ IDEA

IntelliJ IDEA(Фигура 1.4) е интегрирана среда за разработка (IDE) за компютърен софтуер създадена от JetBrains. Достъпна както под [Apache 2 Licensed](https://bg.wikipedia.org/w/index.php?title=Apache_2_Licensed&action=edit&redlink=1) community edition, такаи под платен лиценз - commercial edition. И двете версии на средата позволяват разработка на приложения, разбира се платената версия е насочена към професионалните среди и предоставя значително повече инструменти и гъвкавост. Възходът на IntelliJ започва през 2010 година, когато получава най-висок резултат от четирите най-използвани инструменти за програмиране в Java, а именно - [Eclipce](https://bg.wikipedia.org/w/index.php?title=Eclipce&action=edit&redlink=1), IntelliJ IDEA, [NetBeans](https://bg.wikipedia.org/wiki/NetBeans) and [JDeveloper](https://bg.wikipedia.org/w/index.php?title=JDeveloper&action=edit&redlink=1). [Интегрираната среда за разработка](https://bg.wikipedia.org/wiki/Интегрирана_среда_за_разработка) (IDE) подпомага разработчиците, чрез множество инструменти, подсигуряващи завършеността и кохерентността на програмния код, чрез анализ на контекста на програмния код, улеснена навигация в кода, избистрена структура – позволяваща създаване на разнородни структури, като класове, декларации, директно в програмния код, преработка му, както и система предлагаща поправки в програмния код при синтактични грешки или други несъответствия.



*Фигура 1.4*

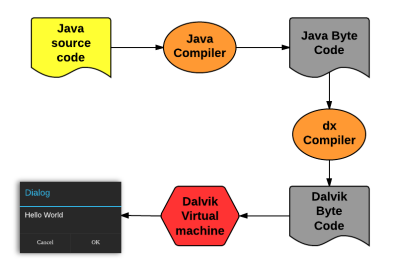
### 1.2.3. Basic4Android

Basic4Android(Фигура 1.5) е RAD(Rapid Application Development) инструмент, използван далеч по-рядко, тъй като е разработен на основата на Visual Basic – от където и произлиза името на тази среда – Basic For Android.



*Фигура 1.5*

Средата позволява ненадмината бързина в създаването на програмен код, както и в оформянето на потребителския интерфейс, но бидейки базиран на Visual Basic, не е популярен сред разработчиците от java средите, заради синтактичните разлики между двата езика. Сериозен пропуск в тази среда е липсата на онаследяване на класове, както и рудиментарната форма на класове, липса, която може да бъде заобиколена, но с цената на сериозни усилия и знания от страна на програмиращия. В заключение, тази среда е подходяща за бързо създаване на Android приложения, без да ограничава потребителя от необходимите му функции, но изисква по-дълбоки общи познания от използващите го. Средата разполага и с визуален дизайнер, за създаване на потребителския интерфейс (UI), който обаче има своите ограничения, поради, което в професионалните среди за дизайна се използва изцяло текстовата среда на B4A. Любопитен е процеса, в който средата създава кода - езикът, на който се програмира, е подобен на Visual Basic и Visual Basic.Net, но програмата го „превежда“ на java, след което тя на свой ред бива преработена в машинен код за Dalvik Virtual Machine(Фигура 1.6).



*Фигура 1.6 илюстрира комплексния цикъл през , който преминава машинния код, за да бъде нобработен от микропроцесора на Android устройството.*

### 1.2.4. Java

Java(Фигура 1.7) е [обектно-ориентиран](https://bg.wikipedia.org/wiki/Обектно-ориентирано_програмиране) [език за програмиране](https://bg.wikipedia.org/wiki/Език_за_програмиране). Изходният код, създаден в Java, не се компилира до машинен код за определен , [микропроцесор](https://bg.wikipedia.org/wiki/Микропроцесор)а до междинен език - така нареченият [байткод](https://bg.wikipedia.org/w/index.php?title=Байткод&action=edit&redlink=1). Байткодът не се изпълнява директно от микропроцесора, а от негов аналог – [виртуален процесор](https://bg.wikipedia.org/wiki/Виртуална_машина), наречен [Java Virtual Machine](https://bg.wikipedia.org/wiki/Java_Virtual_Machine) (JVM). Коренът на тази така усложнена система е архитектурата на мобилните устройства използващи Android ОС – те използват ARM микропроцесори (Advanced RISC Machine), които използват различен тип инструкции от микропроцесорите, разпространени сред хардуера за настолни и мобилни компютри. Употребата на ARM процесори за мобилните устройства е продиктувана основно от по-ниската им енергийна консумация, докато същевременно производителността им е достатъчно голяма, параметри с особен приоритет за мобилния хардуер. Друг позитив в използването на ARM орицесорите е динамичната стойност на работните температури – резкия спад в производителността на микйропроцесора, води до рязък спад в температурата на микропроцесора. Ето защо, наред с изящната система за енергиен мейнджмънт, мобилните устройства минимилизират натоварванията върху процесора, чрез различен тип цикъл в живота на програмите, които изпълняват(приложенията). Цикъла на живот на мобилно приложение е Създаване на програмния код (Create Event), последвано от – Пауза (Pause), а именно всяко събитие, което скрива приложението или по някаква причина временно не го използва и/или завръщане към приложението (Resume Event) , ако не се осъществи завръщане към приложението във времевата рамка на Андроид ОС, последната прекратява жизнения цикъл на приложението, освобождавайки ценни ресурси и спестявайки енергия. Тази структиура е изключително важна в контекста на Java програмния език, тъй като последния се използва в JVM, която е известна със затрудненията си да изчиства рам паметта си. Което далеч не превръща java в неподходящ за разработка на мобилни приложения език - напротив , сред основните му предимства са:

* теоритично, лесната преносимост между разнородни платформи (софтуер или хардуер);
* освобождаване на паметта от обекти, които не се използват (Garbage collector), което има коствен ефект и върху енергийната консумация;
* първоначално заделяне на граница за консумация на оперативната памет (memory heap);
* Относителна степен на сигурност поради факта, че програмистите не работят директно с паметта (а когато разглеждаме java в контекста на Android средата дори по-висока, тъй, като всяко приложение съществува и работи само и единствено във свой изолиран, виртуален контейнер;



*Фигура 1.7*

### 1.2.5. Kotlin

Kotlin(Фигура 1.8) е крос-платформен програмен език за общо предназначение, който е използван основно в Android Studio среда. Още със създаването си е структуриран, предвиждайки тясно взаимодействие с Java и JVM, като в отделни ситуации може да работи в нееднородна среда с java. Kotlin основно работи с JVM, но не се ограничава до нея и може да се компилира до JavaScript или native код. Най-голяма популярност има в изработването на Android приложения, като към момента е основния език за разработка под Android Studio, измествайки от тази позиция java, който макар и поддържан, вече е второстепенен език. Според създателите на езика – стремежът им е бил насочен към улесняване на работния процес, без това да е за сметка на функционалността на езика, по този начин подобрявайки и ускорявайки процеса на разработка на мобилни приложения.

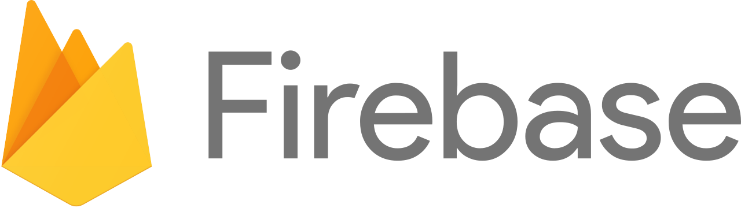


*Фигура 1.8*

### 1.2.6. Firebase

Firebase(Фигура 1.9) е платформа, закупена от Google и представлява инфраструктура за мобилни и web приложения. Инструментите, които предоставя са:

* Инструменти за анализи (по потребители, възрастови групи, културни групи, потребление във времеви отрязъци, на практика основните статистики, които google прилагат върху потребителите си;
* Бази от данни, достъпни на сървъри на google;
* Cloud сървъри за съхранение на данни;
* Hosting услуги.



*Фигура 1.9*

# Втора Глава

# Функционални изисквания към приложението. Проектиране на фитнес приложение

## 2.1. Функционални изисквания към фитнес приложението

* Поддръжка на потребителски регистрации (акаунти) и индивидуализиран профил за всеки потребител, включващ Име, пол, физически характеристики на потребителите, като тегло, възраст и височина;
* Упражнения с детаилна информация за тях, хронология на практикуваните упражнения, включваща сериите и повторенията, направени от потребителя;
* Крачкометър – използващ вградените в мобилния хардуер сензори и технологии в Android ОС за правилно и прецизно отчитане на инфромацията от гореспоменатите сензори;
* База от данни съдържаща в себе си често употребявани храни и напитки, техни характеристики, като калоричност, грамаж на разфасовки им и съставки им. Базата от данни има двуяка роля, тъй, като ще спомогне за бързото намиране на конкретна цена, след сканиране на баркод на продукта;
* Задаване на потребителски цели - желан резултати и изчисляване на нужната загуба или прием на калории за постигането им. Към същия инструментариум ще са включени макроси, както и предложения за тренировки (под формата на кратки статии / инструкции или снимки).
* Списък с приятели – технически други потребители на приложението и създаване на своеобразна социална мрежа около него. Ще бъде включена и функция с потребителски дневник (feed/log) – съставен от списък с извършените от други потребители дейности, записвани в база от данни.
* Проследяване на потребителското тегло, съхраняване на данни за упражненията и храните, които в последвствие ще намерят директно приложение в изчисленията. Тази информация би позволила разширяване набora от изчисления в последващи версии на приложението – бърз и лесен начин за добавяне на нови функции за потребителите.
* Сканиране на баркод на стока - което ще улесни проверка за калорийната стойност на приеманите храни, доколкото храните са записани в базата данни.

## 2.2. Съображения за избор на програмни средства и развойната среда

### 2.2.1. IDE – Android Studio

Сред най-големите позитиви на Android Studio са неговата директна цена – то е безплатно, а богатата документация и широката база от потребители го правят идеално за целите на тази дипломна работа. Средата се комбинира успешно с друг продукт на Google, а именно с Firebase и разполага с богат набор от собствени инструменти, като графичен редактор на xml файлове, система за създване на .apk инсталационни файлове и други.

### 2.2.2. Firebase

Firebase е безплатна и лесна за употреба платформа, която предоставя нужните за дипломната работа функции. Разполага с база данни в реално време, инструменти за анализ на потребителите(analytics) и завършена система за вход и изход (authentication). База данни, която предлага, пази данните синхронно между потребителските приложения, използвайки обекти, които засичат събития в реално време (event listener).

### 2.2.3. Език за програмиране – Java

Основните езици използвани в Android studio са два - Java и Kotlin. Избрах Java, заради предишния си опит и по-доброто познаване на правилата му. Към днешния момент, той все още е по-добре познат на общността около програмирането за мобилни приложения, което обуславя по-улеснения процес по намиране на решения на проблеми в сайтове като github и stackoverflow. От друга страна Kotlin е разработен, като крос език, което означава, че в бъдеще бих могъл да комбинирам новия официален език за Android studio с java, ако това се наложи.

## 2.3. Принцип на софтуерния модел

### 2.3.1. Структура на приложението

Основната структурна единица на Android приложенията е т.нар. Activity. Най-просто казано то е един екран от приложението, с който потребителят може да си взаимодейства.

Приложението съдържа много различни екрани, които ще са свързани помежду си. Ще е свързано с базата данни на Firebase, която структура ще се концентрира около потребителя и неговите дейности в приложението.

# ТРЕТА ГЛАВА

# Програмна реализация на мобилно фитнес приложение

## 3.1. Създаване на първоначалния екран

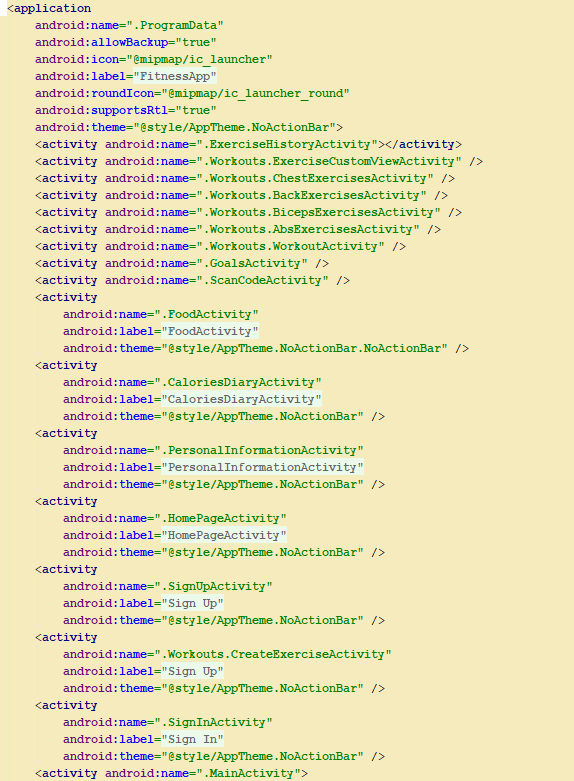
При стартиране на приложението Android Os създава, първоначално своя основен процес – Main Activity, което в интерфейсен смисъл ще представлява и основния екран на приложението. В него са позиционирани единствено два бутона – бутон за регисттрация и за вход – основни стъпки, с които подсигуряваме на приложението потребителски данни, без които иначе приложението не може да продължи зареждането си. Бутоните са свързани ллогически с програмния код, не чрез традиционния ClickListener, който работи, изпълнявайки се след засечено натискане (touch down), а при отпускане на бутона (Touch up). В методите се създава променлива от тип Intent, която е абстрактно описание на действие за преминаване от един процес (activity) в друг. Като тук трябва да направим уточнение за термина процес в контекста на Анdroid ОС и жизнения цикъл на приложенията - процесът в смисъла на activity не е задължително да бъде един единствен за приложението, какъвто е класическия пример с десктопните програми. Напроти, Android приложенията, често използват няколко процеса, а употребата на повече от един е промотирана от създателите на Android платформата. До известна степен, използването на различни процеси само за едно приложение, допринася за по-лесния контрол на енергийна консумация, комбиниране на процеси между приложения (в случай, когато разработика го е предвидил), но и инструментаиум за изолиране на процес от други процеси (били те дори в рамките ба едно и също приложение), с цел по-висока сигурност. В последствие се инициализира променливата mauth, която е от тип FirebaseAuthм последния ще използваме в последствие, като основа за вход и изход на потребителя конектор към база данни и база данни. Както е видно от приложения по-долу код, FireBase модула влиза в действие още от самото начало при вход или ако потребителя използва приложението за пръв път – регистрация, с която потребителя да получи възможност впоследствие за да влезе. В контекста на горното обяснение засягащо процесите в Андроид и цикъла им на живот – видно е, че бутоните извикват отделни процеси - „SignUpActivity“, „SignInActivity“, т.е. още от самото начало, независимо от посоката, в която ще поеме потребителя (вход в приложението или регистрация на потребителски акаунт), приложението вече разполага с два процеса - „Main Activity“ и „SignUpActivity“ или „SignInActivity“ в зависимост от ситуацията.



*Фигура 3.1*

### 3.1.1 Струтуроопределящ елемент на приложението.

Програмен код, в т.нар. манифест(Фигура 3.2) на приложението. В манифеста се дефинират множество програмни характеристики, като името на приложението, схема на дизайна на приложението (app layout).



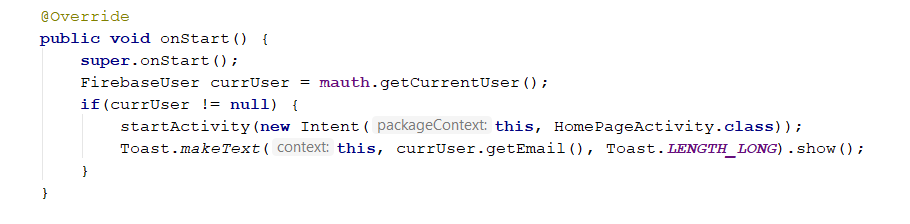
*Фигура 3.2*

На приложената снимка на програмния код, е видно, че екрана е разделен на две части – в лявата част е кода (във xml формат), а в ляво е визуализиран екрана на приложението (често използвана функция в Android Studio). Дизайна на приложението обикновено е кохерентен и универсален за всички екрани в приложението, но разработчика, не е ограничен технически, и при нужда би могъл да зададе различни дизайни за всеки един от екраните (activity).

Отрязък от програмен код от манифеста на приложението, в който се дефинира името на приложението, иконата му, схема на дизайна му, както и дефиниране на всички процеси в него (activity). Името на приложението може да бъде дефинирано и в ресурсите на приложението (а именно strings), където чрез id ключове, да се зададе име на съответния ключ. Подобна функционалност е особено полезна, за приложения, преведени на няколко езика. Приложения с повече от един дефиниран език, проверяват, избрания по подразбиране език от потребителя в Android ОС, и в случай, че разполагат с този език, превключват приложението на него. В тази част от програмния код се дефинират и стиловете на външен вид (theme) на всеки един процес.

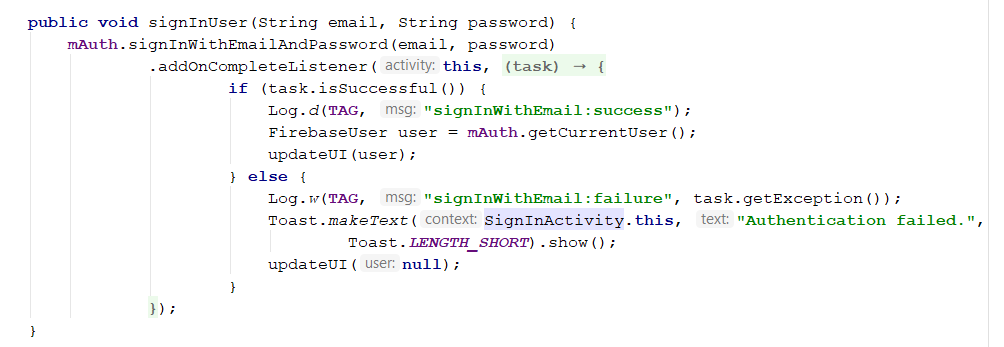
## 3.2. Поддръжка на различни акаунти и начин на регистриране.

В последствие се използва модула Firebase и в процеса на вход – чрез него и в неговата база от данни приложението търси потребителското id, чрез което в последствие ще изпраща заявка към базата данни, за да извлече нужните данни за калкулации. Самия Firebase е лесно достъпен, тъй като в Android Studio са интегрирани и предвидени връзките между средата за разработка и firebase модула, т.е. от разработчика не се изисква да намери начин да извиква тези функции – това вече е сторено за него(Фигура 3.3).

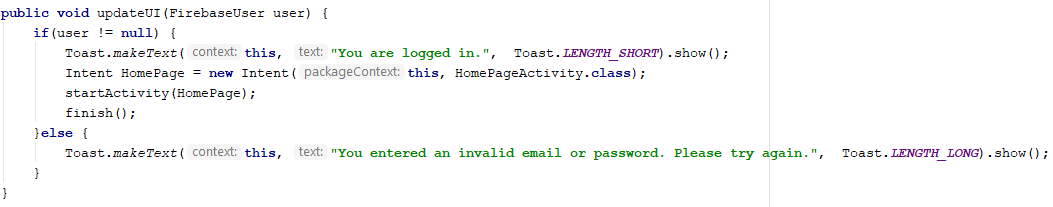


*Фигура 3.3...*

Видно от фигура 3.4 е лесното решение за вход в приложението – отново функционалността на модула FireBase е свързана с Android Studio, така, че на разработчика му остава, само да провери за съответствие между потребителското име попълнено в полето за вход и паролата към него в базата данни на Firebase. Сценарий за неуспешен вход (при липса на съвпадения в базата) е предвиден и целия процес процес на влизане протича в едно съвсем обикновенно условие „ако“ (if) . Макар, съществувщия инструментариум, обаче, като разработчици не ни е е отнета гъвкавостта и при нужда можем дакомбинираме или измислим вариации на тази логика. В случая, тя пасва идеално с целите на това приложение.

*Фигура 3.4.*

Самия Firebase модул е предвидил и справяне с потребителските сесии, т.е. той не просто превежда приложението през целия цикъл на вход и изход от приложението, но предвижда и конфликтни ситуации, например при опит за влизане в системата, когато потребителя вече е вътре. От фигура 3.5, е видно преминаването от процес (activity) в друг процес, събитие, за което потребителя остава изцяло в неведение, поради флуидното функциониране на приложението.



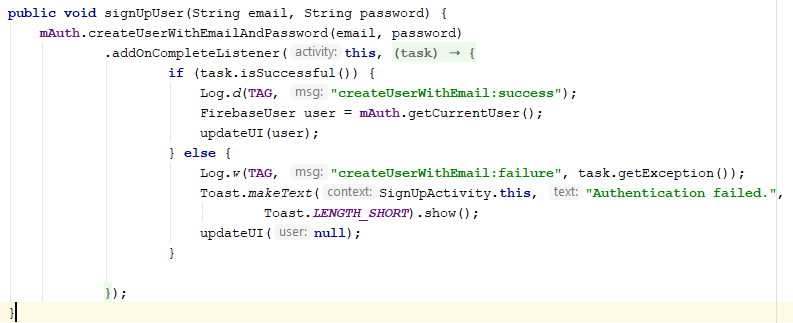
*Фигура 3.5.*

Очаквано, събитието по регистрация на потребителски акаунт (signup), също е изключително интуитивно, а ако не се знаеше, че Firebase е отделен модул, нямаше да го прозрем в процесът на програмиране – поради лекотата на работа с него. Отново, наблюдаваме преминаване от един процес (activity) в друг – след успешен вход, потребителя бива насочен към потребителския панел, в който трябва да попълни лични данни или физически характеристики, сред тях са – Име, полова принадлежност, височина, тегло. Данните се използват в последствие за изчисления и статистика (в смисъла на статистика, за самия потребител, който да наблюдава собствения си прогрес във спортните си занимания).



*Фигура 3.6.*

Успешната или неуспешната регистрация се обработва отново от Firebase модула, в случай на успешна регистрация, данните на новия потребител се записват в базата данни на Firebase (mAuth.createUserWithEmailAndPassword). Любопитно е как модула Firebase, успешно предава и приема информация към контекста на процеса (Activity) .



*Фигура 3.7*

Кодът във фигура 3.8 се грижи за попълване на личните данни на потребителя , които в последствие ще се използват за изчисления, и като условия, с които се разбира дали потребителя постига, предварително зададените от него цели.



*Фигура 3.8*

Информацията за целите на потребителя и въвеждането на данни е максимално опростена, каквато е и ситуацията със структурата на базата данни – тази архитектура цели по-прости, а от там бързи заявки към базата данни. Тъй като скоростта на работа на приложенията е критична в сферата на мобилните приложения. Отново модулът Firebase комуникира не с основния процес (Main Activity), а с конкретния процес – визуализиран от екрана Информацията и целите на потребителя.



*Фигура 3.9*

## 3.3. Упражнения с характеристики и история със серии и повторения.

В следващия отрязък от програмен код наблюдаваме процеса по въвеждане на информацията за хронологията на упражнения, тази информация е необходима, за да се попълва дневника за упражнения на всеки потребител, който в последствие да бъде видим за останалите потребители, които той е посочил, като приятели. Данните ще бъдат подреждани по хронологичен ред, като в най-горната част на екрана, ще бъдат визуализирани, най-скорошните упражнения на потребителя, а в най-долната част ще са най-старите му дейности. В програмния код използвам и валидация, за да подсигуря приложението от неправилно въведени или форматирани данни, които потенциално биха довели до грешки в изпълнението на програмния код, а от там и до видими за потребителя грешки и дефекти.



*Фигура 3.10.*

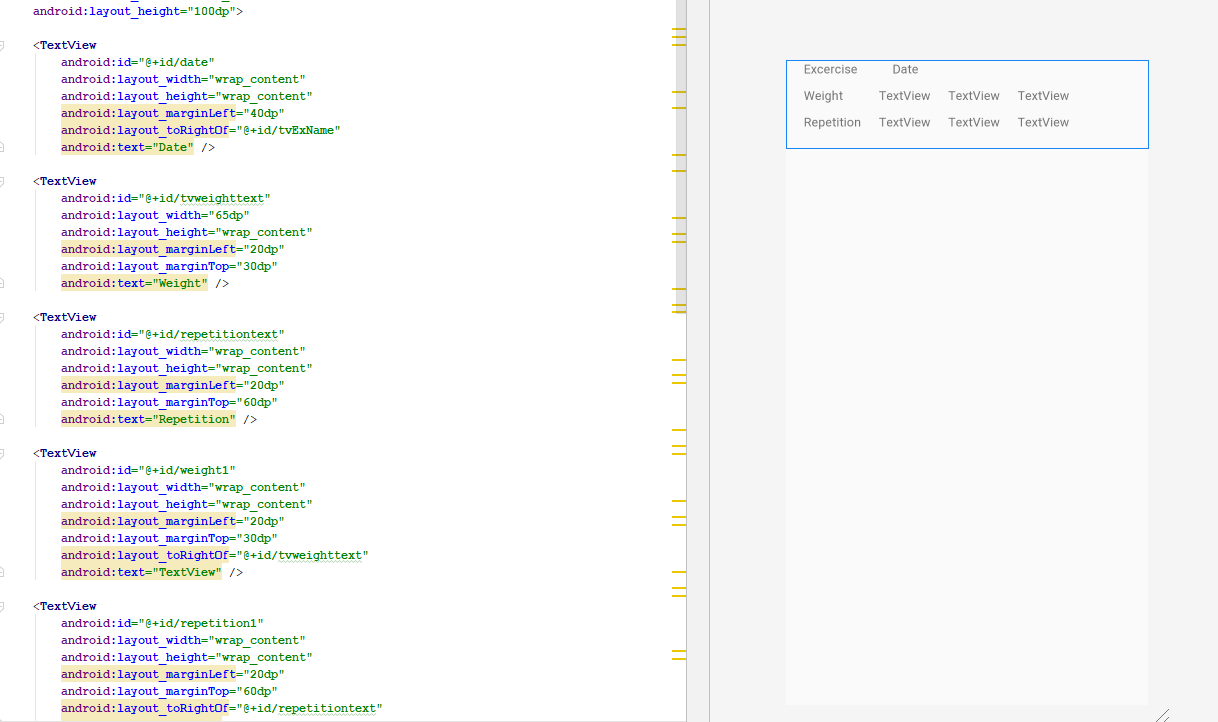
В следващия отрязък от програмен код разглеждаме извличането на информация за екрана с хронология с упражненията на потребителя. Данните се валидират, за да подсигурим коректното действие на приложението. Структурата на упражненията има два параметъра – маса, с която се извършва упражнението и броя повторения. Опционална характеристика е серия, а именно колко серии от конкретно упражнение е извършил потребителя.



*Фигура 3.11*

В този отрязък от програмен код се осъществява управлението на процеса по освобождаване на паметта от обектите, които вече не се използват – чрез изграждането на специален модул – RecycleView.

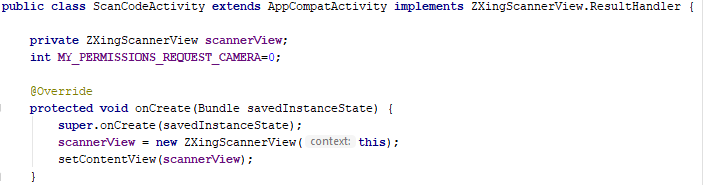
  
*Фигура 3.12.*



*Фигура 3.13*

## 3.4 Сканиране на баркод на стока

Функцията за сканиране, чрез камерата на продукти, всъщност използва съществуващата библиотека, за използване на камерата на мобилното устройство, като скенер – ZxingScanner View(Фигура 3.14).



*Фигура 3.14*

Скенера след това връща сканирания баркод като string, който програмата обрадотва и сравнява с вече съществуваюата база данни от продукти.

# ЧЕТВЪРТА ГЛАВА

# РЪКОВОДСТВО ЗА ПОТРЕБИТЕЛЯ

## 4.1 Изисквания за стартиране на приложението

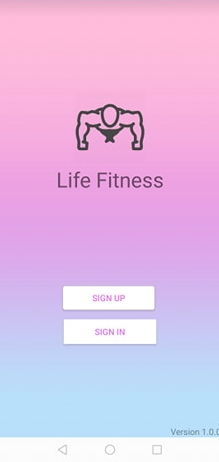
Единствено се изисква наличието на интернет и Android устройството

с минимална версия 9.0 Pie (Api level 28). Това е необходимо за нормална работа с приложението.

## 4.2. Инструкции за ползване

### 4.2.1. Начален екран

При стартиране на приложението потребителя ще види началния екран с логото на приложението и два бутона - единия за регистрация, а другия за влизане в приложението(Фигура 4.1).



*Фигура 4.1*

### 

### 4.2.2. Екран за регистрация

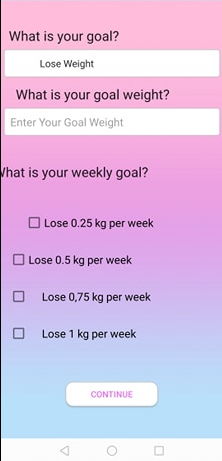
При натискане на бутона за регистриране приложението ще отвори екрана за регистрация. Там се изисква да се въведат парола, имейл и потребителско име. При успешна регистрация потребителя ще екран подобен на Фигура 4.2.



*Фигура 4.2*

Този панел съдържа личната информация на потребителя, нужна за изчислението на необходимите калори . Може да избере своят пол при натискане на бутона под съответния портрет. При натискане на второто поле ще се отвори падащо меню, от което може да се избере нивото на активност. Същото е и при последното поле, където може да избере и своята височина.

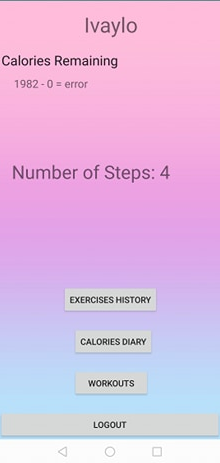
След като е въведена информацията при натискане на бутона „Продължи“се отваря следващият панел, в който се въвеждат главната информация за изчисление на калории. На фигура 4.3 се вижда структурата на екрана и какво трябва да бъде въведено.



*Фигура 4.3.*

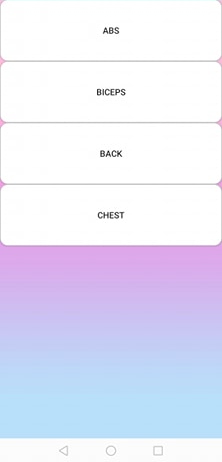
В първото поле се определя целта – дали ще се свалят килограми, качват или поддържат. В зависимост кое от трите избора е избрано по-долната информация ще се промени. Ако потребителя е избрал да сваля килограми до първата отметка ще пише „Lose 0.25 kg per week” , но ако избере да качва, до не ще пише „Gain 0.25 kg per week”. При приключване на попълването на данните потребителя може да натисне бутона „Продължи“, което ще го прати в основния екран.

След регистрирането или влизането, с вече съществуващ акаунт, се отваря главния екран(Фигура 4.4.), в който има крачкометър и вече изчислените калорий. Има няколко бутона, които водят към тренировки, история за проведени упражнения или дневник за калории.



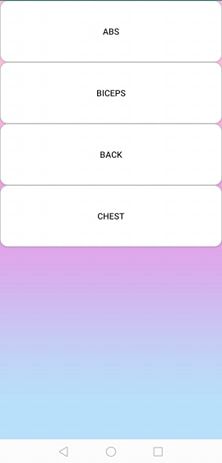
*Фигура 4.4.*

При натискане на бутона за тренировки се показва екран(Фигура 4.5.), на който има видовете упражнения(за гърди, гръб, корем и т.н.).



*Фигура 4.5.*

При избирането на някое от тях се отваря екран(Фигура 4.6.) с упражненията, които са от съответния вид избран в предишното Activity.



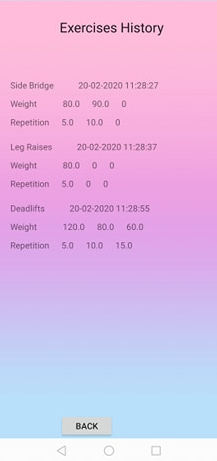
*Фигура 4.6.*

При избиране на упражнение се отваря екран(Фигура 4.7) със снимка на начина на правене му и две полета – тежест, с която си правил серията, и повторения(колко пъти си направил упражнението).



*Фигура 4.7.*

При въвеждането им и натискането на бутона „Добави“ информацията за повторения и тежест се записва с базата данни на Firebase. След което ако се натисне бутона за история на упражненията, който се намира в главния екран, ще се покаже записаната тренировка както е изобразено във фигура 4.8.



*Фигура 4.8.*

# Заключение

В заключение можем да кажем, че фитнес приложението би било полезно на всички, които се занимават или ще започнат да се занимават с физически дейности . Това е първото приложение, което правя на Android Studio, но макар трудностите в началото сега мога да твърдя, че съм се запознал в голяма част с това IDE и с неговите функционалности. Изисквания в по-голяма си част са спазени.

Бъдещото развитие се състои в това да се подобри интерфейса на приложението като да се добавят различни анимации, текстури и други.

Друго нещо, което може да направи работата с приложението по приятна и комуникативна, е да се добави секция за постове и коментари на различни потребитили.

# Източници

1. Android developer guide.
2. Firebase documentation.

# Съдържание

[Увод 4](#_Toc33094686)

[ПЪРВА ГЛАВА 5](#_Toc33094687)

[Обзор на съществуващите решения. Преглед на технологии подходящи за изпълнение на дипломната работа. 5](#_Toc33094688)

[1.1. Съществуващи приложения 5](#_Toc33094689)

[1.1.1. MyFitnessPal 5](#_Toc33094690)

[1.1.2. Fitness & Bodybuilding 6](#_Toc33094691)

[1.2. Технологии и средства за развой 7](#_Toc33094692)

[1.2.1. Android Studio 7](#_Toc33094693)

[1.2.2. IntelliJ IDEA 8](#_Toc33094694)

[1.2.3. Basic4Android 9](#_Toc33094695)

[1.2.4. Java 11](#_Toc33094696)

[1.2.5. Kotlin 13](#_Toc33094697)

[1.2.6. Firebase 14](#_Toc33094698)

[Втора Глава 15](#_Toc33094699)

[Функционални изисквания към приложението. Проектиране на фитнес приложение 15](#_Toc33094700)

[2.1. Функционални изисквания към фитнес приложението 15](#_Toc33094701)

[2.2. Съображения за избор на програмни средства и развойната среда 17](#_Toc33094702)

[2.2.1. IDE – Android Studio 17](#_Toc33094703)

[2.2.2. Firebase 17](#_Toc33094704)

[2.2.3. Език за програмиране – Java 18](#_Toc33094705)

[2.3. Принцип на софтуерния модел 18](#_Toc33094706)

[2.3.1. Структура на приложението 18](#_Toc33094707)

[ТРЕТА ГЛАВА 19](#_Toc33094708)

[Програмна реализация на мобилно фитнес приложение 19](#_Toc33094709)

[3.1. Създаване на първоначалния екран 19](#_Toc33094710)

[3.1.1 Струтуроопределящ елемент на приложението. 21](#_Toc33094711)

[3.2. Поддръжка на различни акаунти и начин на регистриране. 22](#_Toc33094712)

[3.3. Упражнения с характеристики и история със серии и повторения. 28](#_Toc33094713)

[3.4 Сканиране на баркод на стока 32](#_Toc33094714)

[ЧЕТВЪРТА ГЛАВА 32](#_Toc33094715)

[РЪКОВОДСТВО ЗА ПОТРЕБИТЕЛЯ 32](#_Toc33094716)

[4.1 Изисквания за стартиране на приложението 32](#_Toc33094717)

[4.2. Инструкции за ползване 33](#_Toc33094718)

[4.2.1. Начален екран 33](#_Toc33094719)

[4.2.2. Екран за регистрация 34](#_Toc33094720)

[Заключение 41](#_Toc33094721)

[Източници 42](#_Toc33094722)

[Съдържание 42](#_Toc33094723)