



## FitNote



### אפליקציה למשך אחורי פעילויות ספורטיביות

מגיש : אלון בן דוד

ת.ז. : -----

המנחה : ניצן

חולפה : תכנות טלפונים ניידים

תאריך הגשה : 20/05/2021

## **תוכן העניינים**

5.....	מבוא
7.....	מדריך לתוכנה
7.....	אלגוריתמים מרכזיים
7.....	השימוש במחלקה PictureFileHelper כדי לשמר תמונה בטלפון :
9.....	השימוש בפעולה onActivityResult כדי לקבל מידע מאפליקציה אחרת בטלפון :
10.....	שימוש במתאים להציג נתונים ברשימות :
11.....	תאגרים מרכזיים .....
11.....	בדיקה התראות באפליקציה :
11.....	שמירת תמונות בטלפון בקבצים :
11.....	הגדרת התרשימים בסיס הטבלה :
12.....	תהליך בניית בסיס הנתונים עם העוזר :
13.....	תcn .....
13.....	ע"ז מודולים .....
13.....	תרשים Use Case .....
14.....	תרשים מחלקות .....
15.....	Activities .....
36.....	דוגמא לשילוח הودעה ב-WhatsApp .....
42.....	FeedbackActivity .....
43.....	Services .....
45.....	Content Providers .....
48.....	Broadcast Receivers .....
51.....	שילוב תcn (design) .....
54.....	Resources .....
59.....	קבצים .....
59.....	SQLiteDataBase .....
59.....	קובץ info פנימי בזיכרון הטלפון .....
60.....	סכמת Database .....
62.....	מדריך משתמש .....
62.....	מטרת המערכת .....
65.....	יכולות המערכת .....
66.....	תפעול המערכת .....
66.....	תרשים זרימה בין המרכיבים .....
67.....	הרשאות .....
67.....	דרישות מיוחדות ו מגבלות .....
68.....	רפלקציה .....
69.....	ביבליוגרפיה .....
70.....	נספחים .....
70.....	תיעוד – קוד הפרויקט .....
70.....	עזרה ב בניית בסיס הנתונים ( DataBaseHelper ) .....
92.....	דוגמא למתאים (Adapter) .....

95	מחלקה עוזר לשימירה ולקיחת קבצי תמונות (PictureFileHelper)
97	מחלקה להשמעת התראה בעזרת BroadcastReceiver (ReminderBroadcast)
CountDownTimer ו MediaPlayer	מחלקה להשמעת מוזיקה והפעלת טיימר בעזרת Service (MusicAndTimerService)
99	מסמכים נוספים
102	כל המחלקות והקבצים בפרויקט
114	AndroidManifest
116	Java Folder Classes
116	AddExerciseActivity
121	Admin
122	ContactListAdapter
124	ContactModel
125	DataBaseHelper
147	EditUserActivity
154	ExecuteExerciseActivity
158	Exercise
160	ExerciseAdaptor
161	FeedbackActivity
164	InformationActivity
167	MainScreenActivity
170	ManagerUsersActivity
175	MusicAndTimerService
178	PictureFileHelper
180	ProgramUserActivity
187	RegisterActivity
195	ReminderBroadcast
197	SettingsActivity
209	ShareActivity
214	Song
215	StatisticsActivity
223	User
226	UserExercise
228	UserExerciseDoneAdapter
230	UserExerciseNotDonteAdapter
233	UserExercisesInnerJoinEx
235	UsersAdapter
238	ViewExercisesResultsActivity
242	WhatsAppSendActivity
248	res Folder
248	drawable
264	layout
312	menu

314 .....	mipmap
314 .....	ic_launcher file
314 .....	ic_launcher_round file
314 .....	raw
315 .....	values
317 .....	Grable Scripts

## מבוא

### הרקע לפרויקט :

במקרים רבים בזמן האימונים שאני עושה אני מרגיש את הצורך להשתמש באפליקציה פשוטה אחת ויחידה שתאפשר לי לעקוב אחרי הביצועים שלי במגוון תרגילי ספורט. היה לי רצון שהאפליקציה תהיה כזו שתסייע ותעוזד את המשטמש להתקדם באימונים שלו. המשטמש יוכל בקלות להבין את מצבו ביחס לביצועים קודמים ולהבחן במדויקות של שיפור/האטה באופן ברור. זאת דרך ג害羞ים עמודות ופרמטרים הקשורים לתרגיל. מעבר על חנות האפליקציות הבנאי שהרעיון הזה יכול לסייע לי ולרבים. חלק גדול מהאפליקציות שראיתי בנושא ניהול אימוני הקשור מסווגות ועמוסות מדי במידע ובפרטים.

בפרויקט זה החלטתי ליצור מקום נוח גמיש ומסודר שבו ניתן לעקוב אחרי מגוון פעילותות הגוף שונות. המטרה היא כתוב אפליקציה להכנת אימון כושרiesel ולמעקב אחרי האימון. האפליקציה תשמש ככלי לארגון מספר תרגילים למשתמש לבצע. המשטמש יוכל לבצע את התרגילים מתי שהוא רוצה, באיזה אופן שהוא חושב לנכון והוא יקבל התראות לחזור לאמון רק בזמןים שהוא עצמו קובע. למשטמש באפליקציה שליטה מוחלטת על אופן האימון שלו.

### תהליכי המחבר :

התנסות באפליקציות קיימות, וברחבי הרשות. צפייה סרטונים ב-YouTube.

### סקירהו המצב הנוכחי בשוק :

נכון לעכשו בשוק האפליקציות עמוסות במידע ורובם הגדול מתמקד בחזוק שרים שקל לראות כמו חזה, כתפיים ורגליים. אפליקציות כאלה, בעיני, הופכות את האימונים להרבה פחות מענים מפני שבדרך למשתמש שביצועם הוא רק למען שיפור המראה החיצוני. בנוסף, עומס המידע, הפתורים ואפשרויות עלול להפוך את השימוש באפליקציה ללא נוח בכלל. לכן, אני סבור שיש לשנות מצב זה באמצעות פיתוח אפליקציה חדשה יידידותית יותר.

### אילו חידושים יש בפרויקט :

הפרויקט מאפשר למשטמש לנחל אימון פשוט, ברור וgemäßיש מאוד ללא תלות בדבר. זאת בכך שהוא רוצה לנחל אותו. המשטמש יוכל בקלות לעקוב אחרי ביצועיו, לראות מה מצבו ולהבחן במדויקות של שיפור/האטה באופן ברור. כל זאת למען מגוון מטרות שיש למשטמש ולא בהכרח כדי להיראות "יפה" יותר.

### אתגרים מרכזיים :

אתגר מרכזי אותו נתקלתי הוא שבירת מידע על מגוון תרגילים שונים באופן כללי כך שאוכל להתיחס לכלם כאלו תרגילים במחלקה אחת בלבד. למשל, صحיה נמדדת במטרים ושכבות שמייה בכמות הירידות והעלויות. על מנת להתמודד עם אתגר זה הגדרתי מאפיינים כלליים אמורים לאפשר לתאר כל סוג של תרגיל כושר וכך יהיה פשוט יותר לשמור מידע על התרגילים בסיסי הנדרש ולאחר מכן המשטמש. הגדרת הטבלאות עם הדרישות שהיו לי הייתה מأتגרת. היה צורך להפריד בין תרגילים יבשים עם מידע בסיסי כמו תמונה ותיאור של התרגיל ובין תרגילי משתמש בהם יש גם את שם המשתמש, את ה-ID של התרגיל ומידע על ביצוע התרגיל (תאריך, זמן ביצוע, דירוג ומספר החזרות).

### **הסיבות לבחירת הנושא:**

בחرتني לכתוב אפליקציה בנושא אימוני כושר מפני שהייתי סבור שיש לפתח אפליקציה פשוטה ו互動ית לאקטיבית למתאמנים רבים שתספק מידע רב בקלות. עם הכוח הרב שיש בהתעסקות בטבלאות בסיס נתונים ב-SQL הייתה לי היכולת להחזיק בטוחה הארוך מגוון רחב של מידע בכל מיני סוגים תרגילים. באופן זה משתמשים יכולים לבדוק במוגנות של השתרונות/האטיה באופן ברור ומקיף על אימונים שהם עשו במשך שנים. בנוסף, אני מאמין שאפליקציה בנושא אימוני כושר דורשת מגוון רחב של כלים שיעזרו למתאים לנתח את אימונו. ישומון שכזה דורש שימוש במגוון כלים שיש ב-android studio כמו טבלאות, רשימות, טימר, ContentProvider ו-BroadCastReceiver על מנת לנתח את כל המידע על הא蒙ים בצורה המקופה והברורה ביותר. בחירת נושא זה עוזרת לי ללמידה על הרבה כלים חדשים ושימושיים רק מבנית אפליקציה אחת.

### **מוטיבציה לעבודה:**

מפני שאפליקציית אימוני כושר יכולה לכלול כל כך הרבה פיצרים, במהלך העבודה לא הרגשתי פעם אחת שחרר מה להוסיף לאפליקציה. אפשר לראות ו לנתח את אותם תוכאות האימונים בכמה דרכים (טבלאות, רשימות). כמו במרבית אפליקציות הקשורות הקימות, ניתן גם לשתף אחרים בהישגי התרגילים שבוצעו. תהליך האימון אינו סופי ותמיד אפשר לחזור ולהתאמן. לכן, אני רואה באפליקציה זו ארגז חול גדול למשימוש מגוון כלים בסיסים ומתקדמים של android studio, ומסיבה זאת, לא היה רגע אחד בו לא הייתה לי מוטיבציה להוסיף, להרחיב, ללמידה ולשפר את האפליקציה שלי.

### **על איזה צורך הפרויקט עונה? איזה פתרון הפרויקט הזה בא לתת?**

הפרויקט מהווה תוכנה המאפשרת למשתמש להכין אימון כושרiesel לעצמו ולעקב אחר הביצועים שלו באמצעות ביצוע התרגילים ושמירת הנתונים על הביצוע בתוכנה. למשתמש יש אפשרות להוסיף תרגילים לרישימת התרגילים לביצוע. הוא יכול לבצע את התרגילים באימון בכל סדר שירצה. סדר ביצוע התרגילים באימון דינמי כדי למנוע הגבלה על המתאמן. למשל, אם המתאמן נמצא בחדר כושר וההיליכון תפוס ע"י מתאמן אחר, הוא יוכל לבחור תרגיל אחר באימון שלא דורש שימוש במתקו זה.

המידע עליו מתבצע המעקב באפליקציה הוא רק על תרגילים שבוצעו באופן מלא בתהליך הביצוע הכלול. המעקב על ביצוע התרגילים מटבṭא במסק הגרפים בו מוצג גרפ' עמודות דינמי המציג את הנתונים על הביצוע התרגיל: זמן שלקח ביצוע התרגיל, תאריך ביצועו, דירוג המשתמש לקושי הביצוע ומספר החזרות בתרגיל. נוסף על כך התוכנה מכילה את נתוני המשתמש: גיל, גובה, משקל. כך היא מחשבת את תוצאתה BMI שלו ומציגה בפניו את מצבו הנוכחי.

## מדריך למתכנת

### אלגוריתמים מרכזיים

#### השימוש במחלקה PictureFileHelper כדי לשמר תמונה בטלפון:

באפליקציה יצרתי מחלקה שטandardה לסייע לי בשימירת תמונות משתמשים.  
אני משתמש באפליקציה בקובץ שמכומש על ידי SharedPreferences בקובץ info בטלפון לשימרת תמונה  
משתמשים באפליקציה.  
מחלקה זו יש שלושה פעולות:  
פועלה ראשונה:

```
public static void writeFileToInternalStorage(Context context, Bitmap bitmap,
String filename)
{
    SharedPreferences sp= context.getSharedPreferences("info",0);
    int counter=sp.getInt("counter", 0);
    try {
        FileOutputStream os = ((Activity)context).openFileOutput(filename+counter,
Context.MODE_PRIVATE);
        //Here compress 50%, store the compressed data in os.
        bitmap.compress(Bitmap.CompressFormat.PNG,100,os);
        counter++;
        SharedPreferences.Editor editor=sp.edit();
        editor.putInt("counter",counter);
        editor.commit();
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    }
}
```

פועלה שמאפשרת כתיבת קובץ של תמונה משתמש אותה אני רוצה לשמר בקובץ info בטלפון.  
בשמירה אני דוחס את התמונה לפורמט PNG ב-100%, ואת הנתונים אחסן ב-os.  
שם התמונה יהיה שם המתmesh שmagiu עם הקיראה לפעולה זו + מספר מונה.  
באופן זהה, גם אם משתמש יש מספר תמונות משתמש באפליקציה (לא אפשר כי ברישום יכול להיכנס רק פעם אחת משתמש עם שם מסוים) תבחר זאת האחרונה (בהמשך יהיה ברור למה).

פעולה שנייה :

```
public static Bitmap readFileFromInternalStorage(Context context,String filename)
{
    Bitmap b=null;
    try {
        InputStream in = ((Activity)context).openFileInput(filename);
        b= BitmapFactory.decodeStream(in);
        in.close();
    } catch (IOException e) {
        e.printStackTrace();
    }
    return b;
}
```

פעולה לקרוא הקובץ info במכשיר הטלפון ולאחר מכן שיליפת התמונה הרצויה. בעזרה ה-**InputStream** אני ניגש לתמונה עם השם שקיבלנו filename.

כמו שהסבירתי קודם, גם אם יש כמה תמונות עם אותו שם משתמש. לכל תמונה מסוימת מספר ייחודי המעיד על מיקומו בקובץ info וגם בקריאה לפועלה עם שם המשתמש יוחזר התמונה האחרונה בקובץ info. הפועלה מחזירה תמונה שהיא מצאה.

פעולה שלישית:

```
public static Bitmap getPic(Context context, String name)
{
    File mydir = context.getFilesDir();
    File lister = mydir.listFiles();
    Bitmap bitmap=null;

    for (String list : lister)
    {
        if(list.toString().contains(name)) {
            //Toast.makeText(context, list, Toast.LENGTH_LONG).show();
            bitmap = readFileFromInternalStorage(context, list);

        }
    }

    return bitmap;
}
```

הפעולה הזאת מושגה גישה לקובץ המכיל את התמונות ועובדת על כל התמונות. תמונה עם שם המשתמש שקיבלו בקריאה לפועלה תישמר ב-Bitmap. לבסוף תמונה זו תוחזר למשתמש.

בעזרת מחלוקת זו כשאר המשתמש נרשם וambil את עצמו ומנטו נשמרת בקובץ info בטלפון תחת שמו ומספר מונה. כך שבכל מסך שאני רוץ להציג את תמונת המשתמש אני לוקח את שמו מבסיס הנתונים ומחפש בעזרה הפועלה getPic(Context context, String name) את תמונה המשתמש ומראה אותה במסך.

## השימוש בפעולה onActivityResult כדי לקבל מידע מAPPLICATION אחרה בטלפון:

כשהמשתמש נרשם לאפליקציה או משנה את הפרטים שלו, הוא יכול לצלם תמונה שתיהיה תמונה המתשמש שלו. כדי לצלם תמונה משתמש התוכנה מבצעת מספר צעדים :

(1) ליצור Intent שמטרתו היא לצלם תמונה, ואליו אני מצינים את הפעולה שהוא יבצע :

```
//the intents intention is to capture an image
//we have to specify the action for the intent
Intent imageTakeIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
```

(2) עכשו אפשר לבדוק אם טלפון אכן יש אפליקציה כלשהי לצילום תמונה שיכולה לבצע את הפעולה הנדרשת לכך. אם אין כזות פעולה צילום התמונה לא יתרחש :

```
//now we are able to check if there is any application
//capable of handling this intent
//otherwise your'e application will crush
if(imageTakeIntent.resolveActivity(getApplicationContext()) != null)
{
    //we have to user startActivityForResult method
    //second parameter is the requestCode
    startActivityForResult(imageTakeIntent, REQUEST_IMAGE_CAPTURE);
}
```

(3) אם יש אפליקציה כזות אני אקרא ל-`onActivityResult` שתזיכה לדווח שተגעה מאפליקציה שיכולה לצלם. ב-`onActivityResult` יש המתייחס לצילום התמונה :

```
<uses-feature
    android:name="android.hardware.Camera"
    android:required="true" />
```

(4) בקבלת התזאה מהאפליקציה בטלפון לביצוע הצילום, מתבצעת בדיקה נוספת של קוד הבקשת ובודקת נכונות התזאה. לאחר מכן נשמר ב-`Bitmap` התמונה שצולמה.

בעזרת הפעולה `saveToInternalStorage(String username)` עליה `PictureFileHelper.writeFileToInternalStorage` דיברתי קודם במחלקה `PictureFileHelper` האפליקציה שומרת את התמונה בקובץ `info` בטלפון.

```
//in order to receive the result from the other application we
//have to override a method called onActivityResult
@Override
protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {
```

```
//we have to check if the requestCode is the same
//AND if the resultCode is ok
super.onActivityResult(requestCode, resultCode, data);
if (requestCode == REQUEST_IMAGE_CAPTURE && resultCode == RESULT_OK) {
    //here we have the data available on this intent
    //now we can retrieve the data
    Bundle extras = data.getExtras();
    Bitmap imageBitmap = (Bitmap) extras.get("data");

    //now we can display the image on the image view
    imgProfilePic.setImageBitmap(imageBitmap);

    //getting the image to save
    bitmapToSave = imageBitmap;
}}
```

## שימוש במתאים להצגת נתונים ברשימות:

באלפיקציה מימושי מחלקות רבות למתאים היורשים מההמattaם הבסיסי `BaseAdapter` על מנת להתאים את כל הרשימות לצורות שאני רוצה להציג בהם את הנתונים.

המחלקות האלו הם :

- `ContactListAdapter` להצגת אנשי קשר מסווג `ContactModel` ברשימה `SearchView` במסמך `.ShareActivity`
- `UsersAdapter` להצגת משתמשים מסווג `User` ברשימה `SearchView` במסמך `.InformationActivity` להצגת תרגילים מסווג `Exercise` ברשימה `ListView` במסמך `ExerciseAdaptor`
- `UserExerciseDoneAdapter` להצגת תוצאות תרגילים שבוצעו מסווג `UserExercise` ברשימה `.ViewExercisesResultsActivity` במסמך `ListView`
- `UserExerciseNotDoneAdapter` להצגת תרגילי משתמש שלא בוצעו מסווג `UserExercise` ברשימה `.ProgramUserActivity` במסמך `ListView`

תהליך ייצרת מתאם :

- (1) כתיבת ה-XML המתאים לעיצוב האיבר ברשימה `BaseAdapter`
- (2) יצירת מחלקה היורשת מ-`BaseAdapter`
- (3) יצירת המתאים במחלקה בה נמצאת הרשימה
- (4) שילוב הרשימה עם הנתונים והמתאים שייתאים את הנתונים לרשימה  `ListView`.

בעזרת מחלקות אלו אני מנפח לכל איבר ברשימה עיצוב משלו המיעיג באופן ברור ומקיף את כל המידע הרלוונטי לסוג נתון זה. כך המתאים יכול להבין מהם התרגילים באלפיקציה, מהם הנתונים שלו ומה תוצאות הביצועים שלו בתרגילים שביצע.

## אתחרים מרכזיים

### בדיקות התראות באפליקציה:

באפליקציה במאסך התרגילים של המתמש (ProgramUserActivity) המשמש יכול להפעיל התראה מתואמן לזמן שהוא רוצה שהיא תפעל. התראה זו תעוזד את המתמש לחזור להתקמן ותיקח אותו חזרה לאפליקציה. בעת הפעלת התראה לא יהיה ברור ממעבר על מסך ה-RUN אם התראה עומדת לפועל או לא בהצבת זמנים מסוימים. לאחר בדיקה על נושא התראות באינטרנט הבנתי שעלי לבדוק את תקינות התאריך ולודודא שהתאריך שאינו משתמש בו לצורך התראה דרך ReminderBroadcast הוא תקין והוא לאל לפני כרגע. אם התאריך הוא לפני עכשו התוכנה לא תיצור את ה-AlarmManager ליצירת התראה בזמן הנitin ותישלח הודעה מתאימה במקום. אם הזמן תקין והוא אחורי עכשו ה-AlarmManager יבנה.

```
//checking that the time given is not before now
if (alarmStartTime <= System.currentTimeMillis()){

    Toast.makeText(this, "This time is before now! : " + startTime.getTime(),
    Toast.LENGTH_LONG).show();

} else{
    // Set Alarm
    //AlarmManager.RTC_WAKEUP - wakes up the device to fire the pending intent
    //at the specified time
    alarmManager.set(AlarmManager.RTC_WAKEUP, alarmStartTime, pendingIntent);

    Toast.makeText(this, "Set Done! : " + startTime.getTime(),
    Toast.LENGTH_LONG).show();
}
```

### שמירת תמונות בטלפון בקבצים:

לא ניתן לשמר תמונות ב-drawable ברגע שאינו רוצה לשמר תמונות משתמשים. לאחר חקירה של פתרונות אפשריים גילית שאני יכול לעזור בחלוקת נפרדת לשטמון את התמונה המצלומה בטלפון בקובץ בתוך הטלפון עצמו מחוץ לאפליקציה עצמה. לאחר קליטת התמונה שצולמה באפליקציה נפרדת ב-ActivityResult onActivityResult, אני נעזר בחלוקת PictureFileHelper לשמר את התמונה המשתמש עם שמו ומספר מונה. כך נשמר בזיכרון הטלפון בקובץ info תמונה המשתמש. אם נרצה לגשת לתמונה זו נעזר שוב בחלוקת PictureFileHelper כאשר השם שנחפש הוא שם המשתמש מפני שתמונה נשמרה עם שם המשתמש.

### הגדרת התרשים במאסך הטבלה:

הגדרת הצירים של הטבלה במאסך הטבלה (StatisticsActivity) הייתה מאתגרת. כדי להראות את התאריך של ביצוע התרגיל עם נתונים נוספים עבור כל תרגיל בנפרד לא יכולתי להתייחס לתאריך המלא בציר האקס מפני שאז התרשים היה אורך מדי. לכן, כדי לפטור בעיה זו הגדרתי שני ספינרים לסינון המידע בתרשימים: אחד לבחירת סוג התרגיל ואחד לבחירת השנה. באופן זה מיקום בציר ה-x עבר כל איבר בתרשימים מייצג את החודש בו בוצע התרגיל ומיקום בציר ה-y מצינו את מספר החוזות ביצוע התרגיל.

## תהליך בניית בסיס הנתונים עם העוזר:

הדרישות לאפליקציה שלי דרשו בניהלה של מספר טבלאות בסיס הנתונים. התהליך של בניית בסיס הנתונים המתאים ביותר לאפליקציה שלי ולרעיון שלי היה מואט גור.

לאחר מעבר על דרכי שמיירת המידע מצעתה שהשימוש במחלקה DataBaseHelper היורשת מ- SQLiteOpenHelper היא הדרך היעילה ביותר לשימרת מידע וניהולו.

כדי לשמר את המשתמשים של האפליקציה יצרתי טבלה בשם Users השומרת את פרטי המשתמשים.

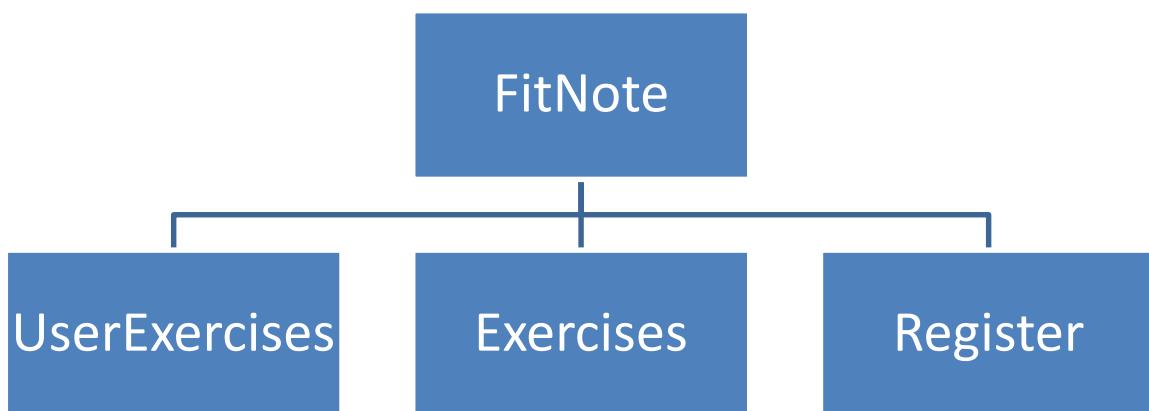
כדי לשמר את התרגילים המתמשכים יצרתי טבלה בשם Exercises השומרת את פרטי התרגילים.

כדי לשמר את תרגילים שבוצעו ונתונים נוספים על ביצוע התרגילים (במידה והתרגיל בוצע).

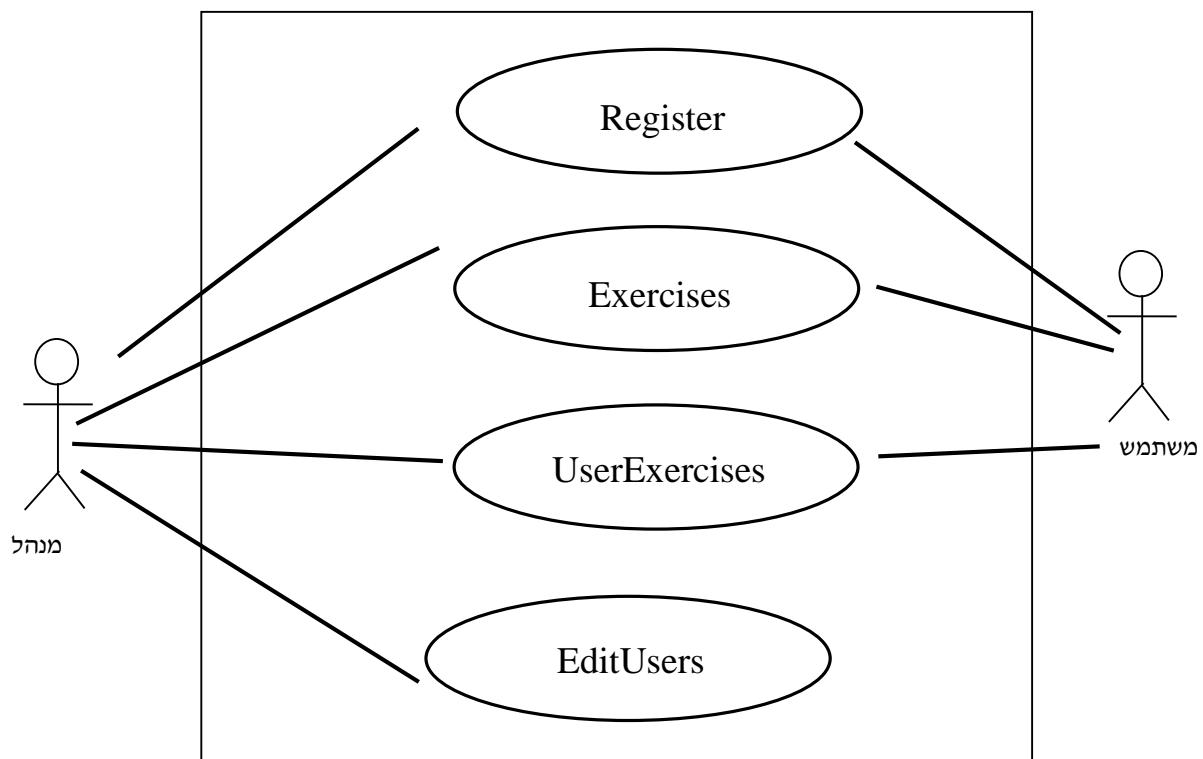
כדי לשמר את השירים של האפליקציה יצרתי טבלה בשם Songs השומרת את פרטי השירים שאנו משתמשים בעת ביצוע תרגילים מסך ביצוע התרגילים (ExecuteExerciseActivity).

## תכלית

### ענמודולים

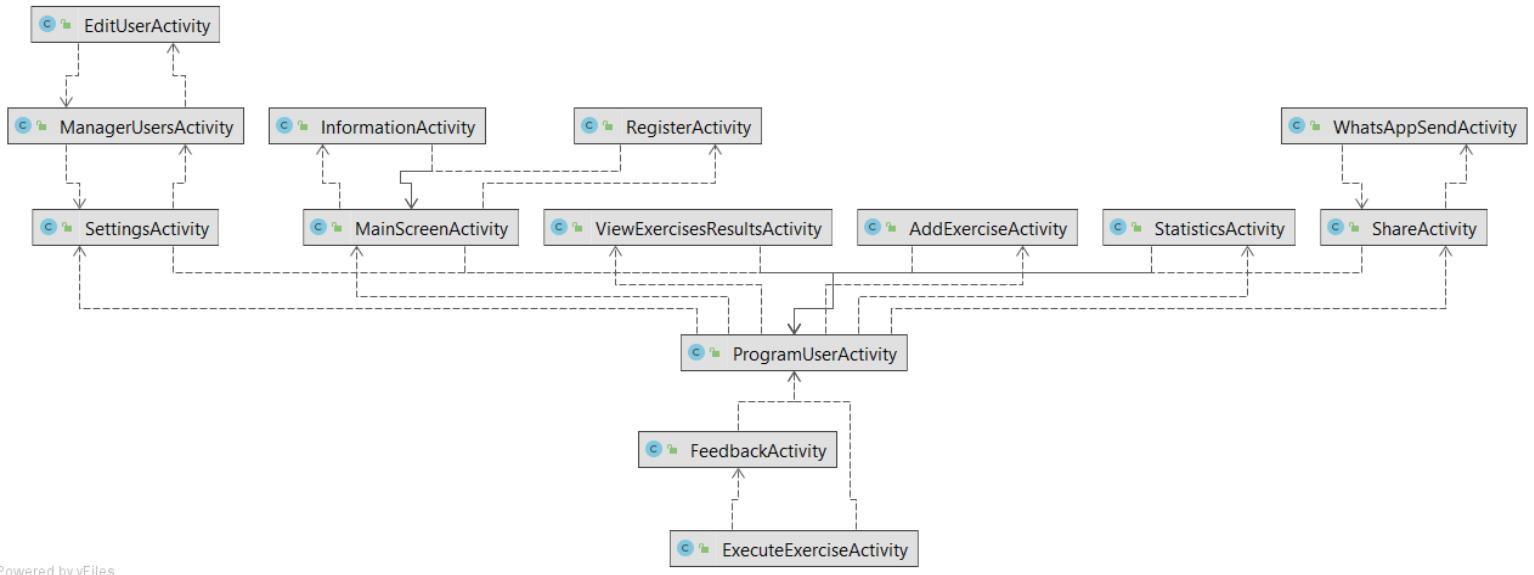


### תרשיים Use Case

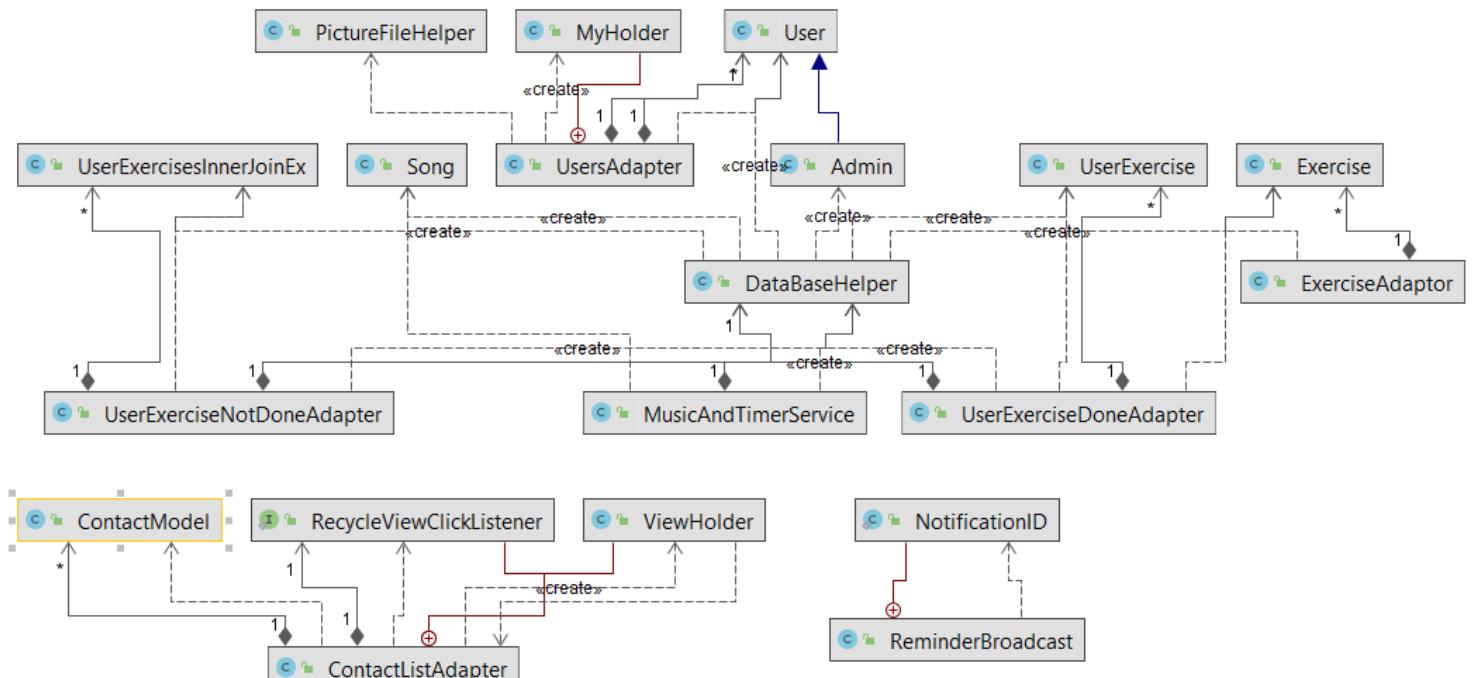


## תרשים מחלקות

תרשים UML של מסכימים (Activitys)



תרשים UML של שאר המחלקות באפליקציה (BroadcastReiever ,Service ,Adapter ועדי)



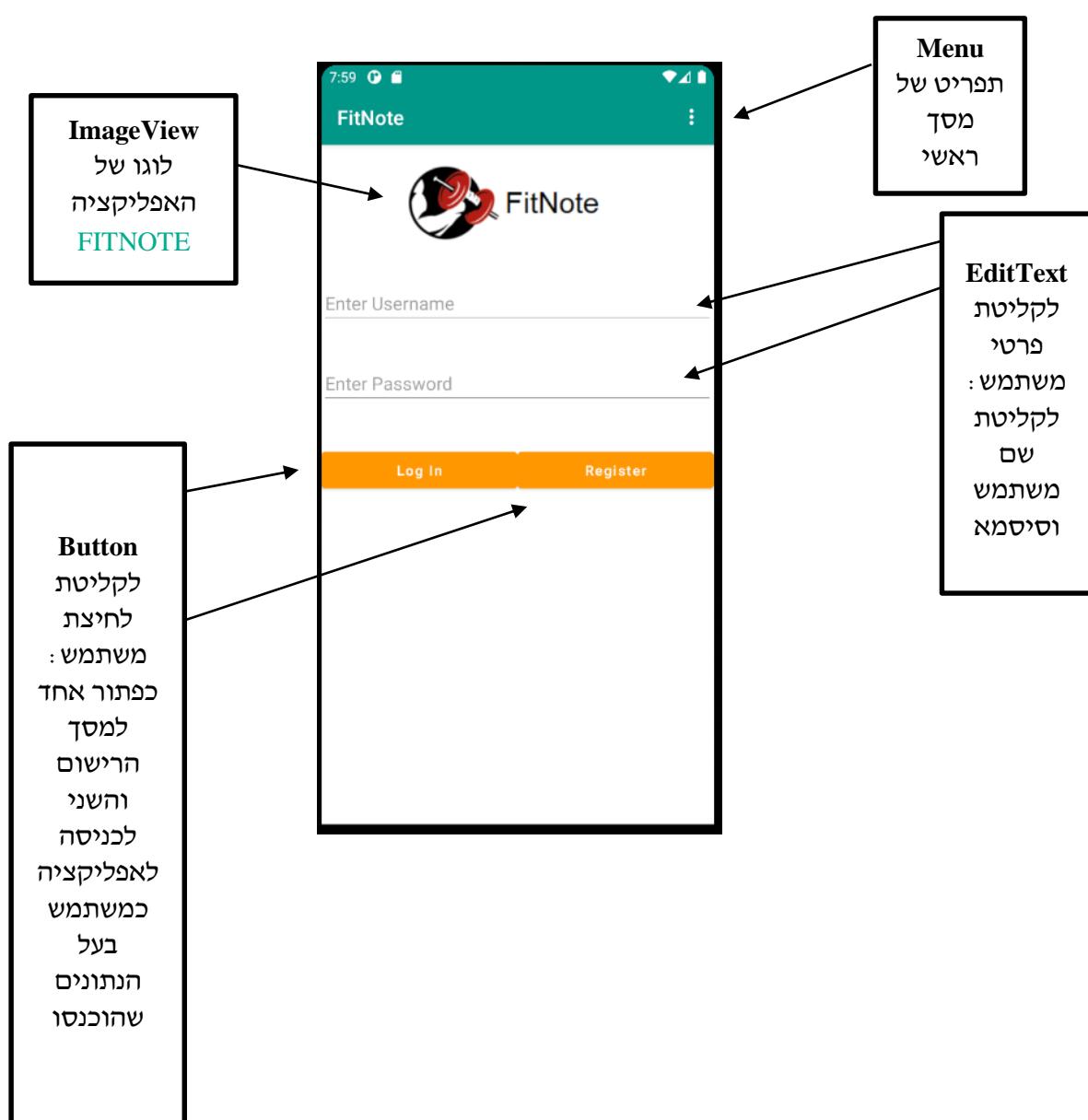
## Activities

### MainScreenActivity

**מסך הפתיחה :** במסך זה מתממש המסך הראשון והראשי של האפליקציה.  
לחיצה על כפתור הרישום מעבירה את המשתמש למסך הרישום **.RegisterActivity**.  
לחיצה על כפתור הכניסה מעבירה את המשתמש למסך התרגילים של המשתמש **.ProgramUserActivity**.  
כולל את הרכיבים הבאים :

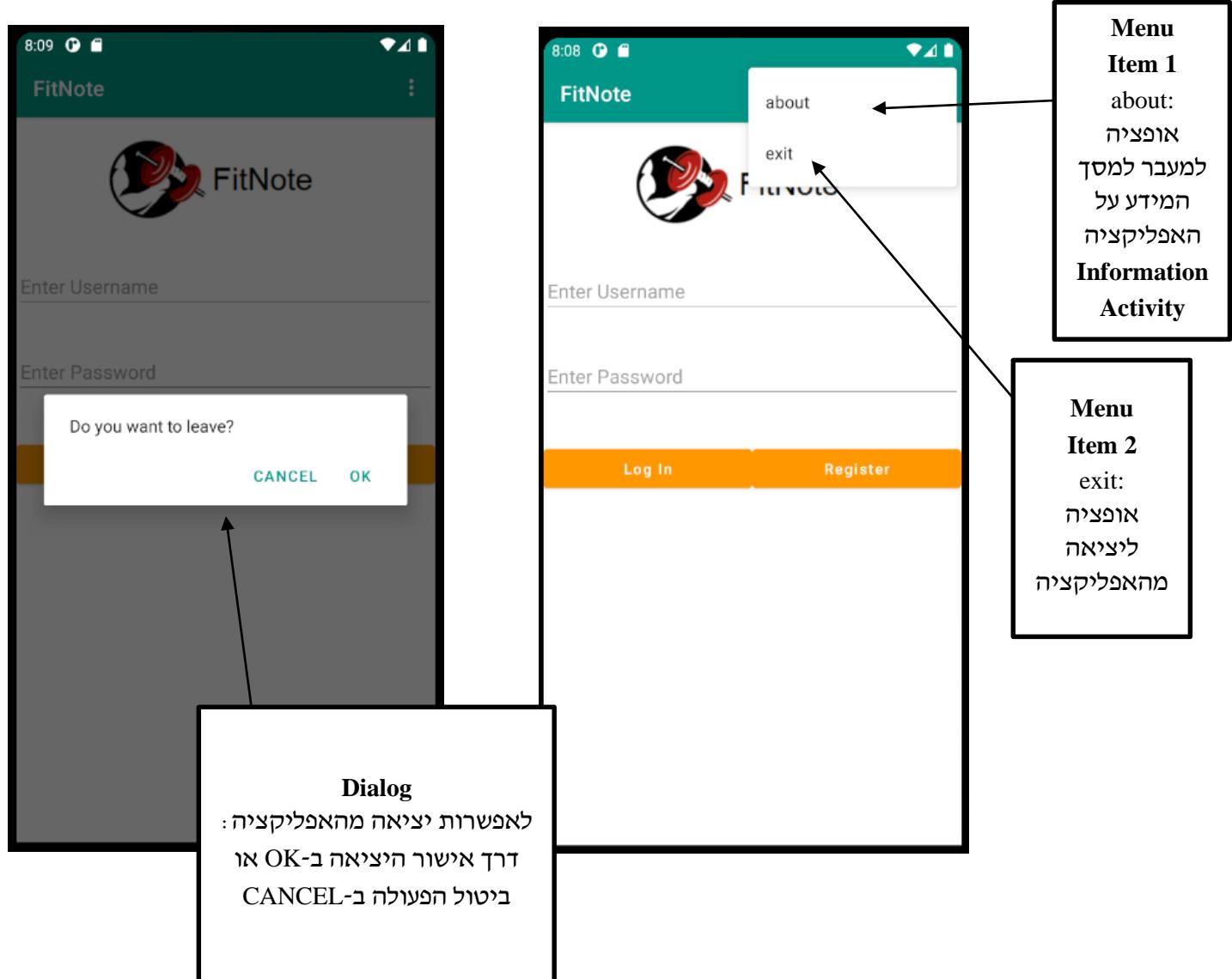
**EditText, Button, ImageView, Menu, Dialog**

צילום מסך ראשי :



תפריט של מסך ראשי כולל אפשרות ללקת למסך מידע על האפליקציה ואפשרות לצאת מהאפליקציה דרך מענה על Dialog של הסכמה לצאת מהאפליקציה.

צילום מסך ראשי עם הופעה של התפריט וה-dialog:



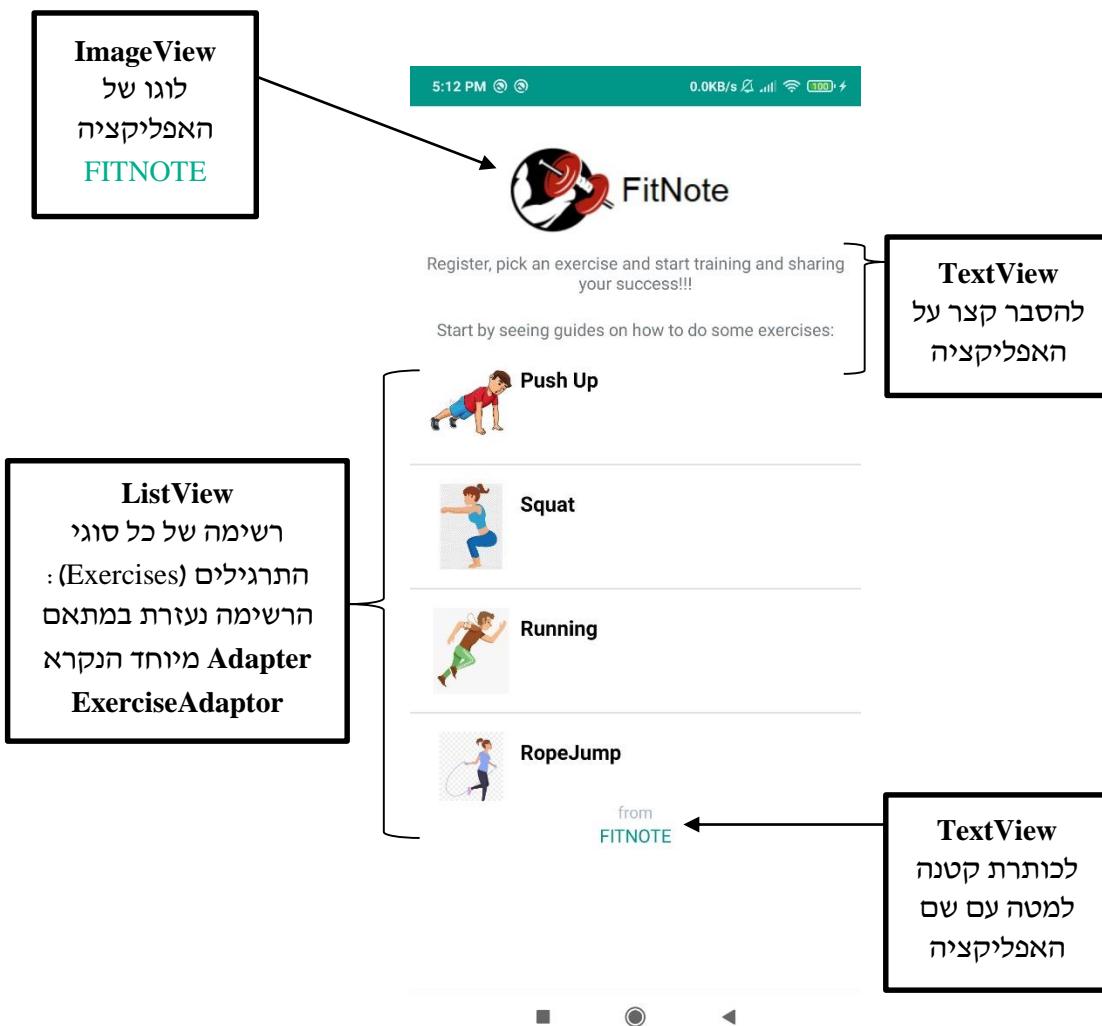
## InformationActivity

**משמעות המידע:** מסך זה כולל הסבר קצר על מה אפשר לעשות באפליקציה.  
ברשימה למטה (**ListView**) מופיעים כל התרגילים שיש באפליקציה כך שגם משתמשים שרק התקינו את האפליקציה יוכל לחוץ על כל תרגיל ולהבין איזה תרגילים קיימים בתוכנה זו.

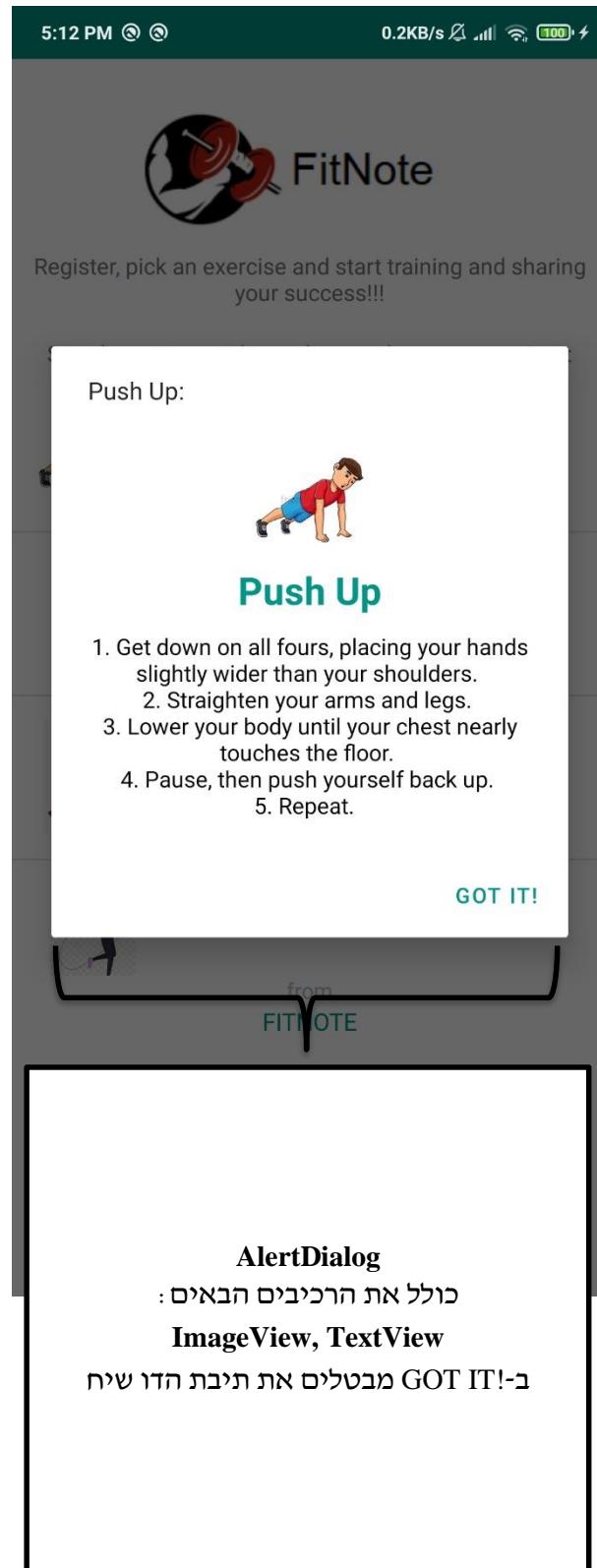
כולל את הרכיבים הבאים:

**ImageView, TextView, ListView**

צילום של מסך המידע:



צילום מסך המידע עם הופעה של dialog :



## RegisterActivity

**מיסוך הרישום :** במסך זה מתבצעת תהליכי הרישום של האפליקציה למשתמשים חדשים.  
**בתהליכי הרישום נמנעת האפשרות להירשם עם אותו שם משתמש יותר מפעם אחת!**

לחיצה על אחד הcptוריהם מטה מחזירה את המשתמש למסך הכניסה למשתמש החדש.  
**MainScreenActivity**

כולל את הרכיבים הבאים :

### ImageView

תמונת של  
משתמש :  
מראה את התמונה  
המשתמש לאחר  
קליטת התמונה

### EditText

לקלייטות פרטי משתמש חדש :  
לקלייטות שם משתמש וסיסמה

### TextView

لتיאור המידע הנוכחי  
מה-SeekBar

### SeekBar

לקלייטות משקל וגובה של  
המשתמש

### TextView

להסביר איזה  
מידע לוקחים  
בcptור

### Button

cptור לקליטת לחיצה של  
המשתמש :  
בלחיצה מופיעה ה-

### DatePickerDialog

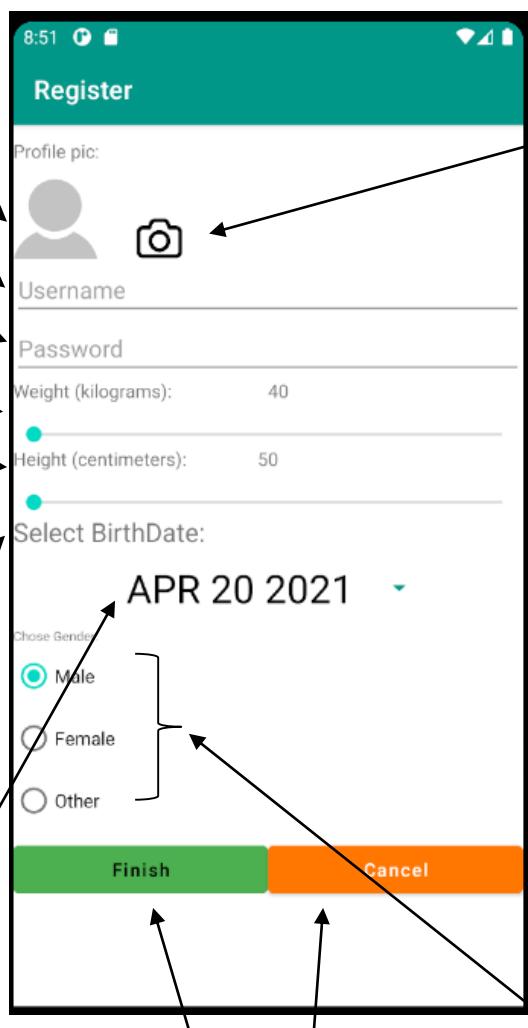
שהוא לוקחים את תאריך  
לידתו של המשתמש.  
התאריך מועבר מה-

### DatePicker

בcptור או מהטקסט  
נקח המידע לבסיס  
הנתונים

### ImageView, Button, EditText, SeekBar, TextView, RadioGroup, RadioButton

צילום מסך ראשוני :



### ImageView

תמונת של מצלמה :  
לחיצה עליה  
פותחת פעולה של  
ליקחת תמונה עם  
המצלמה בטלפון.  
אם אין אישור  
לגשת למצלמה לא  
תהיה אפשרות  
לצלם בטלפון,  
ובהכנותת פרטי  
המשתמש יוכנס  
תמונה הפרופיל  
הברירת מחדל. אם  
המשתמש כן צילם  
את עצמו ווכנס  
לבסיס הנתונים  
התמונה שהוא  
צילם

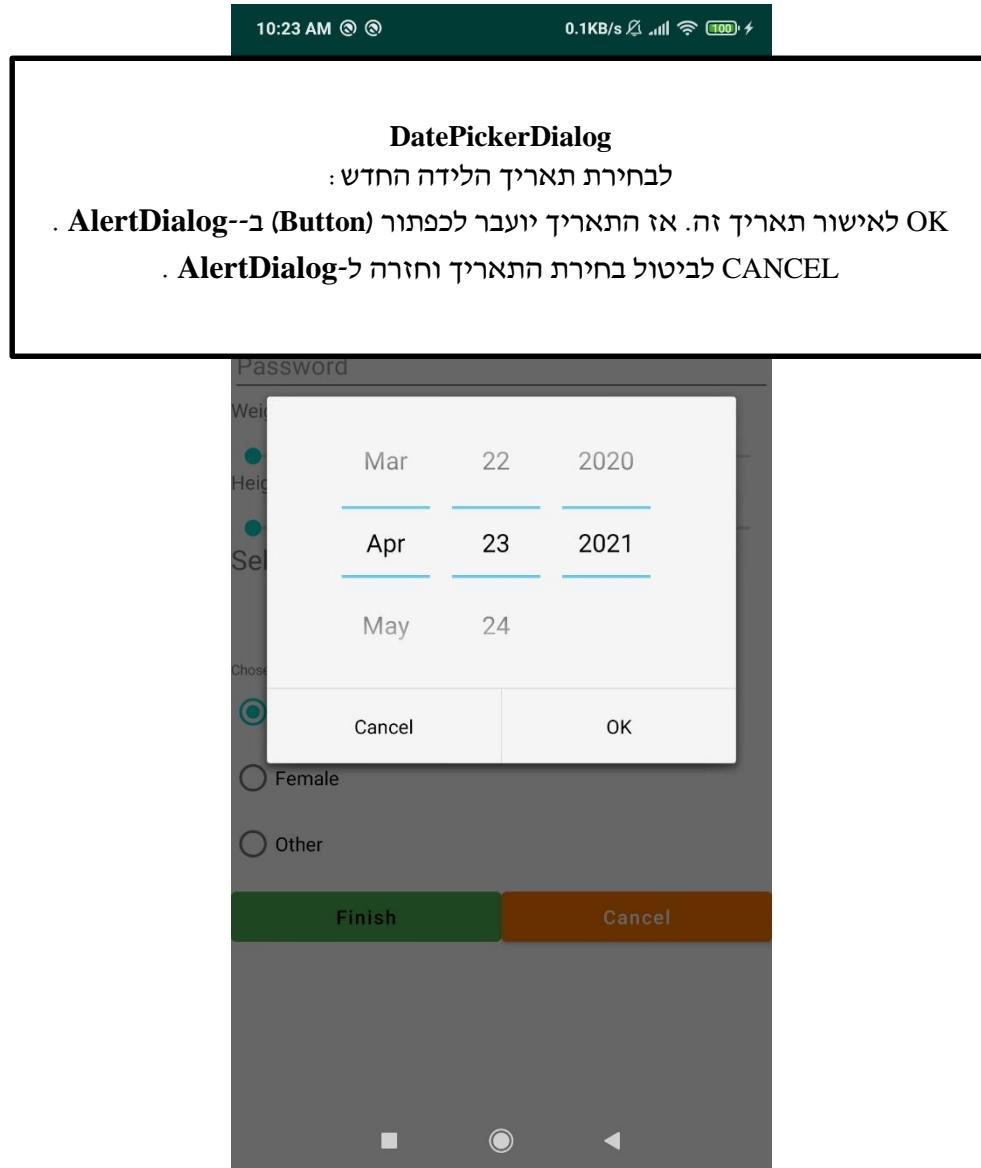
### RadioGroup

שמאגד כמה  
-Buttons  
cptורים אלו  
קובלים את מין  
המשתמש

### Button

cptורים לשימוש הרישום :  
וכcptור לביטול הרישום :  
בלחיצה על שניהם חזררים  
למסך הראשי

בלחיצה על כפתור התאריך במסך הקודם מופיע ה- **DatePickerDialog** שהוא לוקח את תאריך לידו של המשתמש.  
: **DatePickerDialog** צילום של ה-



## ProgramUserActivity

**מיצ' התרגילים של המשתמש :** במצ' זה מופיעים כל התרגילים (UserExercises) של המשתמש אותם הוא עוד לא ביצע.

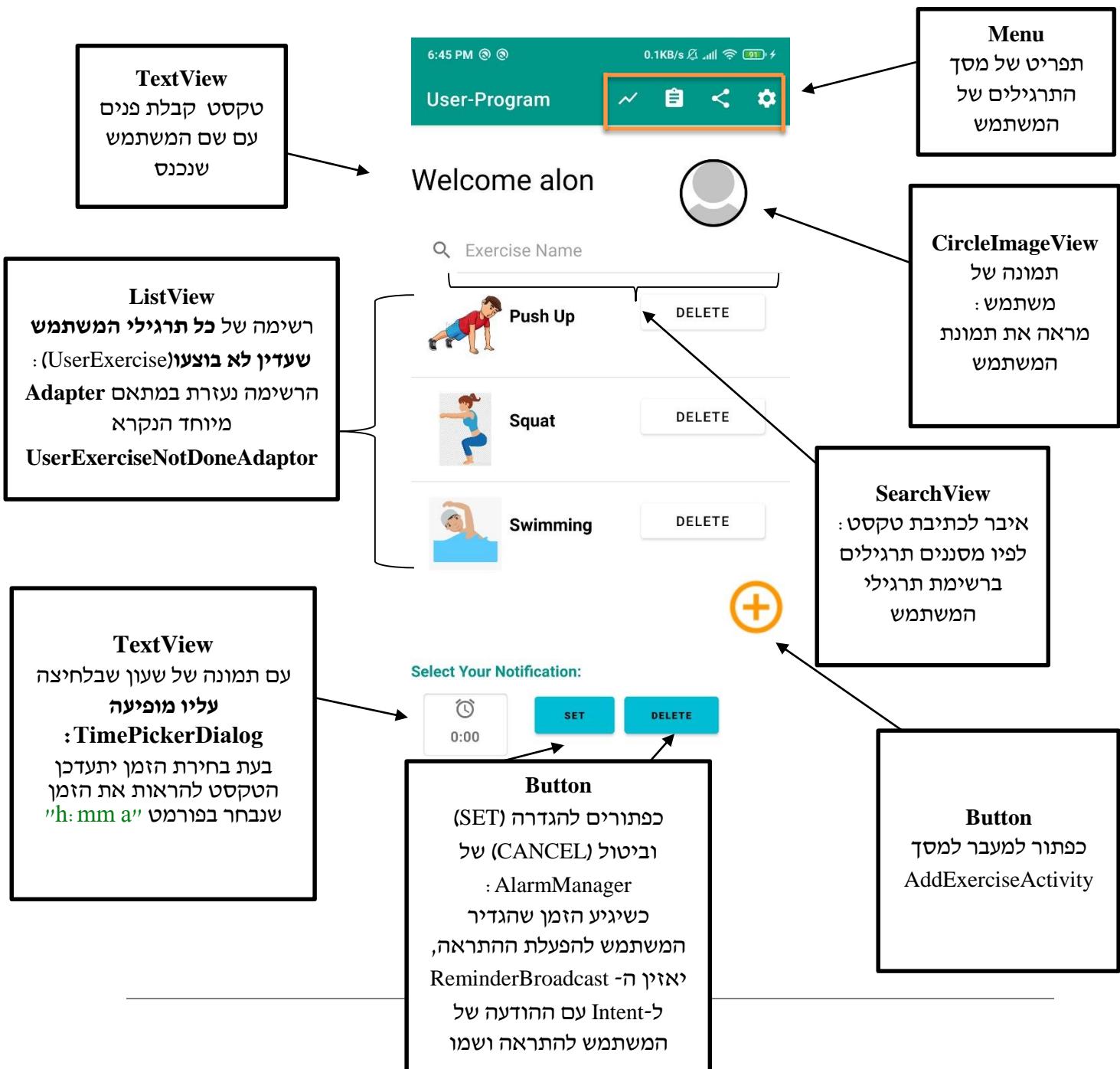
למעלה בתפריט מופיעים ארבעה איקונים למסכים שונים. יש טקסט קבלת פנים למשתמש הנקנס ומוצג תמונה הפורפיל של המשתמש. ברשימה יופיעו התרגילים (UserExercises) אותם המשתמש עוד לא ביצع אך הוסיף אותם לרשימת התרגילים שהוא ביצע (כלומר, אלו תרגילים בלי תוצאות). למטה יש אפשרות להוסיף עוד תרגילים ואפשרות להכין הטראה לזמן מסוים שתעדוד את המשתמש לחזור להתקאה.

לחיצה על כפתור הפלוס לוקחת את המשתמש למצ' הוספה התרגילים .**AddExerciseActivity** כולל את הרכיבים הבאים :

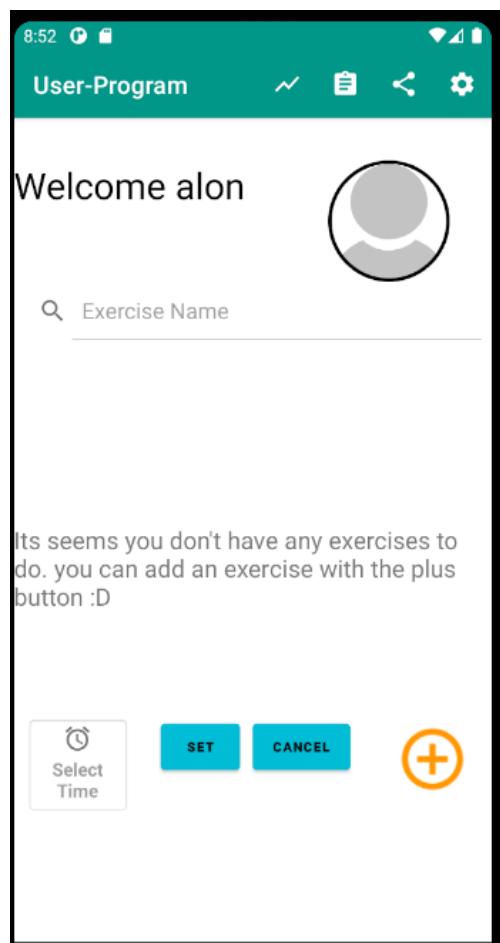
<https://github.com/hdodenhof/CircleImageView> : CircleImageView

**TextView, Button, ImageView, CircleImageView (from a library on GitHub), ListView, Menu, SearchView**

צילום של מצ' התרגילים של המשתמש :

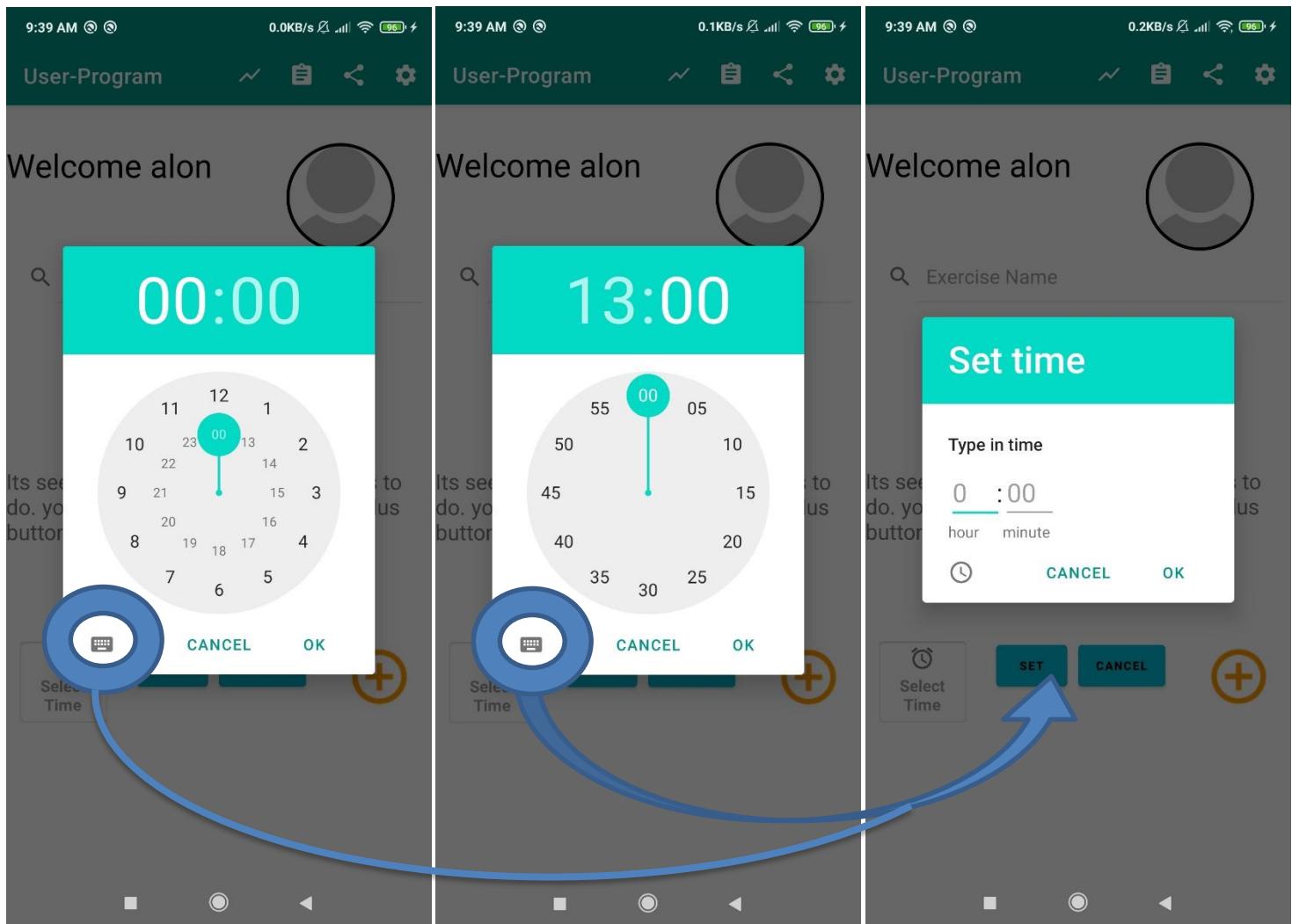


- אם משתמש אין תרגילים לביצוע או שנכנס משתמש חדש, רשימת התרגילים תהיה ריקה ובמקום זאת יופיע טקסט שיסביר איך להוסיף תרגילים לביצוע. לעומת זאת, הtekסט יופיע (יהיה visible) אם DataBaseHelper היה ריק, והרשימה תהיה ריקה אם DataBaseHelper לא נמצא אף תרגיל שהמשתמש לא ביצע.  
כילום של מסך התרגילים של המשתמש כאשר אין תרגילים ברשימה :



אם המשתמש לוחץ על ה-**TextView** מלטה עם תMOVות השעון, מופיעה **TimePickerDialog** בו אפשר לבחור את הזמן (שעות ודקות) להופעה של **Notification** דרך המחלקה **ReminderBroadcast**. לאחר בחירת הזמן ב-**TimePickerDialog**, הוא יופיע ב-**TextView**. לבניית ה-**PendingIntent** בו יש **AlarmManager** לזמן שהגדיר המשתמש ב-**TextView**.

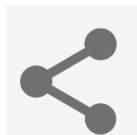
**צילום של TimePickerDialog**



- התפריט של מסך התרגילים של המשתמש כולל מספר אפשרויות :
- 1) לעبور למסך ההגדרות **SettingsActivity** בו המשתמש יכול לשנות את כל הפרטים שלו חוץ משמו.



2) לעبور למסך שיתוף המידע על תוצאות התרגילים **ShareActivity** לאייש קשר לבחירת המשתמש.



- 3) לעبور למסך **ViewExercisesResultsActivity** בו המשתמש יוכל לראות את תוצאות התרגילים שבייצע ברשימה.



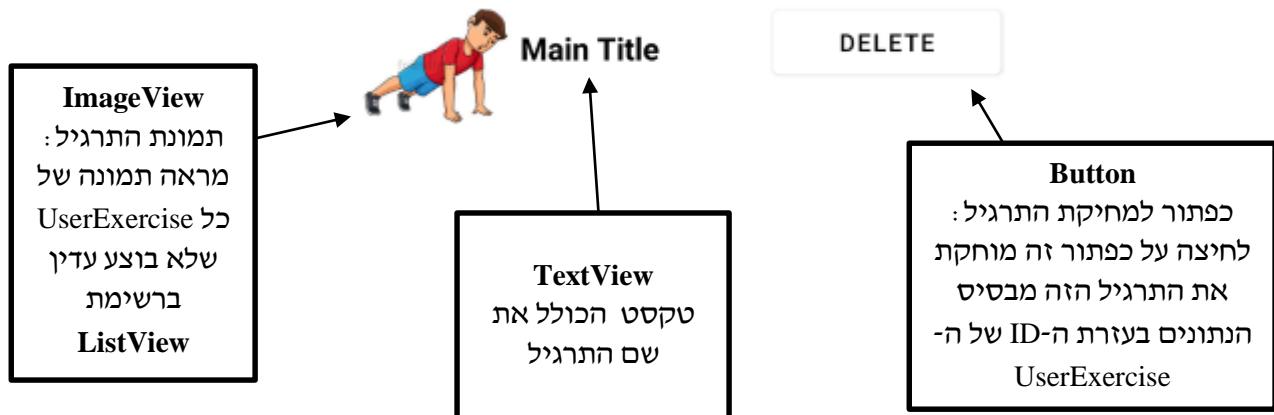
- 4) לעبور למסך הטבלה **StatisticsActivity** בו המשתמש יוכל לראות את תוצאות התרגילים בטבלה אחת עם מספר פילטרים.



- כל איבר ברשימה נבנה לפי העיצוב (layout) הבא :  
לחיצה על איבר ברשימה מעביר את המשתמש למסך **ビ嘱** הניתן להציגו התרגילים הבאים :  
העיצוב כולל את הרכיבים הבאים :

**ImageView, TextView, Button**

: **User\_exercise\_layout**



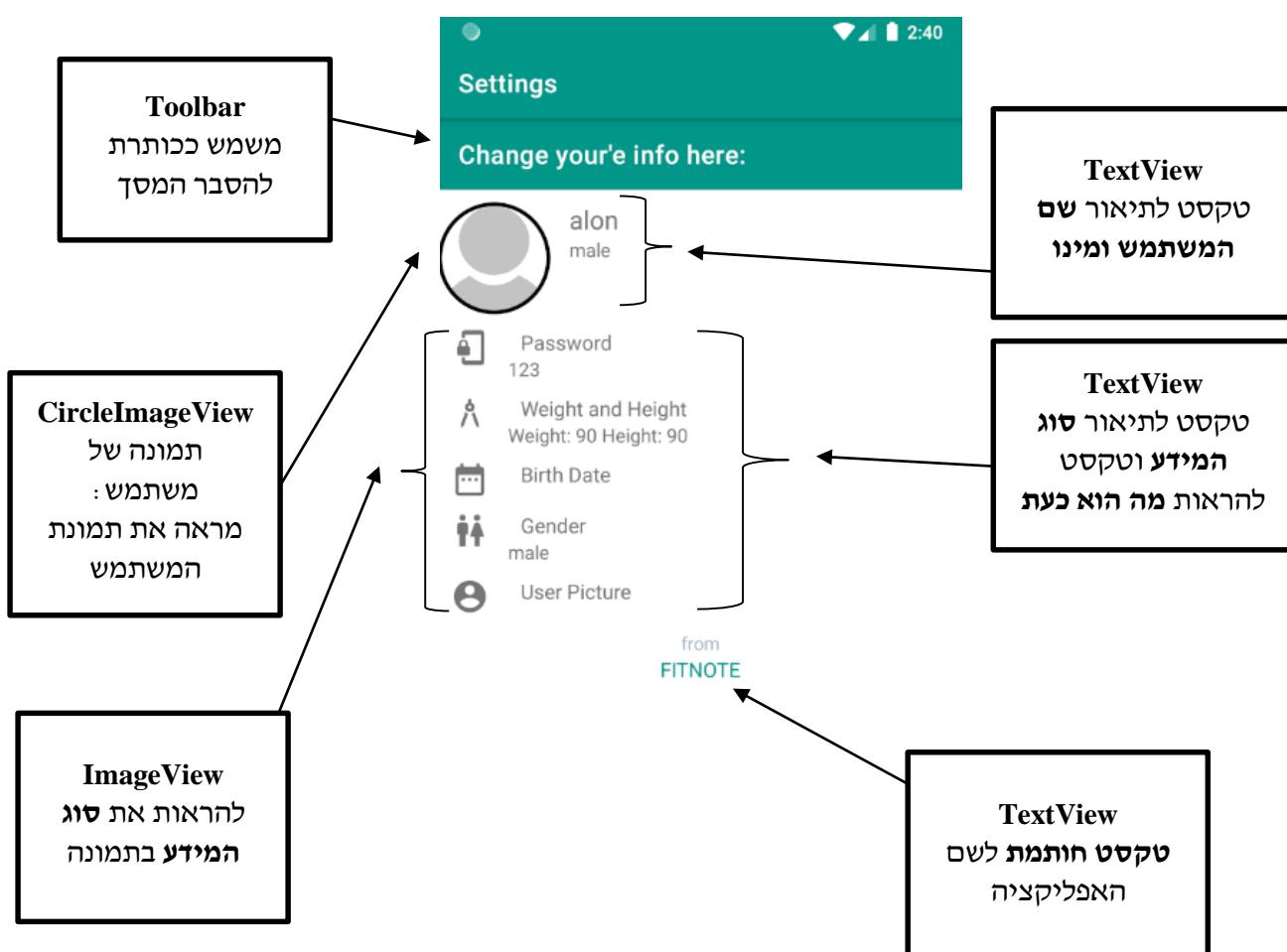
## SettingsActivity

**משמעות הגדירות:** במסך זה המשתמש יכול לשנות את כל פרטי המשתמש שלו (סיסמה, משקל, גובה, תאריך לידיה,מין ותמונה פרופיל) אך לא את שמו מפני שהוא מפתח בסיס הנתונים למציאת המשתמש (בהתחלת הרישום נמנעת אפשרות להירשם עם אותו שם משתמש יותר מפעם אחת!). בלחיצה על כל אחד מה-LinearLayout-ים מופיע AlertDialog מופיע הפרסה (layout) שמאפשר לשנות פרטי המשתמש.

ספרייה של CircleImageView : CircleImageView  
כולל את הרכיבים הבאים :

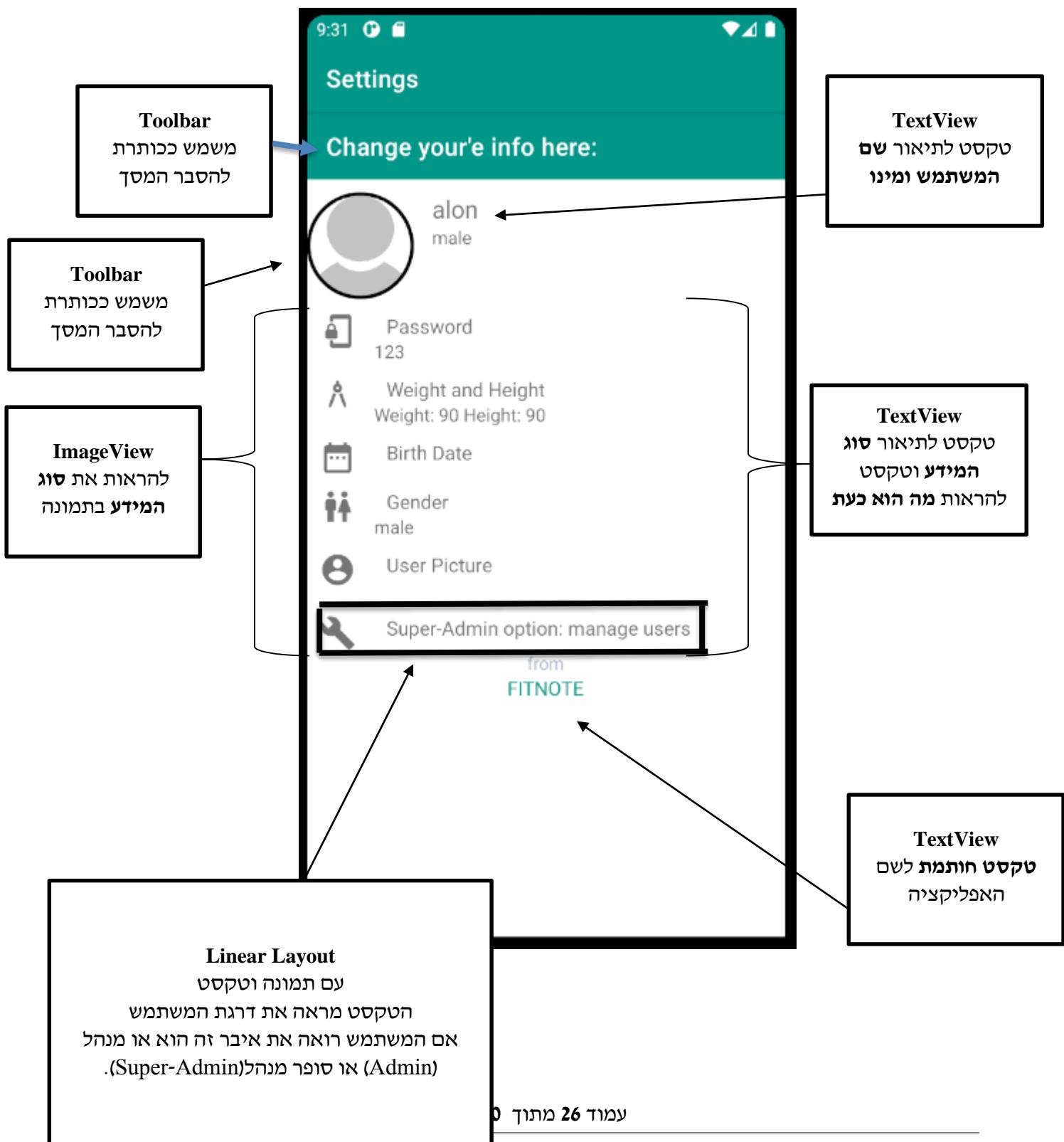
Toolbar, ScrollView, CircleImageView (from a library on GitHub), TextView, View, ImageView

צילומים של מסך ההגדירות אם המשתמש נכנס הוא רגיל (לא מנהל ולא סופר מנהל) :



ה-**LinearLayout** למקרה יופיע אם וرك אם נכנס משתמש שהוא או מנהל (**Admin**) או סופר מנהל (**Super-Admin**). לחייב על-ה-**LinearLayout** עם המידע על דרגת המשתמש וההתמונת של המפתח ברגים לocket את המשתמש למסך **ManagerUserActivity**.

צילום של מסך ההגדירות אם נכנס אליו מנהל או סופר מנהל :



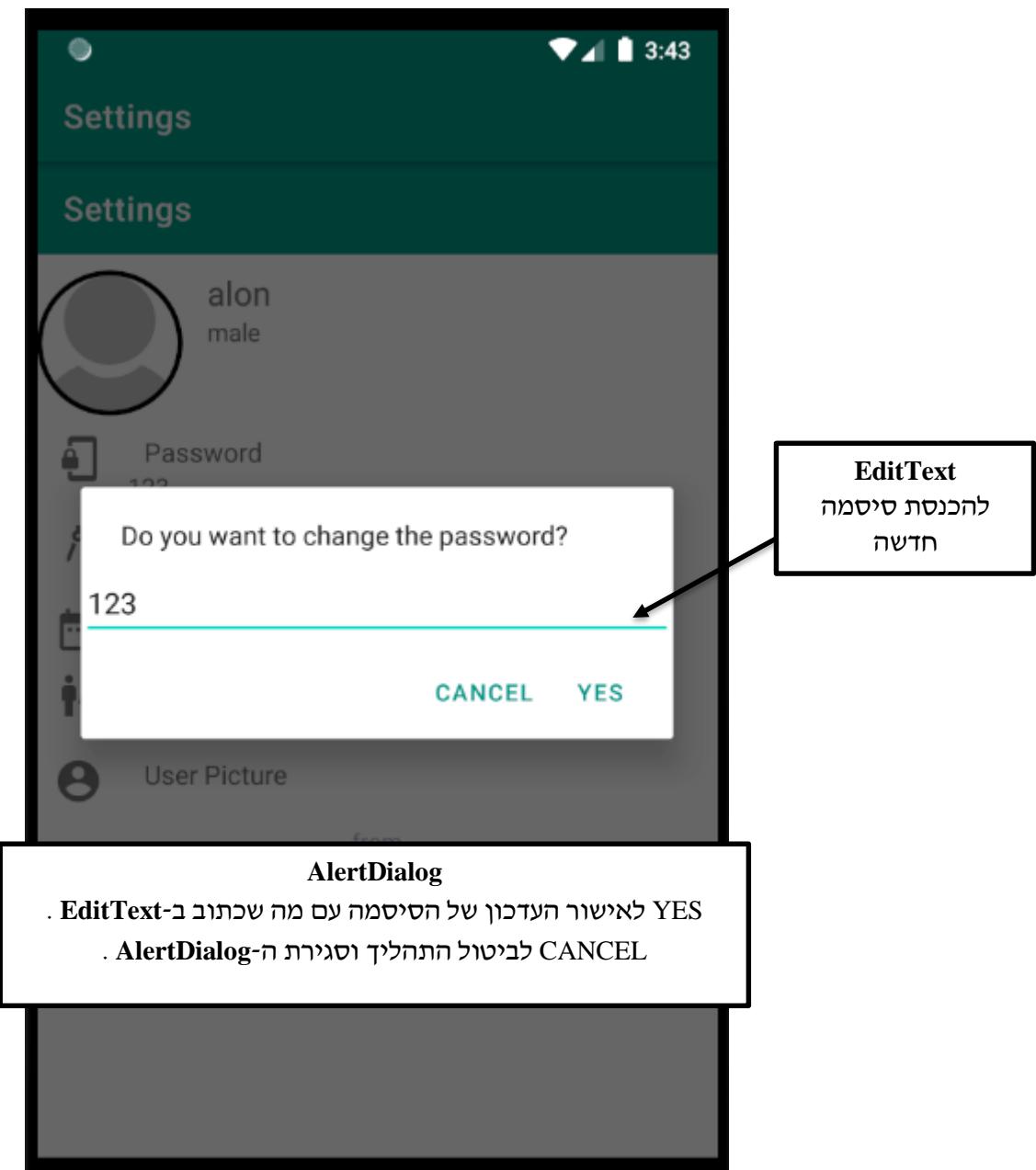
על לחיצת כל שורה בהגדרות מופיעה **AlertDialog** שאיתו אפשר לשנות פרטי מסויים בהתאם של המשתמש.  
כל פרטי המשתמש ניתנים לשינוי חוץ משמו כי זהו מפתח בסיס הנתונים:

שינוי סיסמה:

בחירת הנתונים ב-**AlertDialog** מעכנת את פרטי המשתמש עם הנתון החדש. במקרה זה הסיסמה מתחדשת.

כולל את הרכיבים הבאים:

**EditText**  
צילום של ה-**AlertDialog**

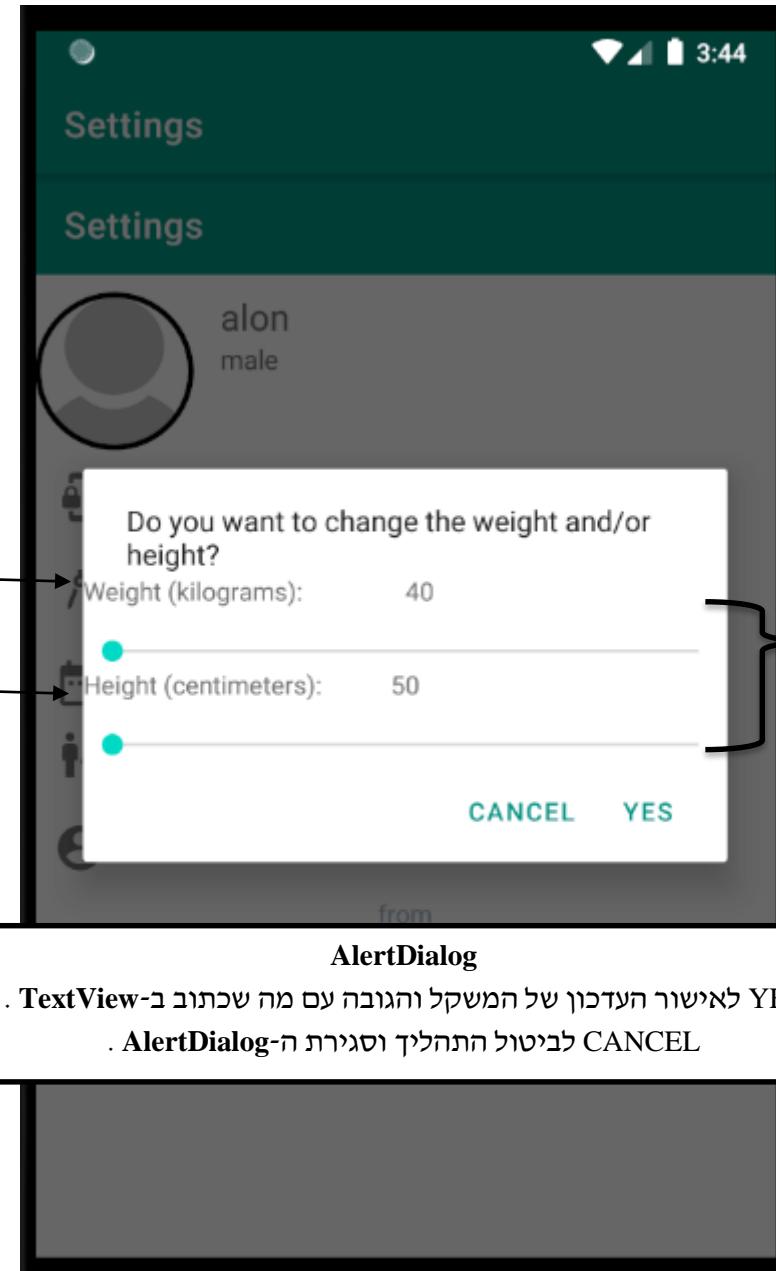


שינוי משקל וגובה :

בחירת הנתונים ב-**AlertDialog** מעדכנת את פרטי המשתמש עם הנתון החדש. במקרה זה המשקל והגובה מटעכנים.

כולל את הרכיבים הבאים :

**TextView, SeekBar**  
צילום של ה-**AlertDialog** :



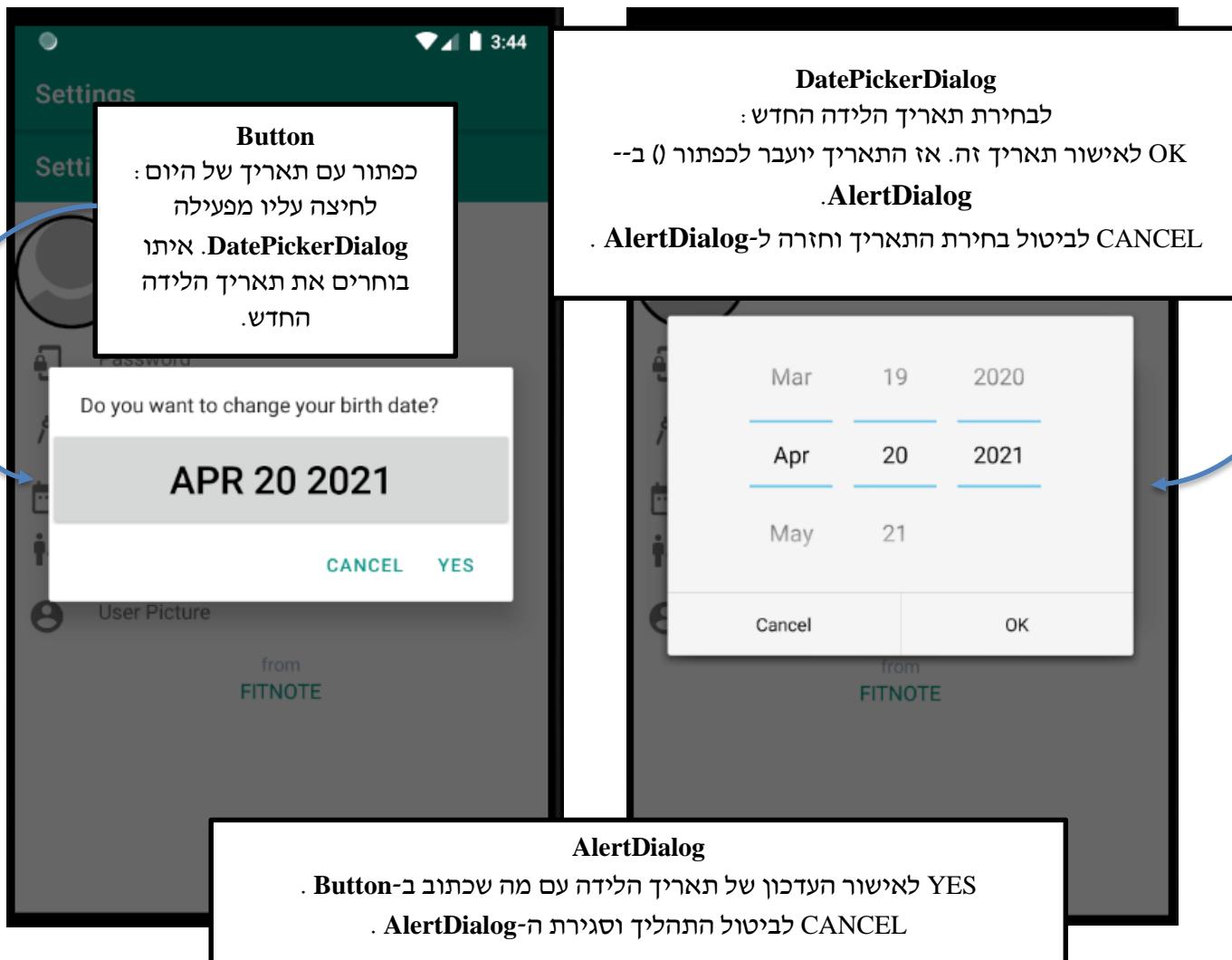
שינויי תאריך לידה:

בחירת הנתונים ב-**AlertDialog** מעדכנת את פרטי המשתמש עם הנתון החדש. במקרה זה תאריך הלידה מותעדן.

כולל את הרכיבים הבאים :

**Button**

צילום של ה-**DatePickerDialog** וה-**AlertDialog**:



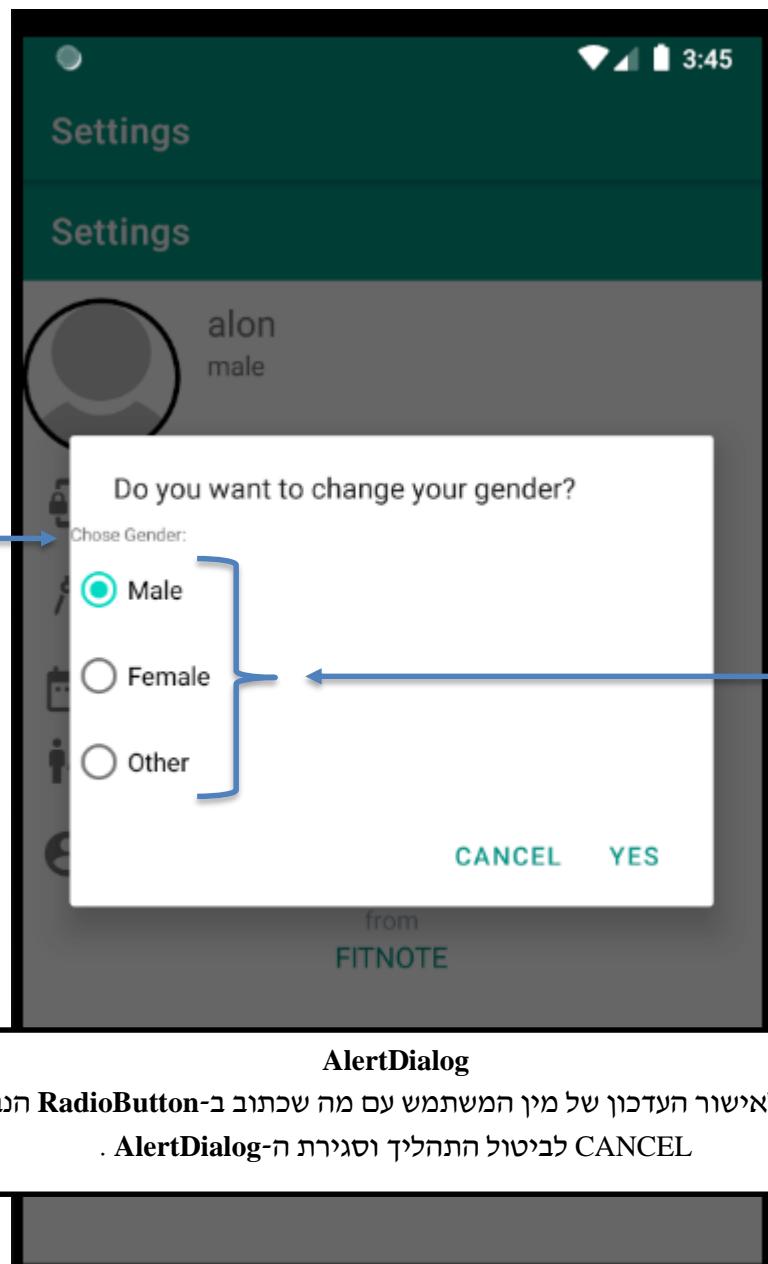
שינויי מין:

בחירת הנתונים ב-**AlertDialog** מעדכנת את פרטי המשתמש עם הנתון החדש. במקרה זה מיון המשתמש מתעדכן.

כולל את הרכיבים הבאים:

**TextView, RadioGroup, RadioButton**

: **AlertDialog** צילום של ה-



**TextView**  
כותרת להסביר

**RadioGroup**  
שמאגד כמה  
**RadioButton**  
אפשרויות אלו  
קובלים את מין  
המשתמש

**AlertDialog**  
YES לאישור העדכון של מין המשתמש עם מה שכתוב ב-**RadioButton** הנבחר.  
. AlertDialog CANCEL  
לביטול התהליך וסגירתה

שינויי תמונה משתמש:

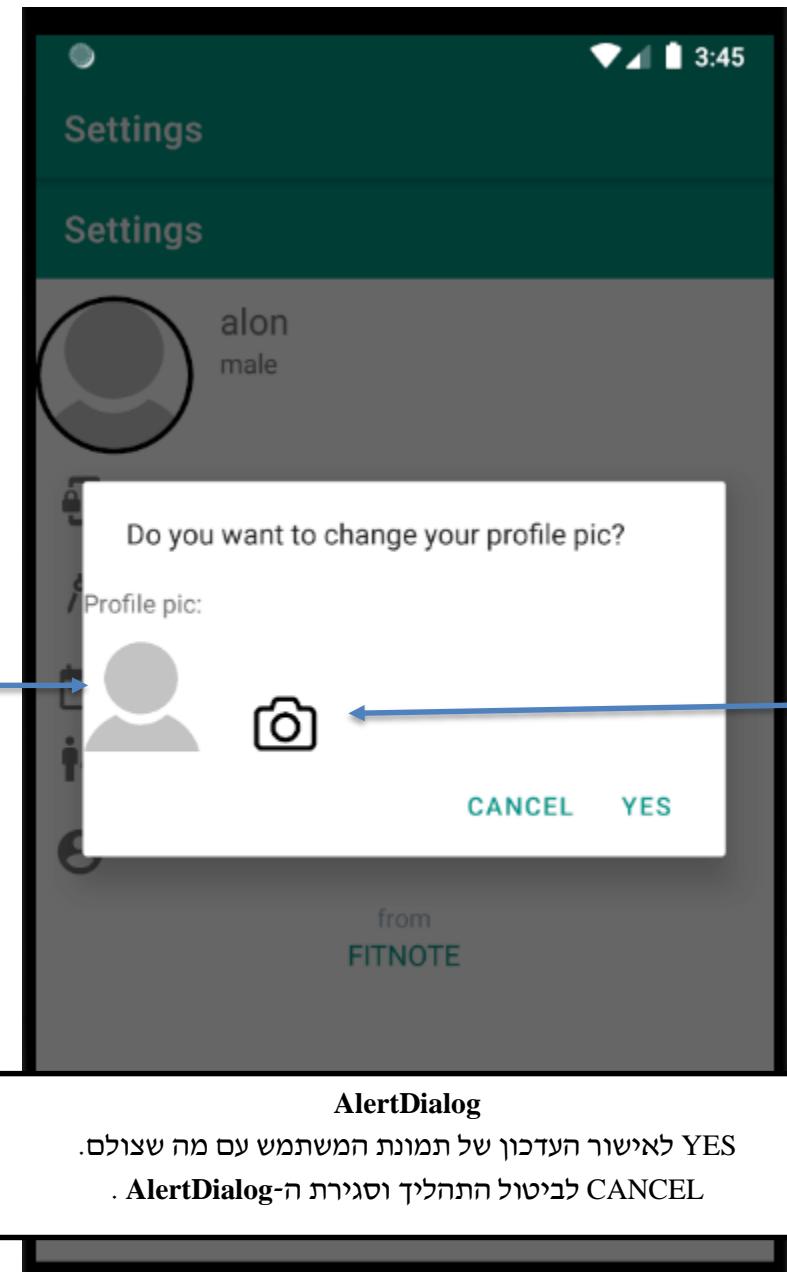
צילום של ה-AlertDialog :

בבחירה הנتونים ב-**AlertDialog** מעדכנת את פרטי המשתמש עם הנתון החדש. במקרה זה תמונה המשתמש מוחדרנת.

כולל את הרכיבים הבאים :

**TextView, ImageView**

צילום של ה-AlertDialog :



#### ImageView

תמונה של  
משתמש :  
מראה את תמונה  
המשתמש לאחר  
קליטת התמונה

#### ImageView

תמונה של מצלמה :  
לחיצה עליה  
פותחת פעולה של  
ליקחת תמונה עם  
המצלמה בטלפון.  
אם אין אישור  
לגשת למצלמה לא  
תיהיה אפשרות  
לצלם בטלפון,  
ובהכנות פרט  
המשתמש יוכנס  
תמונה הפרופיל  
הברירת מחדל. אם  
המשתמש כן צילם  
את עצמו תוכנן  
לבסיס הנתונים  
התמונה שהוא  
צילם

#### AlertDialog

YES לאישור העדכון של תמונה המשתמש עם מה שצולם.

. AlertDialog לביטול התהליך וסגירת ה-AlertDialog . CANCEL

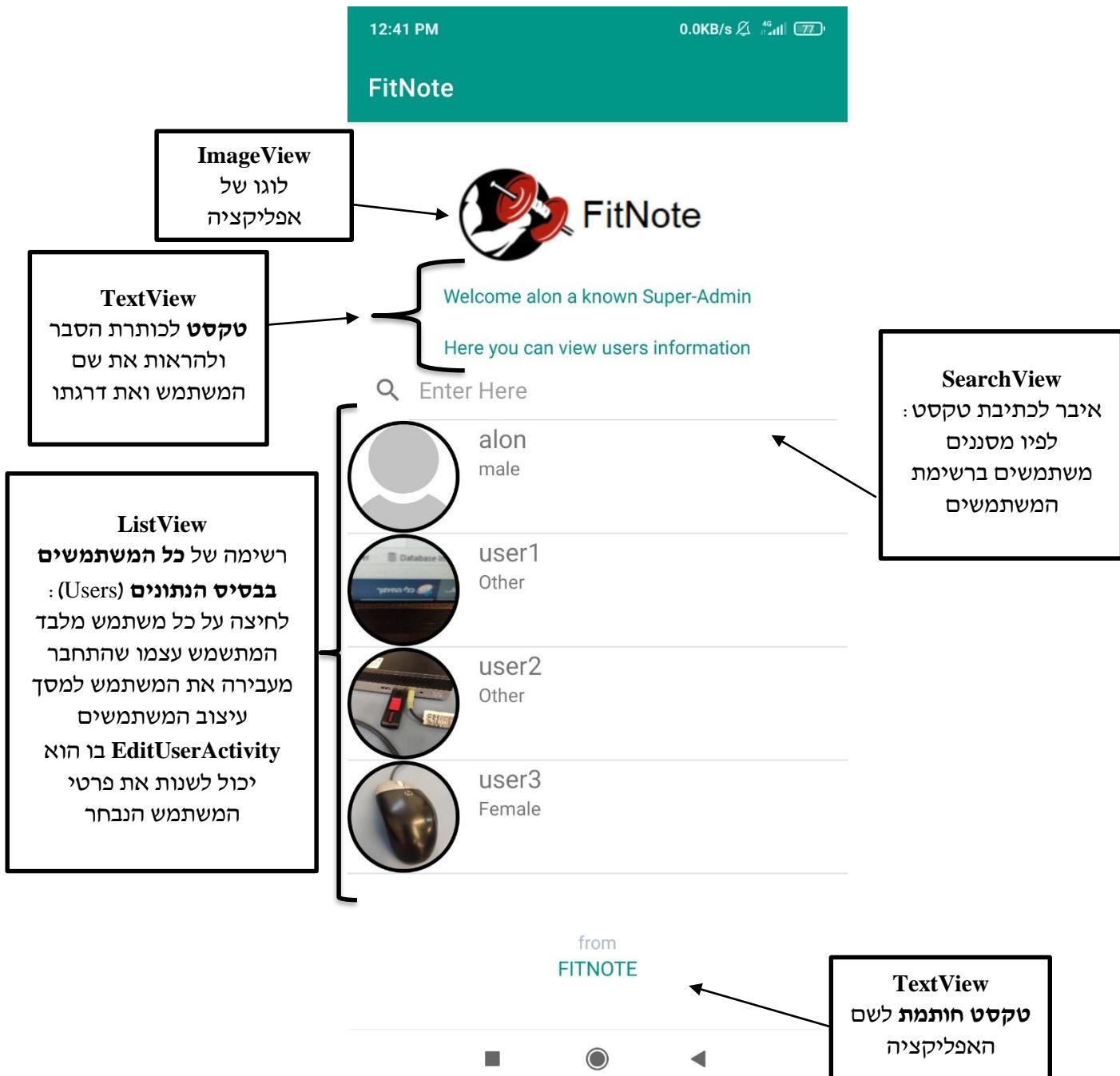
## ManagerUsersActivity

**מץ ניהול המשתמשים :** למס' זה, לפי הגבלה במס' הגדרות, יכול/li יכול להיכנס רק משתמש שaino רגיל. מס' זה מציג את שם המשתמש שנכנס ודרגתנו. אם המשתמש הוא מנהל רגיל (Admin) אז הוא יכול לראות את המשתמשים ברשימה בלבד. אם המשתמש הוא סופר מנהל (Super-Admin) הוא יכול ללחוץ על אחד המשתמשים ברשימה (חוץ מעצמו) ולהגיע למס' (EditUserActivity) בו הוא יכול לשנות את פרטי המשתמש ( בלבד שם ותמונה) ואך למחוק את המשתמש הנבחר.

כולל את הרכיבים הבאים :

TextView, ImageView, SearchView, ListView

צילום של מס' ניהול המשתמשים :



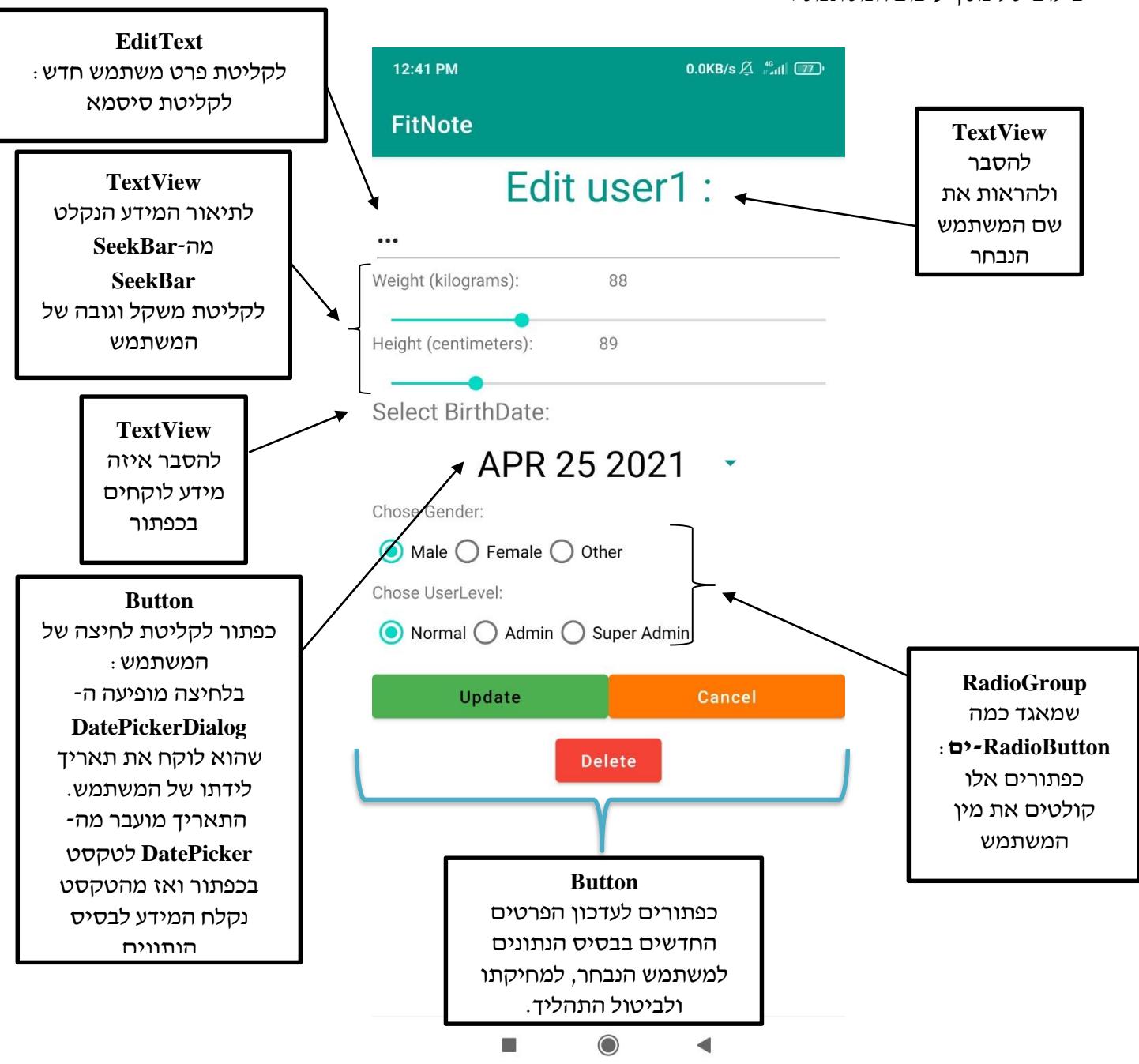
## EditUserActivity

**מסך עיצוב המשתמש :** במסך זה המשתמש שנכנס יוכל לשנות את כל פרטי המשתמש מלבד שמו ותמונה הפרופיל שלו. בנוסף, המשתמש יוכל למחוק את זה שהוא בחר. מחלוקת, עדכון או ביטול בלחיצה על הcptוראים CANCEL ו-UPDATE DELETE בהתאם מוחזירה את המשתמש למסך ניהול המשתמשים.

כולל את הרכיבים הבאים :

**TextView, EditText, SeekBar, Button, RadioGroup, RadioButton**

צילום של מסך עיצוב המשתמש :



## ShareActivity

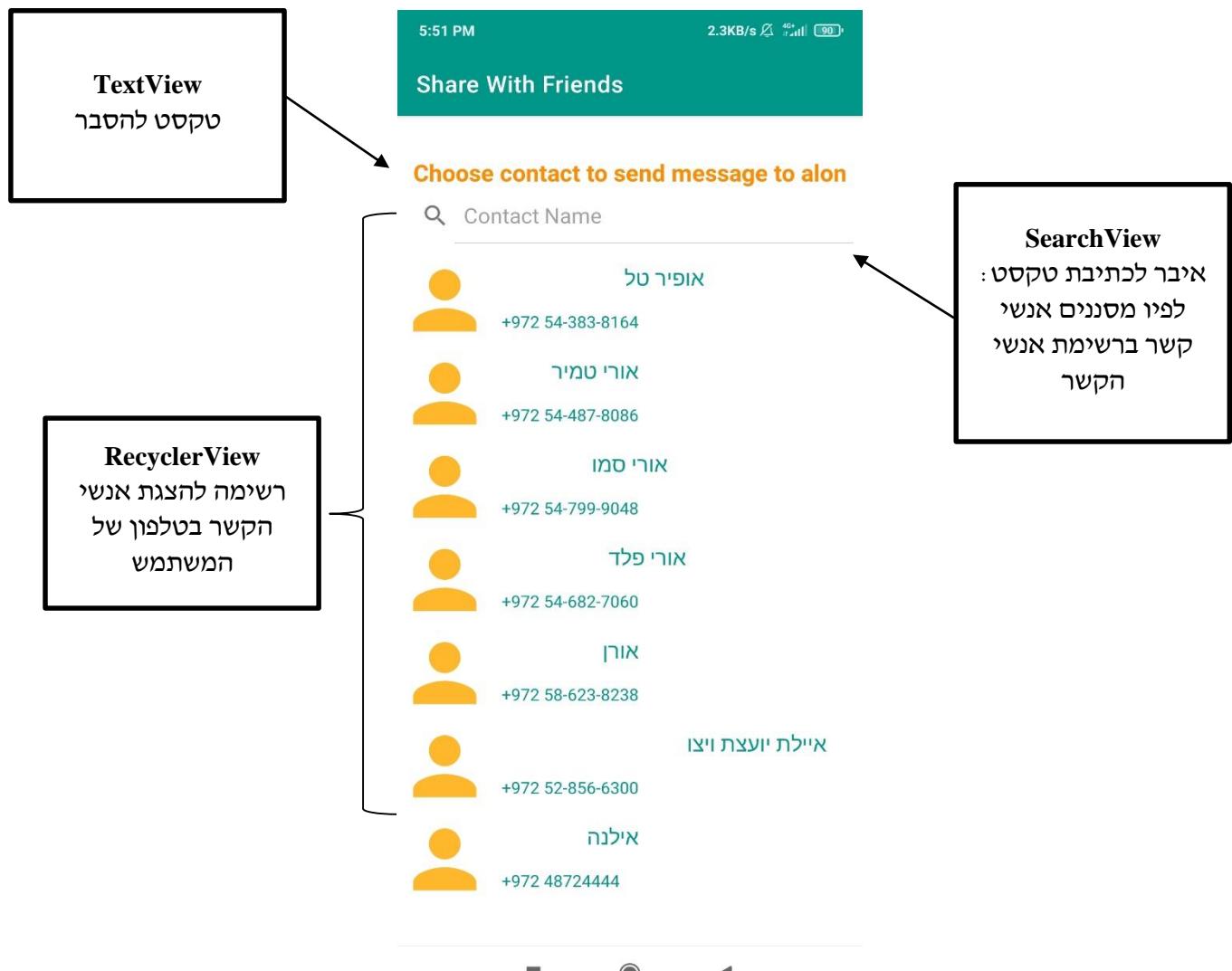
**מיסך שיתוף המידע :** במסמך זה המשתמש יכול, במידה וアイשר לאפליקציה לגשת לאנשי הקשר שלו, לבחור איש קשר אליו הוא רוצה לשלוח הודעה עם **סיכום על כל תוכאות תרגilioו בשבוע האחרון** ועם הודעה נוספת שהוא כותב בהתחלה.

לחיצה על איש קשר ברשימה למיטה מעבירה את המשתמש למסך WhatsAppSendActivity בו שולחים את הודעה למספר הטלפון הנבחר.

כולל את הרכיבים הבאים :

**SearchView, TextView, RecyclerView**

צילום של מסך שיתוף המידע :

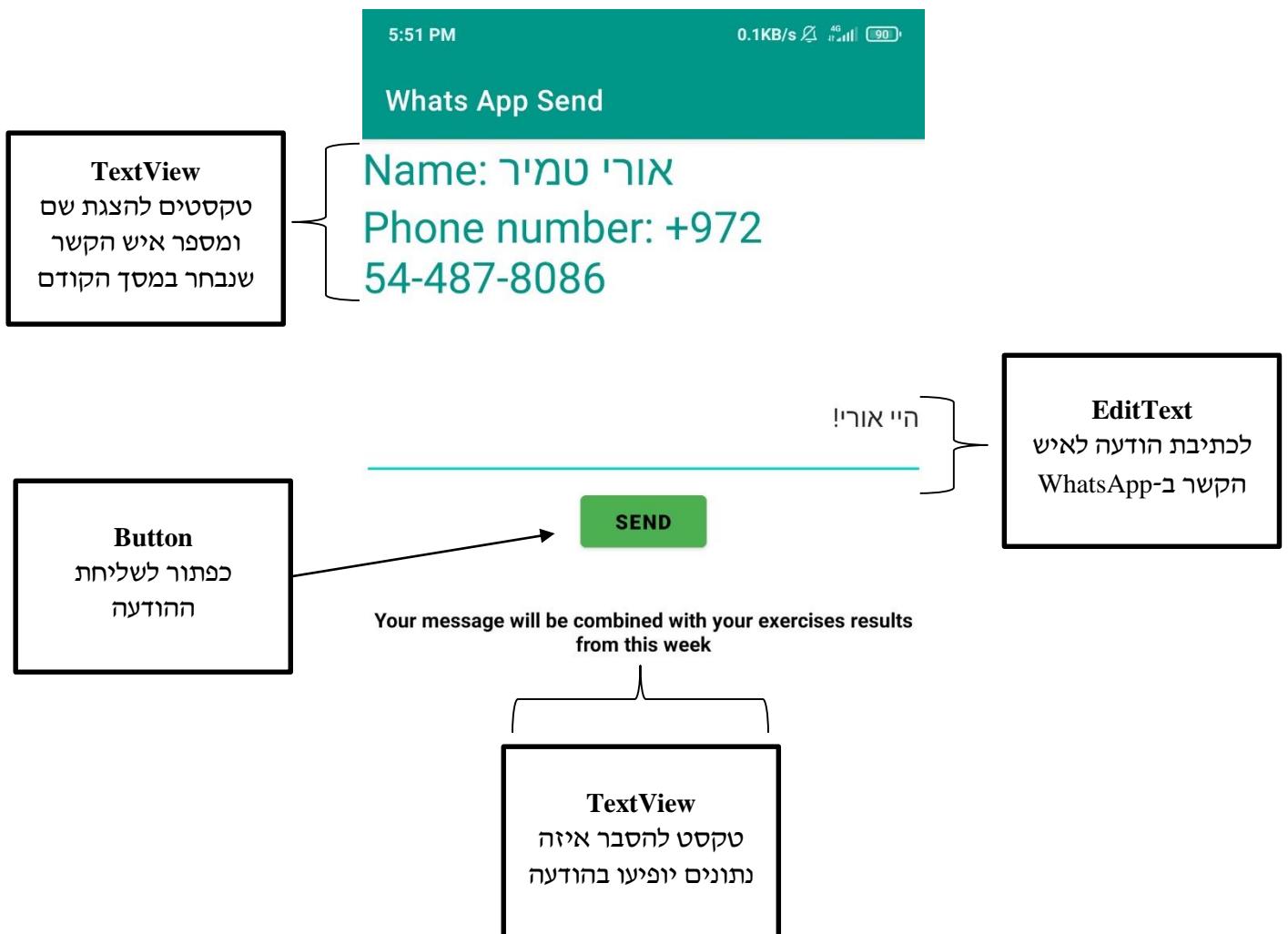


## WhatsAppSendActivity

מוך שליחת הודעה לאיש קשר באוטצאפ : בMarcus זה המשתמש כותב הודעה הנשלחת לאיש הקשר אותו בחר Marcus בMarcus ShareActivity. להודעה הנשלחת מתווסף טקסט אותו יכול לכתוב המשתמש ב-EditText. תהיה אפשרות לשלוח הודעה באוטצאפ רק עם האפליקציה WhatsApp מותקנת בטלפון. כולל את הרכיבים הבאים :

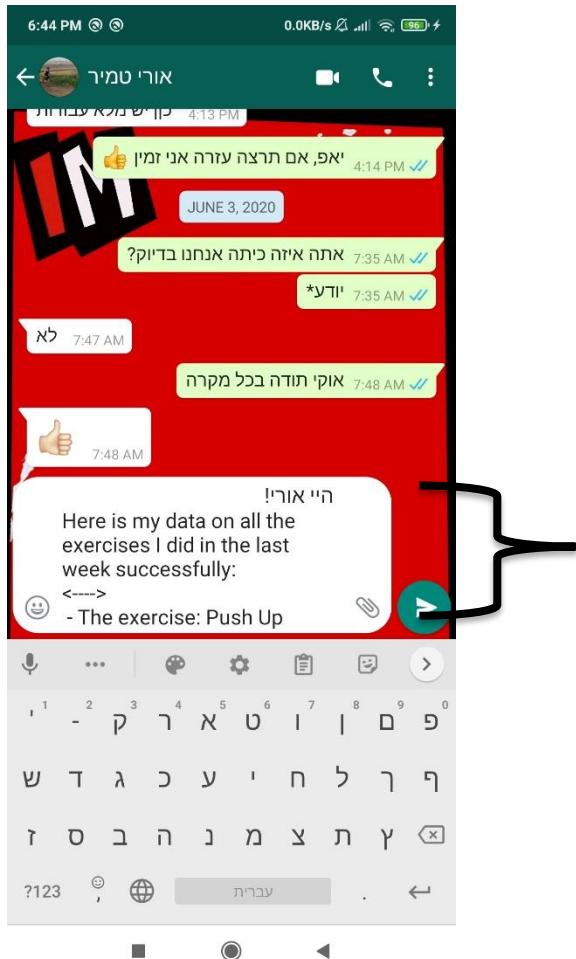
**TextView, EditText, Button**

צילומים של מוך שליחת הודעה לאיש קשר באוטצאפ :



## דוגמא שליחת הودעה ב-App

צילום מסך של שליחת הודעה באוטומטוף הכלולות הודעה אישית של המשתמש ונתונים על תוצאות התרגילים מהאפליקציה:



ה"י אורי!

Here is my data on all the exercises I did in the last week successfully:

<---->

- The exercise: Push Up
- The date in which the exercise was done: APR 21 2021
- The time it took to perform the exercise: 00 : 00 : 10
- The rating alon gave to the exercise: Easy-Medium
- alon did the exercise 47times

<---->

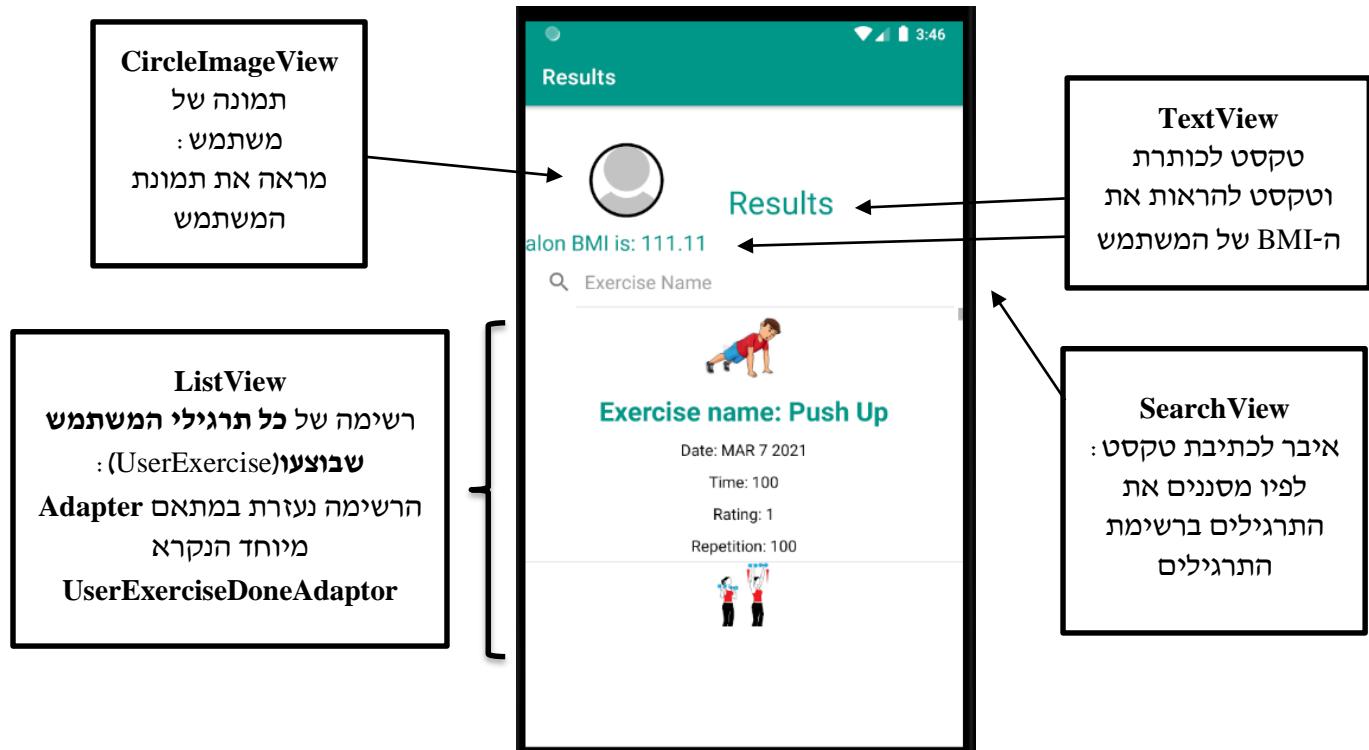
- The exercise: HammerCurls
- The date in which the exercise was done: APR 21 2021
- The time it took to perform the exercise: 00 : 00 : 13
- The rating alon gave to the exercise: Medium
- alon did the exercise 30times

## ViewExercisesResultsActivity

**מוך נתונים על התרגילים שבוצעו ברשימה :** במסמך זה מופיעים כל התרגילים שביצע המשתמש והנתונים עליהם ברשימה. מעלה מופיעה ה-BMI של המשתמש שמחשב לפי הגובה של המשתמש ומשקלו(BMI זה משקל חלקי הגובה במטרים בריובע).

<https://github.com/hdodenhof/CircleImageView> : CircleImageView ספרייה של מטרים ברשימה :  
כולל את הרכיבים הבאים :

**CircleImageView (from a library on GitHub), TextView, SearchView, ListView**  
צילום של מוך נתונים על התרגילים שבוצעו ברשימה :



## StatisticsActivity

**מיצ' הטבלה :** במאז' זה מופיעות טבלת chart בה יש נתונים על כל התרגילים שביצע המשתמש.

- צבע העמודות מראה את הדירוג של המתאמן על דרגת הקושי של התרגיל.

- מיקום בציר ה-x מצין את החודש שבו בוצע התרגיל.

- גובה ביחס לציר ה-y מצין את מספר החזרות שביצע המתאמן בתרגיל זה.

- \* מעל הטבלה יש פילטרים להראות סוג מסוים של תרגילים ובסנה מסוימת.

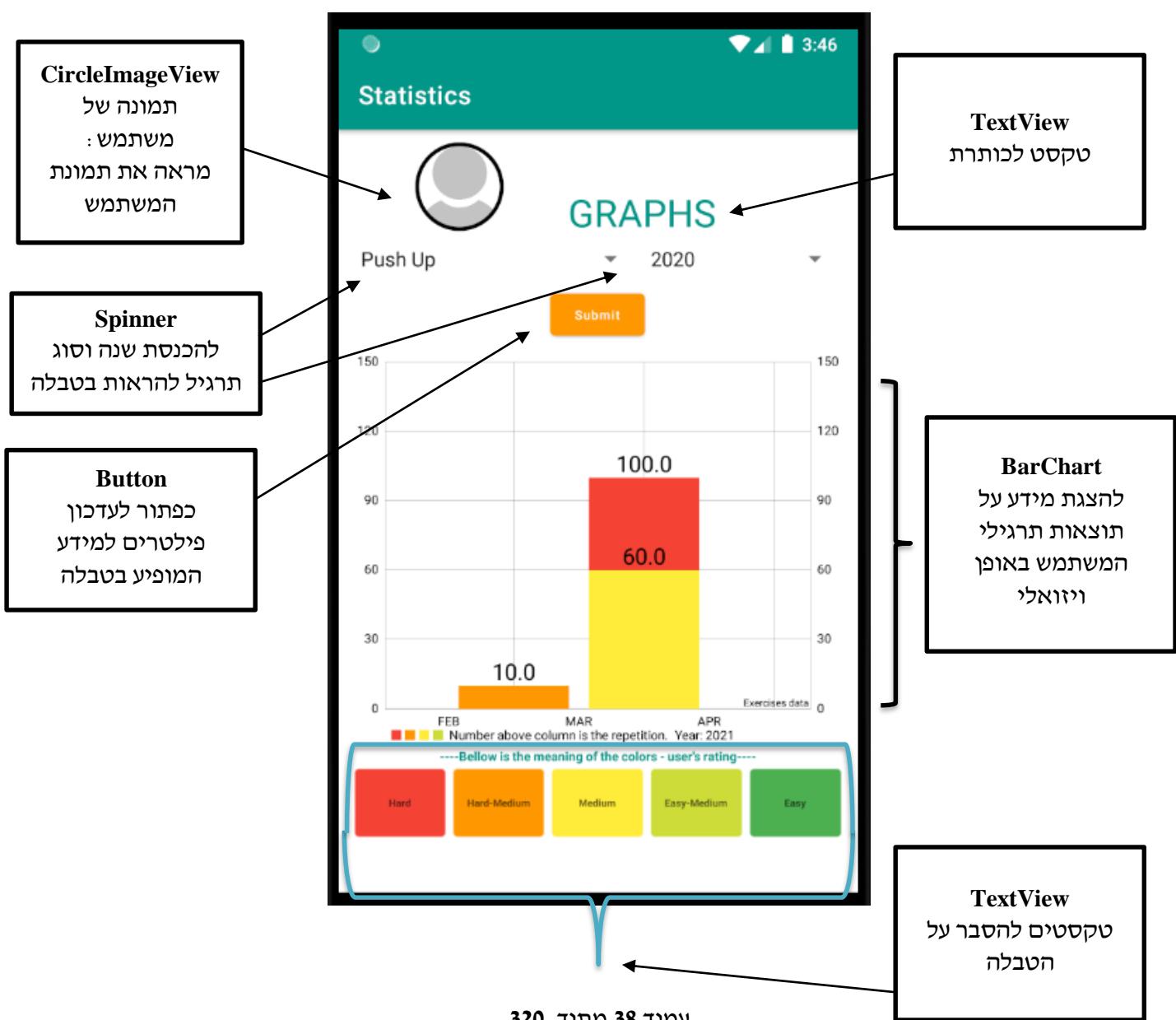
ספרייה של MPAndroidChart : <https://github.com/PhilJay/MPAndroidChart-Realm>

ספרייה של CircleImageView : <https://github.com/hdodenhof/CircleImageView>

כולל את הרכיבים הבאים :

**CircleImageView (from a library on GitHub), Spinner, Button, BarChart (from a library on GitHub), TextView**

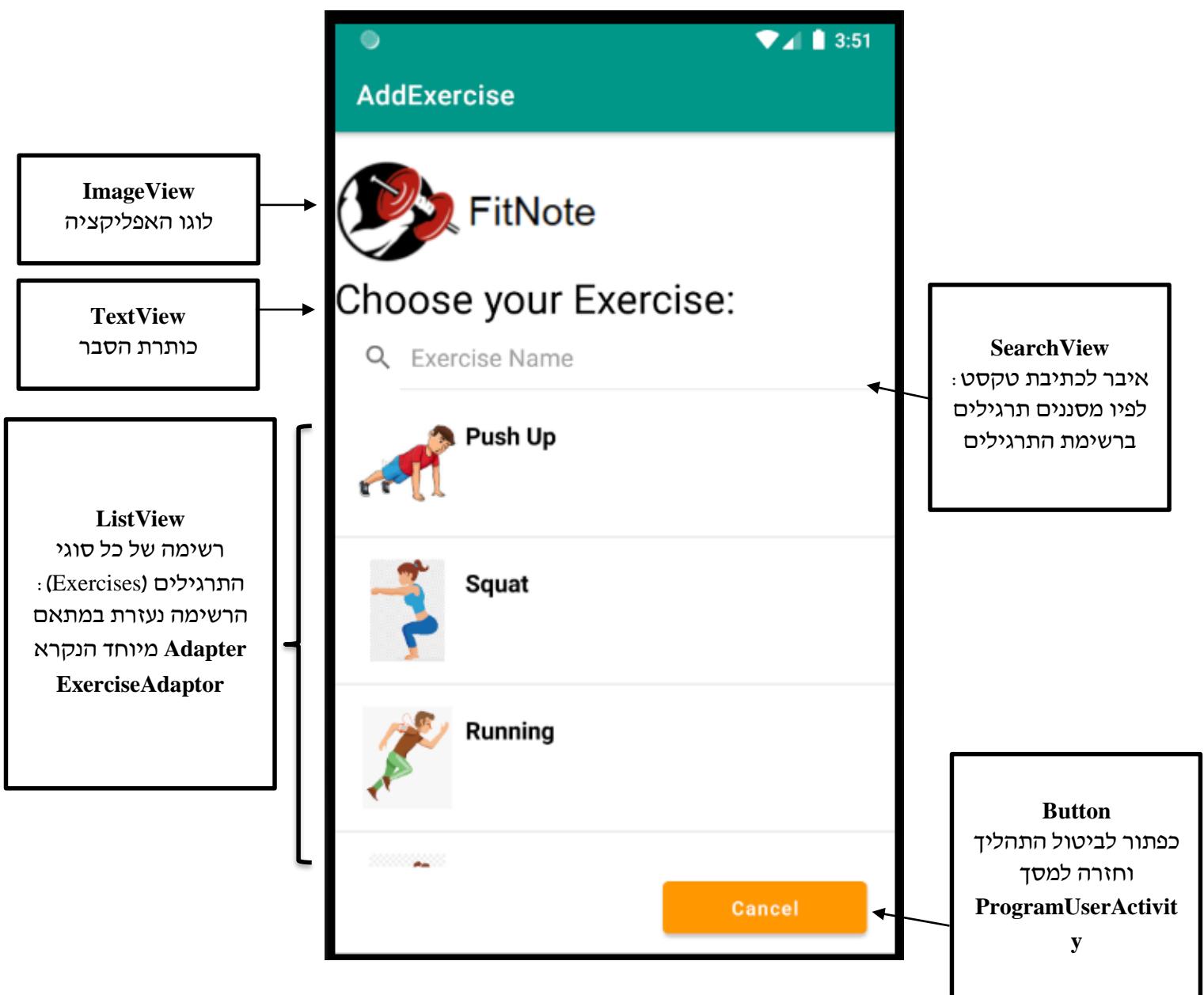
צילומים של מיצ' הטבלה :



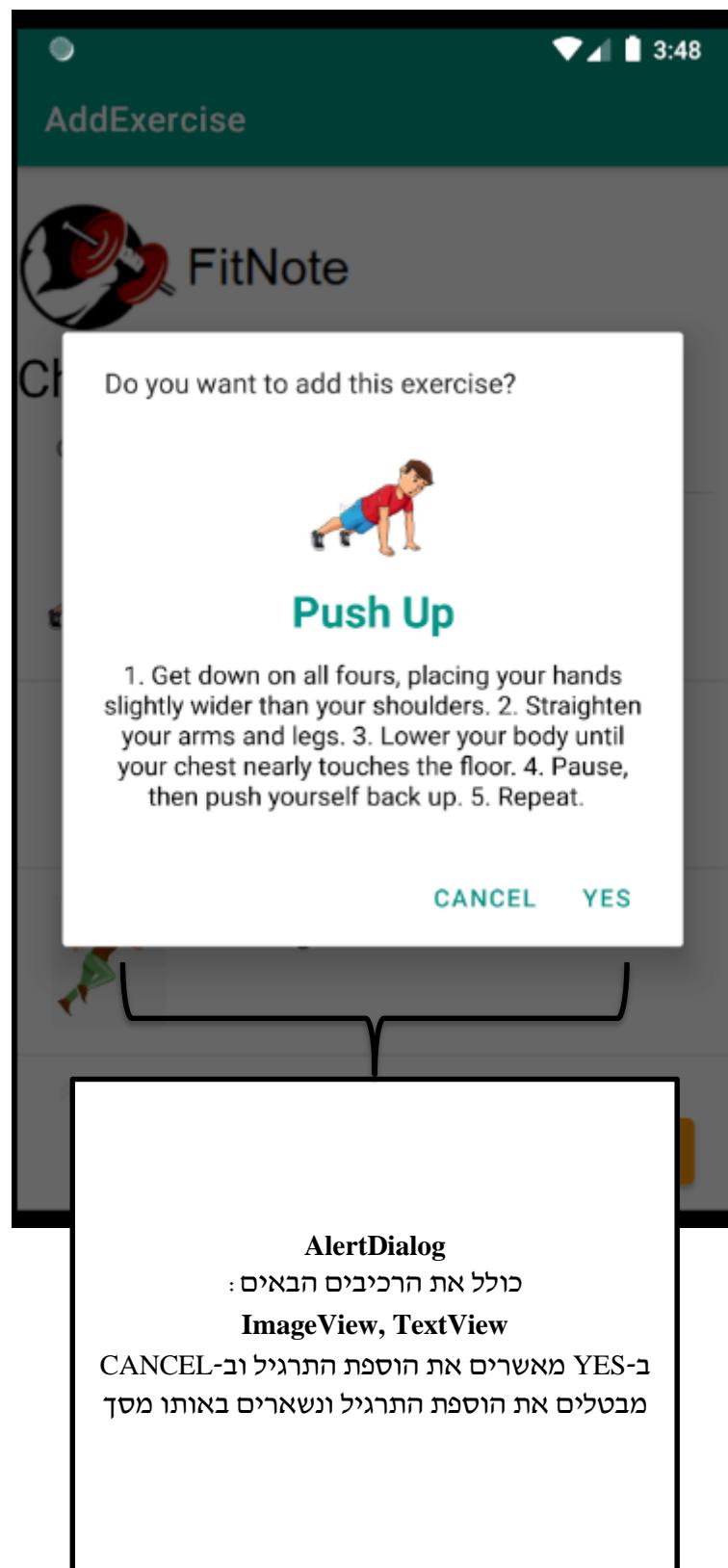
## AddExerciseActivity

**מץ הוספת התרגילים :** במצ' זה המשמש בוחר את התרגיל שהוא רוצה לבצע. לחיצה על התרגיל תפעיל **AlertDialog** בו יופיע הסבר על מה התרגיל שהמשתמש רוצה להוסיף. אישור במצ' הדו שיח יוסיף את התרגיל לרשימת התרגילים שהמשתמש עוד לא ביצע. ביטול במצ' הדו שיח יוריד את ה-**AlertDialog** ו ישאיר את המשמש באותו מסך. כולל את הרכיבים הבאים :

**ListView, TextView, SearchView, Button**  
צילום של מץ הוספת התרגילים :



צילום מסך הוספה התרגילים עם הופעה של ה-dialog :



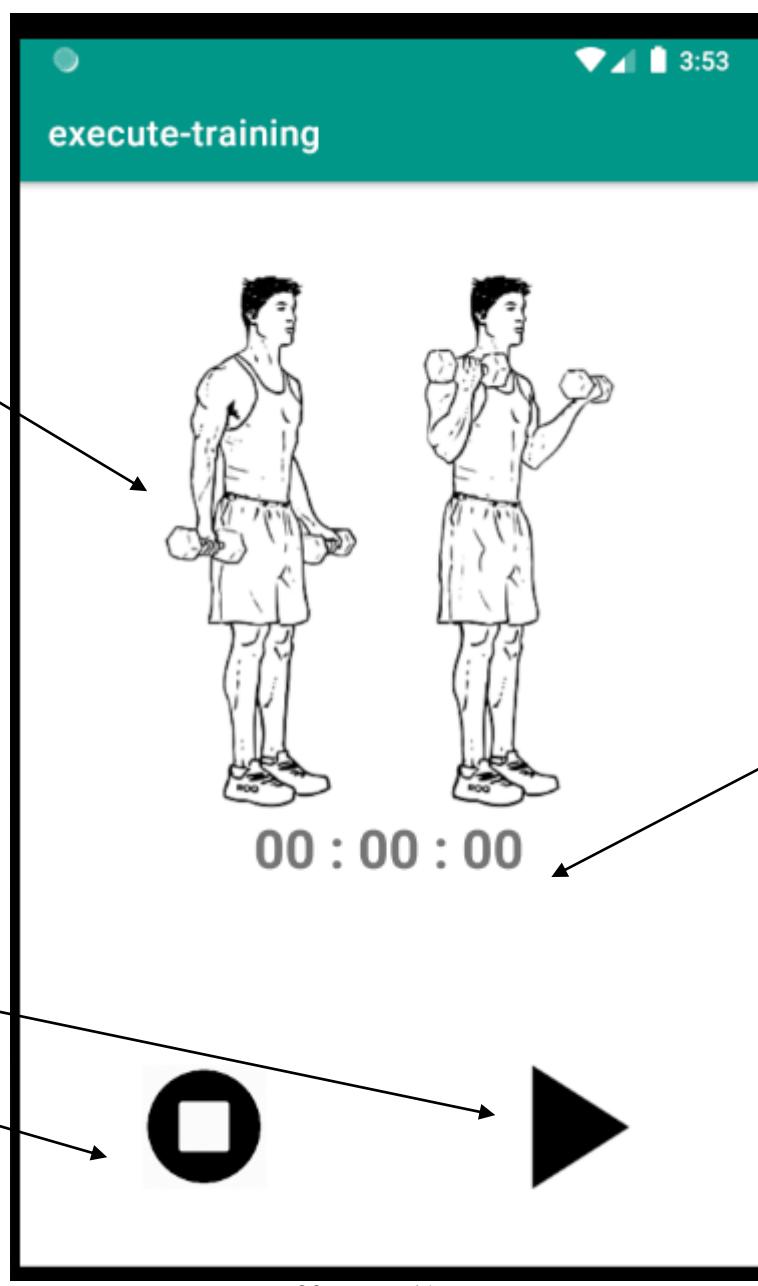
## ExecuteExerciseActivity

**מסך ביצוע התרגילים:** במסך זה המשמש לבצע את התרגיל שהוא בחר מוקדם במסך ProgramUserActivity. לחיצה על התמונה PLAY מימין למיטה מפעיל את שירות (Service) המוזיקה והטיימר MusicAndTimerService. לחיצה על התמונה PAUSE פעם אחת הופכת את המשולש לשני קווים מקבילים שזהו בעצם כפתור ה-PAUSE משacha את השירות והופכת את המשולש למשולש בחזרה לכפתור ה-PLAY. לחיצה על התמונה STOP משמאלי למיטה עצרת את השירות (Service) לסיום ומעבירה את המשמש במסך FeedbackActivity.

כולל את הרכיבים הבאים:

ImageView, TextView

ציילום של מסך ביצוע התרגילים:



**ImageView**  
להראות את תמונה  
התרגיל שמבצע  
המשתמש

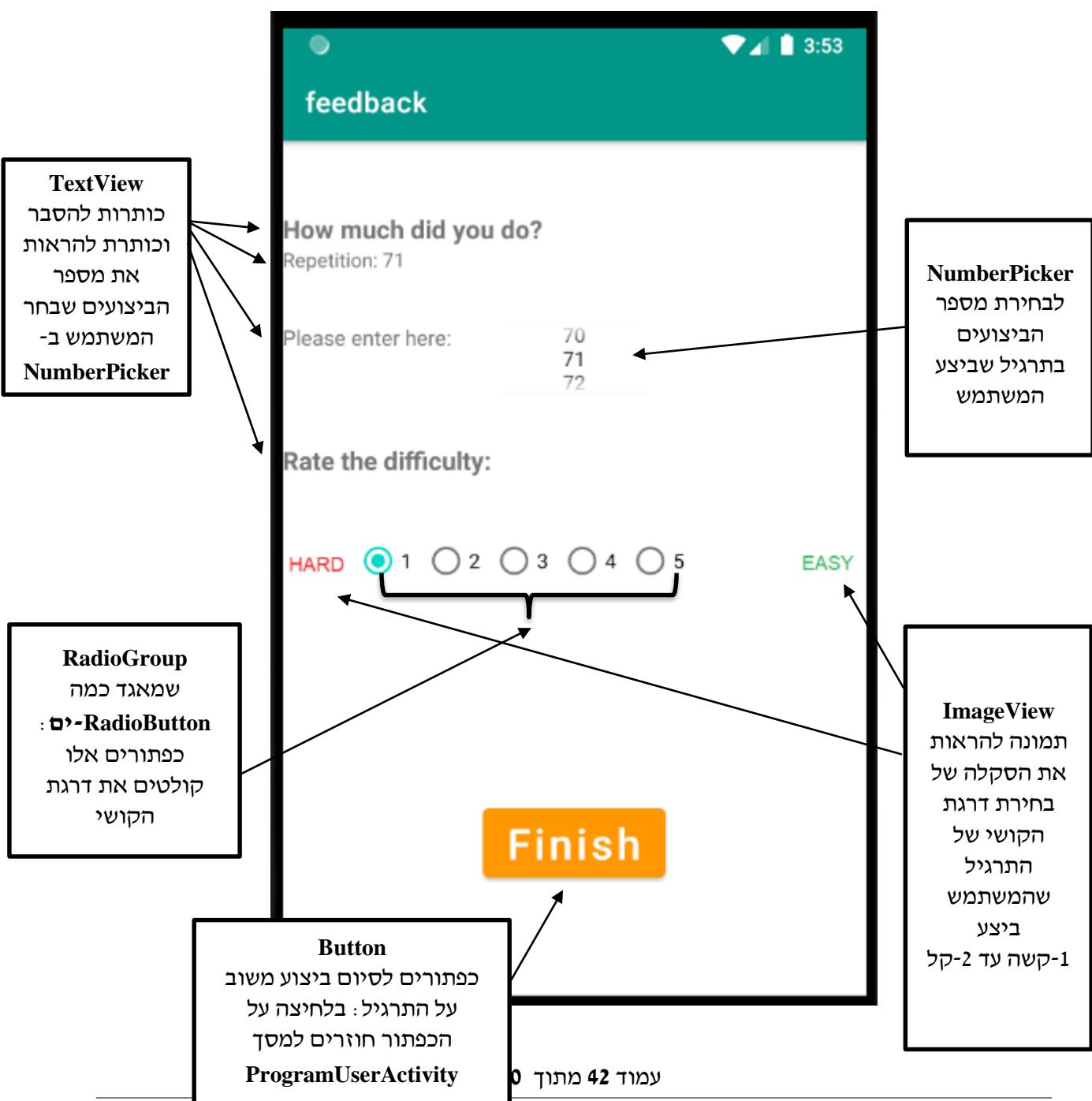
**ImageView**  
כפתוריים להפעלת  
המוזיקה והטיימר,  
השהיינות והפסקתו.  
לחיצה על כפתור  
הפעלה הופכת אותו  
לכפתור השהייה.  
לחיצה על כפתור  
העצירה מעבירה את  
המשתמש במסך  
המושב  
**FeedbackActivity**

**TextView**  
להראות את הזמן  
בטימר המופעל  
בלחיצה על כפתור  
ההפעלה מימין למיטה

## FeedbackActivity

**מץ משוב על ביצוע התרגיל :** במצ' זה המשתמש כותב כמה ביצועים הוא ביצע בתרגיל במסך הקודם ExecuteExerciseActivity, והוא מדרג את רמת הקושי של התרגיל שהוא ביצע. את הזמן שהלך לבצע את התרגיל, את ה-`userExerciseID` של התרגיל מהמצ' הקודם, את מספר הביצועים ואת דירוג הקושי לוחכים ביחס ומעדכנים את תרגיל המשמש בסיס הנתונים עם הנתונים החדשים. **באופן זה התרגיל הופך מתרגיל שהמשתמש לא ביצע לכזה שהוא כן ביצע.** לחיצה על כפתור ה-`FINISH` מסיים את תהליך ביצוע התרגיל והמשתמש חוזר למסך `ProgramUserActivity`. **ככל שמשתמש עם המידע החדש על ביצוע התרגיל והמשתמש חוזר למסך** `feedback`.

**TextView, NumberPicker, ImageView, RadioGroup, RadioButton, Button**  
צילום מסך המשוב על ביצוע התרגילים :



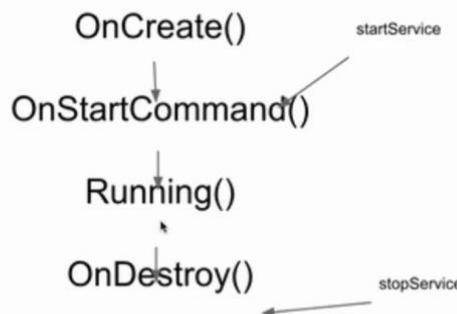
## Services

### הגדרת השירות (Service) :

הוא שירות שנועד לביצוע פעולות ארוכות טווח. טווח של זמן בו יש סיכוי שהמשתמש יעבור לאפליקציה אחרת. כ当下עbor מסך ה-Thread (תהליכון ברקע) יופסק. ככלומר כנסגור את ה-activity ה-thread יופסק. לעומת זאת, ה-Service ימשיך לפעול גם כ当下עbor בין מסכים. נוכל לעצור את ה-Service רק בסיגורה מלאה של האפליקציה או בהפסקת השירות. לשירות יש תוחלת חיים משל עצמו. הוא מוגבל ביכולת שלו לבודד אינטראקטיבית עם רכיבים (components מבחרית UI). רוב השירותים הם `.started`.

שימוש ל-StartService : שמיעת מזיקה ברקע.

### מעגל החיים של service started



### MusicAndTimerService

שירות שפועל מזיקה וטיימר ברקע בעת ביצוע התרגיל במסך ExecuteExerciseActivity. המחלקה יורשת ממחלקה service שהיא מובנת בתוך ה-.android

#### תכונות:

##### (1) שלוש שדות במחלקה :

- `private DataBaseHelper DataBaseHelper;`
- `private MediaPlayer mediaPlayer;`
- `private CountDownTimer countDownTimer;`

##### (2) מימוש פעולה הורשה

- `onCreate` פעולה שנתקראת בעת מימוש השירות שבה משיגים שיר רנדומלי מהשירים הנלקחים מבסיס הנתונים ומממשים את הטיימר ואת נגן המזיקה
- `onStart` פעולה שנתקראת בעת הפעלת השירות. ב-Intent String מועבר String שמסביר אם הפעולה המבוצעת על ידי השירות צריכה להיות הפעלת המזיקה והטיימר (PLAY) או לשחות אותן (PAUSE)
- `onDestroy` פעולה הנתקראת בעת עצירת השירות והשמדתו. הפעולה מבטלת את הטיימר ועוצרת את המזיקה

(3) הרשמת השירות ב-Manifest

&lt;service android:name=".MusicAndTimerService" /&gt;

(4) הפעלת השירות על ידי ה-Intent

לנגינת השיר :

```
Intent startService = new Intent(this, MusicAndTimerService.class);
startService.putExtra("ACTION", "PLAY");
startService(startService);  
לחשיית הנגינה של השיר :
```

```
Intent startService = new Intent(this, MusicAndTimerService.class);
startService.putExtra("ACTION", "PAUSE");
startService(startService);
```

(5) עצירת השירות למימי :

```
//stop button to stop music service
Intent startService = new Intent(this, MusicAndTimerService.class);
stopService(startService);
```

## Content Providers

הגדרת ה-ContentProvider :

הוא רכיב המשמש להעברת מידע בין אפליקציה אחת לשניה. נשתמש ב-ContentProvider שהוא שפה אפליקציה שלנו קיבל מידע מאפליקציה אחרת. לדוגמה באפליקציה זו שולפי את אנשי הקשר שבטלפון וכן הצגת את מספרי הטלפון ושמות אנשי הקשר על מנת לשלוח לאיש קשר הנבחר הודעה הכוללת מידע על תוכנות התרגילים של המשתמש.

תכונות :

(1) בקשה הרשאה לגשת נתונים אנשי הקשר בטלפון :

```
private void checkPermission() {
    //Check condition
    if (ContextCompat.checkSelfPermission(ShareActivity.this
            , Manifest.permission.READ_CONTACTS)
        != PackageManager.PERMISSION_GRANTED){
        //When permission is not granted
        //Request permission
        ActivityCompat.requestPermissions(ShareActivity.this
                , new String[]{Manifest.permission.READ_CONTACTS}, 100);
    }else {
        //When permission is granted
        //Create method
        getContactList();
    }
}
```

(2) שיליפת נתונים לאובייקט (עצם) שנקרא Cursor :

```
private void getContactList() {
    //Initialize uri
    //uniform resource identifier
    Uri uri = ContactsContract.Contacts.CONTENT_URI;

    //Sort by ascending (ASC)
    String sort = ContactsContract.CommonDataKinds.Phone.DISPLAY_NAME+" ASC";

    //Initialize cursor
    Cursor cursor = getContentResolver().query(
            uri, null, null, null, sort
    );

    //Check condition
    if(cursor.getCount() > 0){
        //When count is greater than 0
        //Use while Loop
        while (cursor.moveToNext()){
            //Cursor move to next
            //Get contact id
            String id = cursor.getString(cursor.getColumnIndex(
                    ContactsContract.Contacts._ID
            ));
            //Get contact name
            String name = cursor.getString(cursor.getColumnIndex(
                    ContactsContract.Contacts.DISPLAY_NAME
            ));
            //Initialize phone uri
            Uri uriPhone = ContactsContract.CommonDataKinds.Phone.CONTENT_URI;
```

```

        //Initialize selection
        String selection =
ContactsContract.CommonDataKinds.Phone.CONTACT_ID
        +" =?";
        //Initialize phone cursor
        Cursor phoneCursor = getContentResolver().query(
            uriPhone, null, selection
            , new String[]{id}, null
        );
        //Check condition
        if (phoneCursor.moveToNext()) {
            //When phone cursor move to next
            String number =
phoneCursor.getString(phoneCursor.getColumnIndex(
                ContactsContract.CommonDataKinds.Phone.NUMBER
));
            //Initialize contact model
            ContactModel model = new ContactModel();
            //Set name
            model.setName(name);
            //Set number
            if(number.charAt(0) == '0')
            {
                number = "+972 "+number.substring(1);
            }
            model.setNumber(number);
            //Add model in array list
            contactList.add(model);
            //Close number cursor
            phoneCursor.close();
        }
    }
    //Close cursor
    cursor.close();
}

//setting filtered list to be contactList
filteredContacts = contactList;

//Set Layout manager
recyclerViewContacts.setLayoutManager(new LinearLayoutManager(this));

//setOnItemClickListener
setOnClickListener();

//Initialize adapter
adapter = new ContactListAdapter(this, contactList, listener);
//Set adapter
recyclerViewContacts.setAdapter(adapter);
}

//setting filtered list to be contactList
filteredContacts = contactList;

//Set Layout manager
recyclerViewContacts.setLayoutManager(new LinearLayoutManager(this));

```

הציג הנתונים ב- RecyclerView (3)

```
//setOnItemClickListener
setOnItemClickListener();

//Initialize adapter
adapter = new ContactListAdapter(this, contactList, listener);
//Set adapter
recyclerViewContacts.setAdapter(adapter);

(4) לכל איבר ברשימה יש לקליטת הבחירה של המשתמש באיש קשר מסוים :
private void setOnItemClickListener() {
    listener = new ContactListAdapter.RecycleViewClickListener() {
        @Override
        public void onClick(View v, int position) {
            //now we are creating an intent to go to the WhatsAppSendActivity
            Intent intent = new Intent(getApplicationContext(),
WhatsAppSendActivity.class);

            //now we want to get the contact ID, name and number:
            ContactModel contactChosen = filteredContacts.get(position);
            String name = contactChosen.getName();
            String number = contactChosen.getNumber();

            intent.putExtra("contactName", name);
            intent.putExtra("contactNumber", number);
            intent.putExtra("activeUserName", activeUserName);

            startActivity(intent);

            finish();
        }
    };
}
```

## Broadcast Receivers

### הגדרת ה-Broadcastreceiver

הינה מחלוקת אשר יכולה להזין למסרים (Intent) שימושדים. על מנת ליצור `BroadcastReceiver` צריך ליצור מחלוקת ולרשת מ-`BroadcastReceiver` וליישם את הפונקציה `onReceive`.

דוגמאות לשימושים : האזנה לבטרייה, לשיחה כניסה או ייצא, לשעון, ל-`alarm manager`, לכינסה של sms, לכינסה של push notification.

- `onReceive` ישנה רק פונקציה אחת : לאחר הפונקציה `onReceive` .  
BroadcastReceiver

## **ReminderBroadcast**

ליצירת `notification` מ-`Intent` של `alarmManager` בזמן שהגדר המשמש מראש.  
תוכנות :

(1) יצירת ערך להתראות במידה וצריך – יצירת `NotificationChannel` (בגרסאות חדשות של אנדרואיד  
צריך לשיקן התראות לעורצים כדי שהם יופיעו) :

```
private void createNotificationChannel(){
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O){
        CharSequence name = "LembuitReminderChannel";
        String description = "Channel for Lembuit Reminder";
        int importance = NotificationManager.IMPORTANCE_DEFAULT;
        NotificationChannel channel = new NotificationChannel("notifyLemubit",
name, importance);
        channel.setDescription(description);

        NotificationManager notificationManager =
getSystemService(NotificationManager.class);
        notificationManager.createNotificationChannel(channel);
    }
}
```

(2) יצירת `Intent` עם מידע שיופיע ב-`notification` והפיכתו ל-`PendingIntent` :

```
//Set notificationID & message
Intent notificationIntent = new Intent(
    ProgramUserActivity.this, ReminderBroadcast.class);

notificationIntent.putExtra("message", "Its time to train " + activeUserName);
notificationIntent.putExtra("activeUserName", activeUserName);

//PendingIntent
PendingIntent pendingIntent = PendingIntent.getBroadcast(
    ProgramUserActivity.this, 0, notificationIntent,
    PendingIntent.FLAG_UPDATE_CURRENT
);
//FLAG_CANCEL_CURRENT-if pending already exists, the current one will be
canceled.
```

```
//FLAG_UPDATE_CURRENT-indicates that the pendingIntent
//which we created now can be updated in the future
```

(3) יצרת notification אליו נוסיף את ה-pendingIntent, נגידר בו את הזמן הופעתה וה-AlarmManager ווועל ב-AlarmManager לזמן ההתראה את סוג ה-Alarm (לדוגמא : (RTC\_WAKEUP

```
4) //AlarmManager
//Use with getSystemService to retrieve an AlarmManager
//for receiving intents at a time of your choosing
AlarmManager alarmManager = (AlarmManager) getSystemService(ALARM_SERVICE);

switch (v.getId()){
    case R.id.btnSetAlarm:
        // Set Alarm - we got hour and minutes from: tvHour, tvMinute

        // Create time
        Calendar startTime = Calendar.getInstance();
        startTime.set(Calendar.HOUR_OF_DAY, tvHour);
        startTime.set(Calendar.MINUTE, tvMinute);
        startTime.set(Calendar.SECOND, 1);
        long alarmStartTime = startTime.getTimeInMillis();

        //checking that the time given is not before now
        if (alarmStartTime <= System.currentTimeMillis()){

            Toast.makeText(this, "This time is before now! : " +
startTime.getTime(), Toast.LENGTH_LONG).show();

        }else{
            // Set Alarm
            //AlarmManager.RTC_WAKEUP - wakes up the device to fire the
            pending intent
            //at the specified time
            alarmManager.set(AlarmManager.RTC_WAKEUP, alarmStartTime,
pendingIntent);

            Toast.makeText(this, "Set Done! : " + startTime.getTime(),
Toast.LENGTH_LONG).show();
        }

        break;

    case R.id.btnCancelAlarm:
        // Cancel Alarm
        alarmManager.cancel(pendingIntent);
        Toast.makeText(this, "Canceled", Toast.LENGTH_SHORT).show();
        break;
}
```

(5) קיבלת ה-pendingIntent מה-AlarmManager כזמן ששמור ב-AlarmManager והוא הזמן הנוכחי. בעזרת הבנייה הייעודי – NotificationCompat.Builder – בונים התראה עם המידע מה-pendingIntent

```
6) public class ReminderBroadcast extends BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent) {
```

```

Log.d("RECEIVED", "Notification received!" + " UserName" +
intent.getStringExtra("activeUserName"));

// Get id & message from intent.
String message = intent.getStringExtra("message");
String activeUserName = intent.getStringExtra("activeUserName");

// Call ProgramUserActivity when notification is tapped.
Intent programUserAcIntent = new Intent(context,
ProgramUserActivity.class);

//we need to add this userName in order to take the user to his
//ProgramUserActivity screen
programUserAcIntent.putExtra("activeUserName", activeUserName);

//Intent to go to user's exercise list when clicking on
notification
PendingIntent contentIntent = PendingIntent.getActivity(
    context, 0, programUserAcIntent, 0
);

NotificationCompat.Builder builder = new
NotificationCompat.Builder(context, "notifyLemubit")
    .setSmallIcon(android.R.drawable.ic_dialog_info)
    .setContentTitle("Exercise Time!")
    .setContentText(message)
    .setContentIntent(contentIntent)
    .setAutoCancel(true)
    .setPriority(NotificationCompat.PRIORITY_DEFAULT);

NotificationManagerCompat notificationManager =
NotificationManagerCompat.from(context);

//According to the official documentation
//starting in Android 8.0(API Level 26)
//all notifications must be assigned to a channel

//the notificationID is a unique int for each notification that
must be defined
notificationManager.notify(NotificationID.getID(),
builder.build());

}

public static class NotificationID {
    private final static AtomicInteger c = new AtomicInteger(0);
    public static int getID() {
        return c.incrementAndGet();
    }
}
}

```

## שיקולי תוכן (design)

### : SQLiteOpenHelper

**SQLiteOpenHelper** היא מחלקה עוזר לניהול ייצור מסד נתונים וניהול גרסאות. **SQLiteOpenHelper** מספק כלי עוזר פשוט את המשימות של ייצור ותחול מסד הנתונים אם הוא עדין לא נוצר והמרת תוכן מסד הנתונים כאשר היחסום שלך משודרג וסכימת מסד הנתונים משתנה.

אתה יוצר תת-מחלקה המימושת `onUpgrade(SQLiteDatabase,int,int)` , `onCreate(SQLiteDatabase)` ו`onOpen(SQLiteDatabase)`. מחלקה זו דואגת לפתיחת מסד הנתונים אם הוא קיים, לייצר אותו אם לא, ולשדרוג אותו לפי הצורך. נעשה שימוש בעסקאות כדי לוודא שמסד הנתונים נמצא תמיד במצב הגויוני.

מחלקה זו מקלת על יישומי `ContentProvider` לדוחות את פתיחת וshedrog מסד הנתונים עד לשימוש ראשון, כדי למנוע חסימה של הפעלת יישומים עם שדרוגים ארכטיים טווח של מסדי נתונים.

בנוסף, הרבה יותר פשוט לנחל מחלקה בה יש את כל הפעולות הבסיסיות לשילוף מידע ממושך הנתונים. אין צורך לכתוב את אותו קוד ארוך בו לוקחים, מסננים ומצביעים את המשתנים הספציפיים, ובמקום זאת פשוט קוראים לפועלה המתאימה לסייעת מסויימת. מחלקה זאת ייחודה להעתיקות עם מידע בסיס הנתונים כך שתמיד ברור שם צריך לשולף את המידע מבוסיס הנתונים ואין הבלבול בין מחלקות שונות באפליקציה.

האפשרות השנייה שלי בנושא בסיסי הנתונים ב-`SQLlite` היא להשתמש ב-`SQLiteDatabase`. למרות שבתחילת הפרויקט מימושי בסיס נתונים זהה למזה שקיים בעת ב-`SQLiteOpenHelper` , עברתי לבסיס נתונים מסודר וברור יותר במחלקה היורשת מ-`SQLiteOpenHelper` . זאת מפני שרוב האנשים המתכוונים ב-`AndroidStudio`-ב-`YouTube` וב-`StackOverflow` השתמשו בעוזר כדי לגשת לבסיס הנתונים ולשלוף את המידע המתאים. אני למדתי ממש הרבה עצמים ומחלקות באפליקציה מרווח איך יוטיוברים רבים ומשתמשי `StackOverflow` עושים זאת. לרוב המתכוונים נזورو ב-`SQLiteOpenHelper` לימוש בסיס הנתונים. لكن העדפתית למש את המחלקה הזאת ולנהל את מוסד הנתונים כך.

### : MusicAndTimerService

במחלקה זו יש שירות שמבצע מוזיקה וטיימר ברקע בעת ביצוע התרגיל במסך **ExecuteExerciseActivity**.  
המחלקה יורשת ממחולקת **service** שהיא מובנת בתוך ה-**android**.  
Service הוא שירות שנועד לביצוע פעולות ארוכות טווח. טווח של זמן בו יש סיכוי שהמשתמש עברו לאפליקציה אחרת. כשנעביר מסך ה-**Thread** (תהליכון ברקע) יופסק. ככלומר כנסגור את ה-**activity**-**thread** יופסק. לעומת זאת, ה-**Service** ימשיך לפעול גם כשנעביר בין מסכים. נוכל לעזור את ה-**Service**-**thread** רק בסגירה מלאה של האפליקציה או בהפסקת השירות. לשירות יש תוחלת חיים مثل עצמו. הוא מוגבל ביכולת שלו לבוא באינטראקציה עם רכיבים (**components**) מביתן UI). רוב השירותים הם **started**.  
שימוש ל-**StartService** : שמיעת מוזיקה ברקע.

השירות זהה מופעל בכניסה למסך ביצוע התרגילים (**ExecuteExerciseActivity**) ומופסק ביציאה מסך זה. דרך סיום ביצוע התרגיל או דרך חזרה אחורה למסך התרגילים של המשתמש (**ProgramUserActivity**). מפניה שזו שירות, הוא ימשיך לפעול ברקע אם המשתמש יצא מהאפליקציה וסגור את הטלפון (כל עוד הוא לא סגר אותה באופן מלא). באופן זה המתאים יכול להיכנס למסך ביצוע התרגיל, להפעיל את הטיימר וללכט לבצע את התרגיל כך שהמוזיקה תמשיך לפעול והיא תהיה הסימן לכך שהאפליקציה ממשיכה בטימר. כאשר המשתמש מסיים את ביצוע התרגיל הוא יוכל לפתח את מסך האפליקציה איפה שהוא עזב אותה ולסייע את הביצוע של התרגיל.

כלומר, השימוש בשירות אפשר לי לישם אפשרות להפעיל טיימר ומוזיקה לביצוע התרגיל שימושי גם הטלפון נירדם במצב שינוי וכל עוד לא כיבו את הטלפון למגרי ולא סגרו את האפליקציה למגרי.

**: User Admin Relationship**

חלק שימוש בעקרונות ההורשה והפולימורפיזם, מימושי מחלקה של משתמש ושל מנהל.  
לשימוש השדות הבאים : שם משתמש, סיסמה, משקל, גובה, תאrik לידה,מין, תמונה רפואי.  
`public class User {`

```
private String userName;
private String userPassword;
private int userWeight;
private int userHeight;
private String userBirthDate;
private String userGender;
private String profilePic;
```

מחלקה המנהל יורשת ממחלקת המשתמש ולה שדה חדש : `.isSuperAdmin`

```
public class Admin extends User{
    private boolean isSuperAdmin;
```

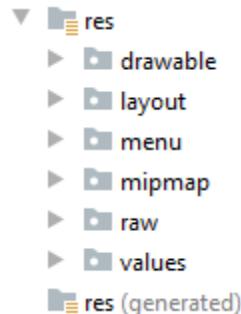
מחלקה זו מאפשרת לתת למשתמשים מסוימים אפשרות לגשת מידע ולאפשרויות שאני לא רוצה שככל משתמש יוכל להטעיק בהם. במצב זה נוצרים מספר דרגות למשתמש : רגיל, מנהל וסופר מנהל.  
לשימוש רגיל אין סמכויות מיוחדות והוא יכול להשתמש באפליקציה באופן רגיל מבלי לדעת דבר על מהם הממשים האחרים דרך האפליקציה.

משתמש מנהל יכול לגשת לרשימה מיוחדת דרך מסך הגדירות (`SettingsActivity`) בווא הוא יכול לראותו ולעבור על רשימה מלאה של כל המשתמשים עם שם ומינם. באופן זה עדין המשתמש מוגבל בלבד בלבד פרטיהם נוספים על משתמשים אחרים שהם : סיסמות, משקלם, גובהם, מינים ודרוגתם.  
משתמש סופר מנהל בעל הכח הרבה סמכויות בתוך האפליקציה. הוא יכול לגשת לרשימה שאליה יכול לגשת המנהל, והוא אף יכול בבחירה אחד המשתמשים הזה לא הוא לגשת למסך עיצוב המשתמש (`EditUserActivity`) בוא הסופר מנהל יכול לעדכן את פרטי המשתמש במקרה חדש ו אף למחוק את המשתמש הנבחר.

בחرتني להשתמש בהורשה ועקרונות הפולימורפיזם מפני שכך יכולתי למש את הרעיון של המנהל והסופר מנהל באפליקציה. רציתי דרך פשוטה וקלה לגשת למידע על המשתמשים מבלי שככל המשתמש יוכל לעשות זאת. לכן, ביצירת בסיס הנטונים נוצר משתמש `alon` שהוא אוטומטית סופר מנהל והוא המשתמש היחיד ביצירת בסיס הנטונים. באופן זה אני יכול לגשת ולבדוק את המידע בסיס הנטונים, ואם ארצת איהפוך המשתמשים נוספים למנהל וסופר מנהלים. עם הפולימורפיזם לפיו אפשר להתייחס אל כמה עצמים שיורשים מאותו מחלקה על כל עצמים מסווג מחלוקת העל יכולתי לספק למשתמשים מסוימים סמכויות מיוחדות ובאותו הזמן להתייחס אל כלם כאלו משתמשים. בנוסף, באופן זה כשם המשתמשים בפעולה `getUserLevel()` של מחלקות המשתמש והמנהל, לפי הפולימורפיזם, מקבלים 0 עבור איבר שמתיחסים אליו כאל מחלקה משתמש הוא בעצם מסווג מנהל.

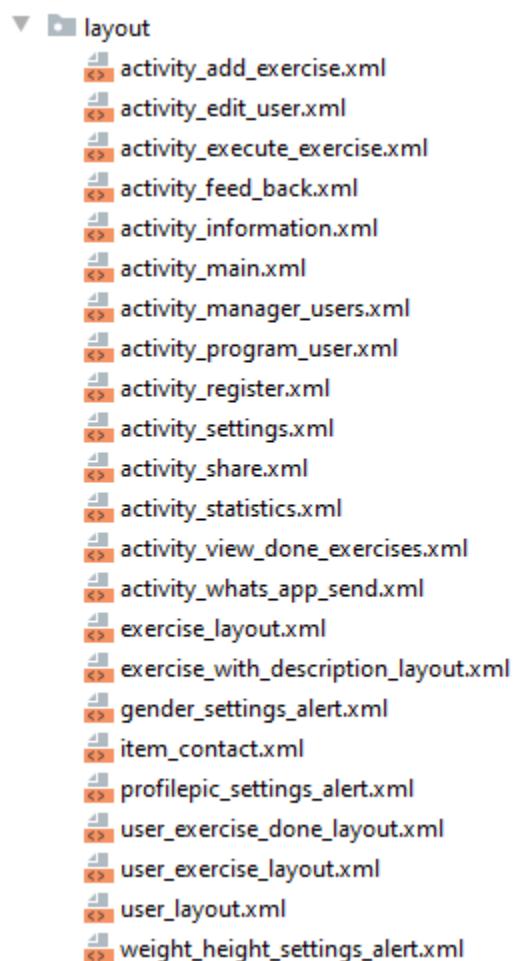
## **Resources**

פירוט ה- resources השונים המשמשים לפרויקט, עם הסבר לגבי כל אחד מהם :



### Layouts

פירוט קבצי ה-XML ולאיזה Activity כל אחד מהם מותאים :



**AddExerciseActivity** : עיצוב של מסך activity\_add\_exercise -

**EditUserActivity** : עיצוב של מסך activity\_edit\_user -

**ExecuteExerciseActivity** : עיצוב של מסך activity\_execute\_exercise -

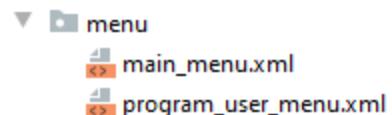
**FeedbackActivity** : עיצוב של מסך activity\_feed\_back -

**InformationActivity** : עיצוב של מסך activity\_information -

**MainScreenActivity** : עיצוב של מסך activity\_main -  
**ManagerUserActivity** : עיצוב של מסך activity\_manager\_user -  
**ProgramUserActivity** : עיצוב של מסך activity\_program\_user -  
**RegisterActivity** : עיצוב של מסך activity\_register -  
**SettingsActivity** : עיצוב של מסך activity\_settings -  
**ShareActivity** : עיצוב של מסך activity\_share -  
**StatisticsActivity** : עיצוב של מסך activity\_statistics -  
**ViewExercisesResultsActivity** : עיצוב של מסך activity\_view\_done\_exercises -  
**WhatsAppSendActivity** : עיצוב של מסך activity\_whats\_app\_send -  
**AddExerciseActivity** : עיצוב של תרגילים ב-ListView במסך exercise\_layout -  
**AddExerciseActivity** : עיצוב של תרגילים ב-AlertDialog במסך AlertDialog\_exercise\_with\_description\_layout -  
**SettingsActivity** : עיצוב של AlertDialog לשינוי מין משתמש במסך gender\_settings\_alert -  
**ShareActivity** : עיצוב של אנשי קשר ב-RecyclerView במסך item\_contact -  
**SettingsActivity** : עיצוב של AlertDialog לשינוי תמונה משתמש במסך profilepic\_settings\_alert -  
**SettingsActivity** : עיצוב של ListView שlógically שboxedו של ListView ב-ListView במסך user\_exercise\_done\_layout -  
**ViewExercisesResultsActivity**  
**ProgramUserActivity** : עיצוב של תרגילים שלא בוצעו ב-ListView -  
**ManagerUserActivity** : עיצוב של משתמש בראשימה במסך user\_layout -  
**ProgramUserActivity** : עיצוב של AlertDialog לשינוי משקל וגובה של משתמש במסך weight\_height\_settings\_alert -  
**SettingsActivity**

## Menus

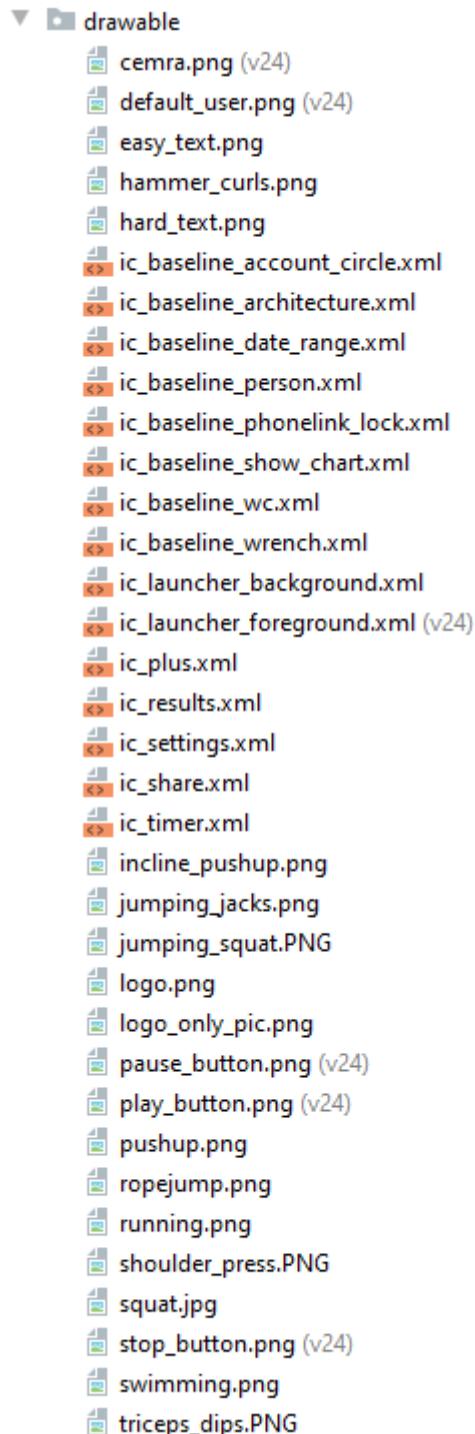
פירוט קבצי ה-XML וציון איזה תפריט כל אחד מהם מממש ובאיזה Activity נעשה בו שימוש:



**main\_menu** : מממיש את התפריט של המסך הראשי MainScreenActivity הכללת אפשרות לעبور למסך המידע על האפליקציה (about) ואפשרות לצאת מהאפליקציה (exit). -  
**Program\_user\_menu** : מממיש את התפריט של מסך התרגילים של המשתמש Program\_user\_menu. תפריט זו כוללת מספר אפשרויות:  
 1) אפשרות ללבת למסך הטבלה StatisticsActivity.(graph)  
 2) אפשרות לעبور למסך תוצאות התרגילים שבוצעו ViewExercisesResultsActivity.(results)  
 3) אפשרות לעبور למסך שיתוף המידע ShareActivity.(share)  
 4) אפשרות לעبور למסך ההגדרות של המשתמש SettingsActivity.(settings)

## Drawables

הציגת תמונות תחת כותרות מתאימות :



**סוגי תמונות:****: (Asset Studio) Vector Asset -**

- ic\_baseline\_account\_circle(1)
- ic\_baseline\_architecture(2)
- ic\_baseline\_date\_range(3)
- ic\_baseline\_person(4)
- ic\_baseline\_phonelink\_lock(5)
- ic\_baseline\_show\_chart(6)
- ic\_baseline\_wc(7)
- ic\_baseline\_wrench(8)
- ic\_launcher\_background(9)
- v24\ic\_launcher\_foreground(10)
- plus(11)
- results(12)
- settings(13)
- share(14)
- timer(15)

**: Exercises -**

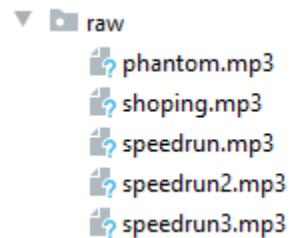
- hammer\_curls(1)
- incline\_pushup(2)
- jumping\_jacks(3)
- jumping\_squat(4)
- pushup(5)
- ropejump(6)
- running(7)
- shoulder\_press(8)
- squat(9)
- swimming(10)
- triceps\_dips(11)

**: תמונות של איקונים -**

- cemra (1)
- default\_user(2)
- easy\_text(3)
- hard\_text(4)
- logo(5)
- logo\_only\_pic(6)
- pause\_button(7)
- play\_button(8)
- stop\_button(9)

## Raw

הציג השירים ב- resources :



כל אחד מ-5 השירים האלה נמצא בסיס הנתונים בטבלת השירים.  
בעת ביצוע התרגיל במקץ ביצוע התרגילים (ExecuteExerciseActivity) מופעל אחד מהשירים האלה באופן רנדומלי.

## קובץ

אני משתמש באפליקציה בשני סוגי קבצים :

### SQLiteDataBase

בבסיס הנתונים קבצים SQLite flies שהם מאוכסנים בזיכרון פנימי. מחלקה DataBaseHelper שומרת נתונים שהיא מקבלת מחלוקת אחרות בקבצים כאלה בסיס הנתונים. קובץ DB הוא קובץ שומר את הנתונים באפליקציה (משתמשים, תרגילים, תרגילי משתמש ושירים).

### קובץ info פנימי בזיכרון הטלפון

מדובר בזיכרון פנימי טלפון מחוץ לאפליקציה. מדובר בקובץ שמוסמך על ידי SharedPreferences בקובץ info טלפון לשימרת תמונה משתמש באפליקציה.

### הפעולות בהן נעשה שימוש בקובץ

- PictureFileHelper פעולות במחלקה זו
- פעולות מרכזיות במחלקה זו DataBaseHelper -

**סכמת Database**

טבלאות והקשרים :

טבלה Users				
טיאור	מפתח זר	טיפוס הנתונים	שם השדה	
שם המשתמש		TEXT	userName	
סיסמה של משתמש		TEXT	userPassword	
דרגת משתמש. כאשר : 0 - גיל -1 מנהל -2 סופר מנהל		INTEGER	userLevel	
משקל משתמש		INTEGER	userWeight	
גובה משתמש		INTEGER	userHeight	
תאריך לידה של משתמש		TEXT	userBirthDate	
מין משתמש		TEXT	userGender	
תמונה פרופיל של משתמש		INTEGER	profilePic	

טבלה Exercises				
טיאור	מפתח זר	טיפוס הנתונים	שם השדה	
ID של תרגיל		INTEGER	exerciseID	
שם תרגיל		TEXT	exerciseName	
תמונה תרגיל		INTEGER	exercisePic	
תיאור תרגיל		TEXT	exerciseDetail	

טבלה UserExercises				
טיאור	מפתח זר	טיפוס הנתונים	שם השדה	
ID של תרגיל משתמש		INTEGER	userExerciseID	 <b>AUTOINCREMENT</b>
שם משתמש	טבלה Users	TEXT	userName	
ID של תרגיל	טבלה Exercises	INTEGER	exerciseID	
תאריך של ביצוע התרגיל		TEXT	date	
זמן שלקח לבצע את התרגיל		INTEGER	time	
דרוג התרגיל בסקלה : מ-1 עד 5. כאשר : 1-הכי קשה		INTEGER	rating	

5-הכי קל				
מספר הביצועים שחיו בביוץ התרגיל		INTEGER	repetition	

<u>Songs</u> טבלת					
תיאור	מפתח זר	טיפוס הנתונים	שם השדה		
ID של שיר		INTEGER	songID		
שם שיר		TEXT	songName		
ID של קובץ raw של השיר		INTEGER	songMP3		

## מדריך משתמש

### מטרת המערכת

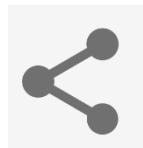
המערכת עוקבת אחרי תרגילים שהמשתמש עושה בזמןו החופשי. באפליקציה ישנו 11 תרגילים פשוטים הכלולים הסבירים מפורטים על האופן בו יש לבצע אותם. מטרת האפליקציה היא **לסייע למתאמנים שרצוים לסדר את התוצאות התרגילים שלהם במקומות אחד בו הם יכולים בקלות לנתח ולהבין את המוגנה שלהם בגיןibus קודמיים.** זאת על ידי מעבר על רשיימה מסודרת של תוצאות התרגילים שביצעו. בנוסף, המשתמש יכול לנתח טבלת עמודות הכוללת נתונים על מספר הביצועים, דירוג הקoshi, החודש והשנה של התרגילים שביצעו על ידי המשתמש. מתאם שרצה **לשתח את חבריו בклות עם תוצאות אימונו** יכול באפליקציה זו לחושף את חבריו ב- WhatsApp לכל תוצאות התרגילים שלו דרך האפליקציה. אם למשתמש יש חברים שגם רוצים להצטרף לאימון הנגדל, הוא יכול **لتת לחברו להתחבר כמשתמשים שונים לאותו הטלפון.** כך הוא מעודד אותם גם לעשות אימון כושר ובמידה ויש להם טלפון המשתמש יכול לשכנע אותם גם להוריד את האפליקציה!

בכניסה ראשונה לאפליקציה **מסך הפתיחה** (MainScreenActivity) המשתמש יכול לעבור דרך התפריטי  **במסך זה למסך המידע** (InformationActivity). במסך זה ניתן לראות מהם התרגילים שיש באפליקציה עוד לפני שנרשמים. לאחר מכן, המשתמש נדרש להירשם כדי להתחיל את האימון הראשון שלו. דרך כפתור ה-Register **במסך הפתיחה** (MainScreenActivity) המשתמש עבר **למסך הרישום** (RegisterActivity). כדי להירשם יש לצלם תמונה פרופיל דרך המצלמה בטלפון ולהכנס שם משתמש, סיסמה, משקל, גובה, תאריך לידיה ומין. בלחיצה על FINISH המשתמש חוזר **למסך הפתיחה** (MainScreenActivity) והוא יכול להירשם ב-LOGIN.

לאחר לחיצה על כפתור ה-LOGIN המשתמש עובר **למסך התרגילים של המשתמש** (ProgramUserActivity) בו הוא רואה את כל התרגילים שלו שהוא עוד לא ביצע. כדי להוסיף תרגילים ליבצעו עליו ללחוץ על כפתור הפלוס ולעבור **למסך הוספה התרגילים** (AddExerciseActivity) בו המשתמש בוחר תרגיל לביצוע מתוך רשימה של תרגילים. לאחר שבחר תרגיל מסוים ו אישר את הוספו, המשתמש עובר חזרה **למסך התרגילים של המשתמש** (ProgramUserActivity) שם הוא יראה את התרגיל החדש ברשימת תרגילי המשתמש. לחיצה על תמונה התרגיל או על הטקסט לידיה (כל מקום על האיבר בלבד כפתור המהיקה DELETE) תעביר את המשתמש **למסך ביצוע התרגילים** (ExecuteExerciseActivity). במסך זה המשתמש מבצע את התרגיל עם הפעלת טימר ומוזיקה תוך כדי. לאחר שסיים לבצע את התרגיל ולחץ כל כפתור ה-STOP שהוא ריבוע, המשתמש עובר **למסך האחרון** בתהליך **ביצוע התרגיל שהוא מסך חשוב על ביצוע התרגיל** (FeedbackActivity). במסך זה המשתמש מכניס את מספר החזרות בתרגיל ואת דרגת הקושי של ביצוע התרגיל. לאחר סיום התהליך בלחיצה על

כפתרו ה-FINISH המשמש חוזר למסך התרגולים של המשתמש (ProgramUserActivity) באופן דומה המשתמש יכול לבצע תרגילים כל הזמן.

בנוסף לתהליך זה של האפליקציה המשמש יכול לנתח, לעקוב ולהבין את מגמת האימוניים שלו דרך גרפים, רשימות ומידע שהוא שולח לחבריו ב-whatsApp.



בלחיצה על כפתרו השיתוף בתפריט במסך התרגולים של המשתמש (ProgramUserActivity) המשתמש עובר למסך **שיתוף המידע** (ShareActivity) בו, באישור גישה לאנשי הקשר בטלפון שלו, יכול לבחור אחד מאנשי הקשר שיש לו ובמסך הבא שהוא מסך **שליחת הודעה לאיש קשר באופןוטzapf** (WhatsAppSendActivity) לשולח לאותו איש קשר הודעה מפורט עם סיכומים על כל נתוני התרגילים שבוצעו על ידי המשתמש.



בלחיצה על כפתרו המידע בתפריט במסך התרגולים של המשתמש (ProgramUserActivity) המשתמש עובר למסך **נתונים על התרגילים שבוצעו בראשימה** (ViewExercisesresultsActivity). במסך זה המשתמש רואה את ה-BMI שלו ואת תוכאותו תרגilio באופן ברור.



בלחיצה על כפתרו הסטטיסטיות בתפריט במסך התרגולים של המשתמש (ProgramUserActivity) המשתמש עובר למסך **טבלה** (StatisticsActivity). במסך זה מופיעה טבלה chart בה יש נתונים על כל התרגילים שביצע המשתמש. בעזרת צבע העמודות, מיקום בציר ה-x וגובה העמודות יכול המשתמש להבין מה מגמותו בכל סוג התרגולים. מגמות של האטה או השתפרות.



בלחיצה על כפתור ההגדרות בתפריט במסך התרגילים של המשתמש (ProgramUserActivity) המשתמש עובר למסך ההגדרות (SettingsActivity). מסך זה המשתמש יכול לשנות את כל הנתונים שלו בסיס הנתונים מלבד שמו (זאת מפני שהוא מפתח בסיס הנתונים).

באפליקציה יש סמכויות מיוחדות למנהלים וסופר מנהלים.

למשתמש באפליקציה זו יש שלושה דרגות אפשרויות :

0- רגיל (Normal)

1- מנהל (Admin)

2- סופר מנהל (SuperAdmin)

כ"כ שבמסך ההגדרות (SettingsActivity) מופיע למשתמש שהוא מנהל או סופר מנהל אפשרות בלחיצה לעבור למסך ניהול המשתמשים (ManagerUsersActivity). אולם, רק סופר מנהל יוכל באמצעות דרך מסך זה לגשת למסך עיצוב המשתמשים (EditUserActivity) דרכו סופר מנהל יוכל לגשת לכל המשתמשים חוץ מעצמו, ולעדכן את פרטי כל המשתמשים חוץ משםם ולמחוק אותם.

## יכולות המערכת

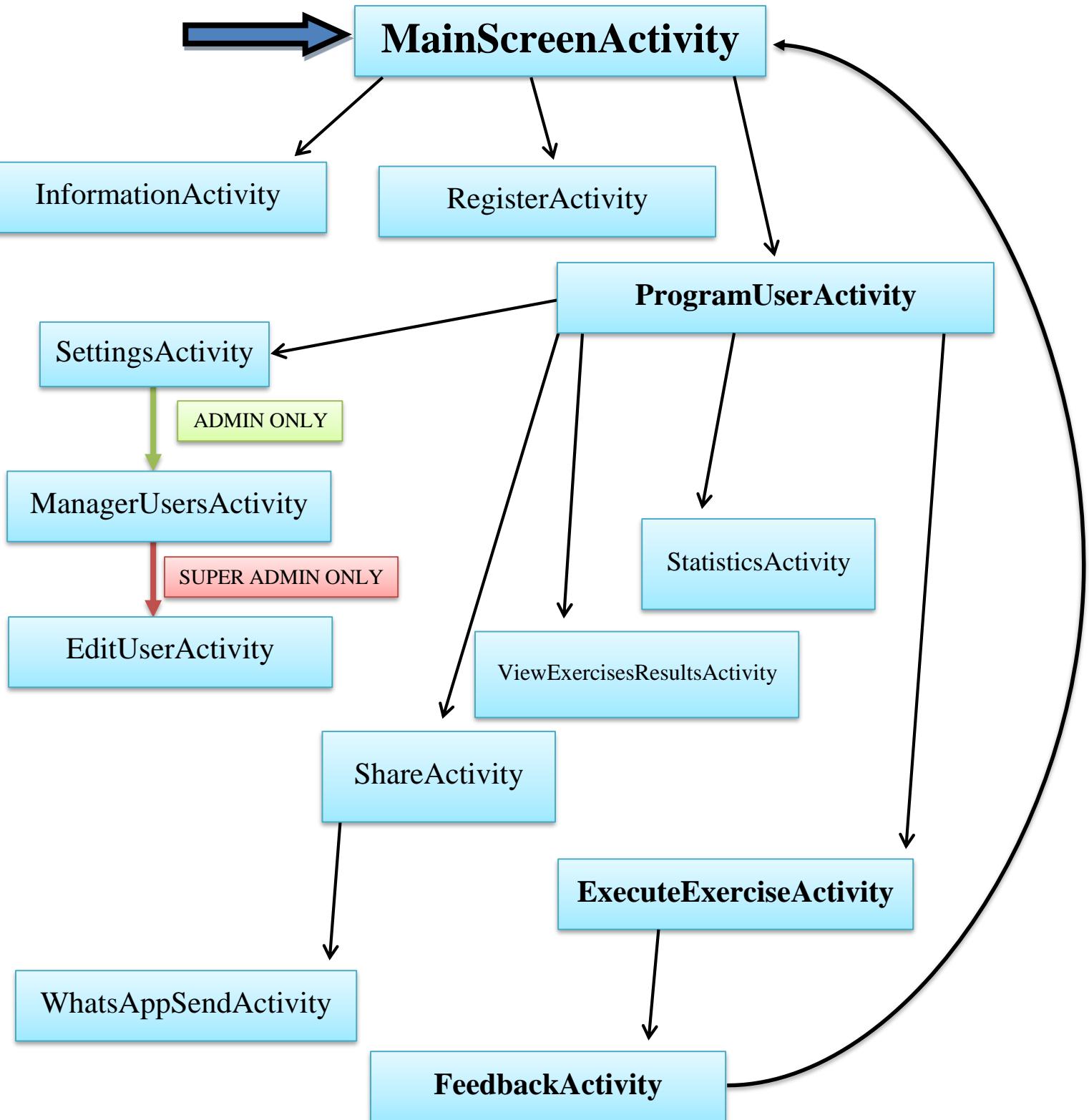
תיאור האופציות שנדרשת המערכת למשתמש (Feature List) :

- רישום משתמש
- שינוי פרטי המשתמש על ידי סופר מנהל או מעבר על רשימת משתמשים בלבד כמנהל
- בחירת תרגיל מרשימה
- שימרת תוצאות של ביצוע תרגילים לפי זמנים
- ביצוע תרגילים מסוימים
- הפעלת התראה לזמן קבוע מראש
- הפעלת מוזיקה ברקע ביצוע התרגיל
- ניתוח סטטיסטי של תוצאות התרגילים מעבר על רשימה וטבלה של תוצאות התרגילים שבוצעו
- שיתוף תוצאות ביצוע התרגילים עם אנשי קשר ב-WhatsApp

## תפועל המעלכת

תרשים זרימה בין המסכים

מתאר את אופן הניווט בין המסכים השונים:



## הרשאות

פירוש הרשאות שהאפליקציה דורשת והסביר על כל הרשותה :

```
uses-permission android:name="android.permission.READ_CONTACTS"
```

נדרש כדי לאפשר קריאה של אנשי הקשר בטלפון.

### הצהרות:

הצהרה של ReminderBroadcast :

```
<receiver
    android:name=".ReminderBroadcast"
    android:enabled="true" />
```

הצהרה של שירות MusicAndTimerService :

```
<service android:name=".MusicAndTimerService"
    android:exported="false"/>
```

פיצר מצלמה :

```
<uses-feature
    android:name="android.hardware.Camera"
    android:required="true" />
```

## דרישות מיוחדות ומוגבלות

יש דרישת להימצאות אפליקציה חיצונית היכולת לצלם בטלפון. אך אם אין כזאת אפליקציה, המשמש יוכל להמשיך להתאמנו כרגיל רק שהוא לא יכול לצלם לאפליקציה תמורה משתמש בעת רישום או בעת שינוי פרטי המשתמש כל עוד אין אפליקציה חיצונית כזו.

כדי לגשת לאנשי הקשר בטלפון נדרש לאשר זאת בכניסה למסך אנשי הקשר באפליקציה (ShareActivity). על מנת לשולח לאיש קשר מסוים הודעה באוטומטית נדרש שהאפליקציה WhatsApp תהיה מותקנת בטלפון. אחרת תופיע בקשה להוריד WhatsApp בטלפון ובאפשרות של שליחת הודעה דרך האפליקציה לאיש קשר ב-WhatsApp לא תהיה אפשר.

## **גירסת Android מינימלית**

גירסת Android המינימלית שנדרשת כדי להריץ את האפליקציה :

מינימום SDK : 19

מקסימום SDK : 30

## **מכשוריהם עליהם נבדקה המערכת**

מכשוריהם בהם נבדקה המערכת : Galaxy Tab S6 Lite ,Pixel 2 API 27 ,Xiaomi Mi 9T

## רפלקציה

העובדת עברית הייתה מתוגרת ומהנה. במסגרת הפרויקט הייתה לי הזדמנות לתכנן ולבנות אפליקציה רצינית שמאפשרת את כל הרווחנות שרציתי לישם בה. מהמורה שלי ניצן קיבלתי את הכלים הבסיסיים עבור בניית מסכימים, מעבר בינהם ובנית בסיס הנתונים. לאחר מכן אני חקרתי ביוטיוב ובאתרים נוספים באינטרנט איך כדי למשביסיס נתונים ואני בונים מחלקות ופעולות שיעזרו לי למשב את הרווחנות שלי בפועל. להמשך אני לוקח את הידע שצברתי על ניהול מוצר נטונים גדול ומסודר דרך מחלקת נפרדת, ואת העבודה והארגון של אפליקציה עם מספר מסכימים בעל תהליכי אחד מרכזי ומסכימים נוספים לניטוח מידע והגדירות. כדי למשב את כל הרווחנות שהיו לי בהתחלה נדרשתי לבצע מחקר עמוק על הדרכים שקיים לשימוש מידע, התאמת נתונים לרשיימה, שימוש בשירותים להפעלת טימיר ומזיקה, שימוש ב-ContentProvider שישיג לי מידע מאפליקציה אחרת בטלפון שהזיקה אנשי קשר ושימוש ב-BroadcastReceiver ליצירת התראה בזמן קבוע מראש. על מנת למשב את כל היכולות הללו הייתה צורך לראות אילו מחלקות ופעולות מתאימות לבדוק לצרכים שלי. המחקר והלמידה העצמית במסגרת הפרויקט הייתה מתוגרת, ובנוסף לכך גם הרבה מאוד זמן לא הייתה בבית הספר עקב התפרצות מגפת הקורונה וסגירת בתיה הספר בעקבות זאת. לכן, את רוב הלמידה הייתה צריכה לבצע בלבד.

לאחר סיום הפרויקט אני מאמין שלמרות הקשיים שהתלו ל飯店 בניית אפליקציה במסגרת לימודי התוכנה שלי בבית הספר הייתה מאוד מעשרה ועזרה לי לקדם צעד אחד קדימה את ההבנה שלי בתוכנה ובניהול פרויקט רב מחלקתי.

לו הייתה מתחילה עם הידע החדש שצברתי הייתה מרחיב את התקשרות של האפליקציה עם אפליקציות אחרות ומשתמש יותר במידע חיצוני כמו מזג האוויר ומיקום בצדה"א.

אני סבור שעבודתי הייתה עיליה ביחס למצב שהיה במהלך תקופה רובה שנת הלימודים. אם בית הספר היה פתוח לאורך כל השנה אני חשב שהייתי גומר את הפרויקט קצר יותר מוקדם מכך שגמרתי אותו עכשו.

לסיום, אני מרגיש מסופק מהאפליקציה FitNote. בעיני היא יכולה לסייע לרבים כמווני שרצו להתאמן ולתרגל תרגילי כושר וגם רוצחים לנתח את תוכאותיהם ולעקוב אחרי מוגמת הביצוע שלהם, בין אם הם השתפרו ובין אם יש האטה ביצוע.

## ביבליוגרפיה

מקורות מידע שנעוזרת בהם :

- ללימוד הנושא
- לגיבוש הרעיון
- לכנתית הקוד

YouTube, StackOverflow, appschool, codeplayon, github, tutorialspoint, developers.android

## נספחים

### תיעוד – קוד הפרויקט

#### עוזר בנית מוסד הנתונים :( DataBaseHelper )

מחלקה עוזרת לבניית בסיס הנתונים מסווג SQLite לאפליקציה וניהול הנתונים בה.  
קוד המחלקה :

```
package com.example.fitnote13022021;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.util.Log;

import androidx.annotation.Nullable;

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

public class DataBaseHelper extends SQLiteOpenHelper {

    //Initialize variables
    private static final String DATABASE_NAME = "fitnote_database12345678";
    private static final int DATABASE_VERSION = 1;

    private static final String TABLE_USER = "Users";
    private static final String TABLE_EXERCISES = "Exercises";
    private static final String TABLE_USEREXERCISES = "UserExercises";
    private static final String TABLE_SONGS = "Songs";

    DataBaseHelper(Context context){
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }

    public DataBaseHelper(@Nullable Context context, @Nullable String name,
    @Nullable SQLiteDatabase.CursorFactory factory, int version) {
        super(context, name, null, version);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        //Create tables
        String tableUser = "CREATE TABLE " + TABLE_USER + " (" +
                "userName TEXT PRIMARY KEY, userPassword TEXT, userLevel INTEGER, userWeight INTEGER, userHeight INTEGER, " +
                "userBirthDate TEXT, userGender TEXT, profilePic INTEGER)";
        //userLevel 0-normal, 1-admin, 2-superAdmin
        String tableExercise = "CREATE TABLE " + TABLE_EXERCISES + " (" +
                "exerciseID INTEGER PRIMARY KEY, exerciseName TEXT, exercisePic INTEGER, exerciseDetail TEXT)";
        String tableUserExercises = "CREATE TABLE " + TABLE_USEREXERCISES + " (" +
                "userExerciseID INTEGER PRIMARY KEY AUTOINCREMENT, userName TEXT, exerciseID INTEGER, date TEXT, time INTEGER, rating INTEGER, repetition INTEGER)";

    }
}
```

```

String tableSongs = "CREATE TABLE " + TABLE_SONGS + " (songID INTEGER
PRIMARY KEY, songName TEXT, songMP3 INTEGER);"

db.execSQL(tableUser);
db.execSQL(tableExercise);
db.execSQL(tableUserExercises);
db.execSQL(tableSongs);

// adding exercises
// Exercise Table;
// (exerciseID INT PRIMARY KEY, exerciseName TEXT,
// exercisePic INT, exerciseDetail TEXT)
db.execSQL("INSERT INTO " + TABLE_EXERCISES + " VALUES (1, 'Push Up',
"+R.drawable.pushup+", '1. Get down on all fours, placing your hands slightly
wider than your shoulders. ' +
"\n" +
"2. Straighten your arms and legs. " +
"\n" +
"3. Lower your body until your chest nearly touches the floor. " +
"\n" +
"4. Pause, then push yourself back up. " +
"\n" +
"5. Repeat. ')");
db.execSQL("INSERT INTO " + TABLE_EXERCISES + " VALUES (2, 'Squat',
"+R.drawable.squat+, '1. Find a foot stance that feels best for you. Pointing
your toes slightly outwards helps some, but keeping them parallel is fine, too. If
you're not sure what's best, start by putting your feet shoulder-width apart and
pointed about 15 degrees outwards. ' +
"\n" +
"2. Tense your abs like someone is about to punch you. " +
"\n" +
"3. Look straight ahead and stand tall!')");
db.execSQL("INSERT INTO " + TABLE_EXERCISES + " VALUES (3, 'Running',
"+R.drawable.running+, 'TIPS: When you first start out, try alternating between
running and walking during your session. As time goes on, make the running
intervals longer until you no longer feel the need to walk. ' +
"Give yourself a few minutes to cool down after each run by
walking and doing a few stretches. Try our post-run stretch routine.')");
db.execSQL("INSERT INTO " + TABLE_EXERCISES + " VALUES (4, 'RopeJump',
"+R.drawable.ropejump+, 'Get a large rope that you can pass under your feet. Move
the rope under your feet while you jump and repeat.')");
db.execSQL("INSERT INTO " + TABLE_EXERCISES + " VALUES (5, 'JumpingJacks',
"+R.drawable.jumping_jacks+, 'JumpingJacks: Stand upright with your legs
together, arms at your sides. ' +
"\n" +
"Bend your knees slightly, and jump into the air. " +
"\n" +
"As you jump, spread your legs to be about shoulder-width apart.
Stretch your arms out and over your head. " +
"\n" +
"Jump back to starting position. " +
"\n" +
"Repeat. ')");

```

```

db.execSQL("INSERT INTO " + TABLE_EXERCISES + " VALUES (6, 'TricepsDips',
"+R.drawable.triceps_dips+", '1. You can do it from a chair or a bench, "+
"you can even do it on the floor. "+
"\n" +
"2. So just getting yourself into position to start by rolling
your shoulders down. "+
"You want to start from good posture. Sometimes we tend to slump
forward and then were gonna potentially cause an extra strain on the shoulder
joint. "+
"\n" +
"3. So set yourself up by rolling the shoulders back, opening up
the chest. Bringing your hands directly underneath your shoulders onto the bench
or onto the ground. "+
"And youre gonna take your legs out keeping your knees bent." +
"\n" +
"4. If you wanted to make it harder you could extend the legs. "+
"So start with the easier option, see how you feel first. "+
"\n" +
"5. Youre gonna bend the elbows, lowering the hips down, and then
exhale to extend the elbows and lift your body up. "+
"Inhale down, exhale up.'");

db.execSQL("INSERT INTO " + TABLE_EXERCISES + " VALUES (7,
'InclinePushUp', "+R.drawable.incline_pushup+", '1. Place your hands on the edge
of the elevated surface. "+
"\n" +
"2. Step your feet back so your legs are straight and your arms
are perpendicular to your body. "+
"\n" +
"3. Inhale as you slowly lower your chest to the edge of your
platform. "+
"\n" +
"4. Pause for a second. "+
"\n" +
"5. Exhale as you push back to your starting position with your
arms fully extended.'");

db.execSQL("INSERT INTO " + TABLE_EXERCISES + " VALUES (8, 'JumpingSquat',
"+R.drawable.jumping_squat+, 'Stand tall with your feet hip-width apart. "+
"\n" +
"Hinge at the hips to push your butt back and lower down until
your thighs are parallel to the floor. "+
"\n" +
"Allow your knees to bend 45 degrees when you land, and then
immediately drop back down into a squat, and jump again.'");

db.execSQL("INSERT INTO " + TABLE_EXERCISES + " VALUES (9, 'HammerCurls',
"+R.drawable.hammer_curls+, 'Step 1 "+
"Stand up straight with a dumbbell in each hand, holding them
alongside you. Your palms should face your body. Keep your feet hip-width apart
and engage your core to stabilize the body. "+
"\n" +
"Step 2 "+
"Keep your biceps stationary and start bending at your elbows,
lifting both dumbbells. "+
"\n" +
"Step 3 " +

```

```

        "Lift until the dumbbells reach shoulder-level, but don't actually
touch your shoulders. Hold this contraction briefly, then lower back to the
starting position and repeat.'');");
    db.execSQL("INSERT INTO " + TABLE_EXERCISES + " VALUES (10,
'ShoulderPress', "+R.drawable.shoulder_press+", 'Stand with feet shoulder-width
apart and hold the dumbbells at shoulder height with your elbows at a 90-degree
angle. "+

"\n" +
"Slowly lift the dumbbells above your head without fully
straightening your arms. Pause at the top. "+

"\n" +
"Slowly return to the start position.')");

    db.execSQL("INSERT INTO " + TABLE_EXERCISES + " VALUES (11, 'Swimming',
"+R.drawable.swimming+, '1. Float with your face in the water, your body straight
and horizontal. Stack your hands and keep your arms and legs long. " +
"Point your thumbs down. " +

"\n" +
"2. Press your hands out and back in a circle, elbows high. Lift
your head slightly and inhale. " +

"\n" +
"3. Bring your hands together in front of your shoulders, thumbs
pointing up. Keep your elbows close to your body. Simultaneously bend your knees,
bringing your feet toward your butt and pointing your feet outward. " +

"\n" +
"4. Reach your arms forward. Kick out and back in a circle then
snap your feet together. Drop your head underwater and exhale. " +

"\n" +
"5. Glide forward and repeat.')");

    // adding songs
    db.execSQL("INSERT INTO " + TABLE_SONGS + " VALUES (1, 'speedrun',
"+R.raw.speedrun+"));
    db.execSQL("INSERT INTO " + TABLE_SONGS + " VALUES (2, 'speedrun2',
"+R.raw.speedrun2+"));
    db.execSQL("INSERT INTO " + TABLE_SONGS + " VALUES (3, 'speedrun3',
"+R.raw.speedrun3+"));
    db.execSQL("INSERT INTO " + TABLE_SONGS + " VALUES (4, 'phantom',
"+R.raw.phantom+"));
    db.execSQL("INSERT INTO " + TABLE_SONGS + " VALUES (5, 'shoping',
"+R.raw.shoping+"));



//Table Users:
//(userName TEXT PRIMARY KEY, userPassword TEXT, userLevel INTEGER,
// userWeight INTEGER, userHeight INTEGER, userBirthDate TEXT, userGender
TEXT, profilePic INTEGER)
//adding user alon - me the SUPER ADMIN
db.execSQL("INSERT INTO " + TABLE_USER + " VALUES('alon', 123, 2, 90, 90,
'APR 2 2003', 'male', 'alon')");



//adding userExercises
//JAN FEB MAR APR MAY JUN JUL AUG SEP OCT NOV DEC
//(userExerciseID INTEGER PRIMARY KEY AUTOINCREMENT, userName TEXT,
```

```

    // exerciseID INTEGER, date TEXT, time INTEGER, rating INTEGER, repetition
    INTEGER)

        db.execSQL("INSERT INTO " + TABLE_USEREXERCISES + " VALUES(1, 'alon', 1,
'MAR 7 2021', 100, 1, 100)");
        db.execSQL("INSERT INTO " + TABLE_USEREXERCISES + " VALUES(2, 'alon', 2,
'JAN 8 2021', 10, 2, 10)");
        db.execSQL("INSERT INTO " + TABLE_USEREXERCISES + " VALUES(3, 'alon', 3,
'FEB 9 2021', 60, 3, 60)");
        db.execSQL("INSERT INTO " + TABLE_USEREXERCISES + " VALUES(4, 'alon', 4,
'APR 10 2020', 30, 4, 30)");
        db.execSQL("INSERT INTO " + TABLE_USEREXERCISES + " VALUES(5, 'alon', 5,
'JUN 11 2020', 50, 5, 50)");
        db.execSQL("INSERT INTO " + TABLE_USEREXERCISES + " VALUES(6, 'alon', 6,
'SEP 25 2020', 100, 1, 100)");
        db.execSQL("INSERT INTO " + TABLE_USEREXERCISES + " VALUES(7, 'alon', 7,
'MAR 7 2021', 20, 2, 20)");
        db.execSQL("INSERT INTO " + TABLE_USEREXERCISES + " VALUES(8, 'alon', 8,
'JAN 8 2021', 56, 3, 56)");
        db.execSQL("INSERT INTO " + TABLE_USEREXERCISES + " VALUES(9, 'alon', 9,
'FEB 9 2021', 34, 4, 34)");
        db.execSQL("INSERT INTO " + TABLE_USEREXERCISES + " VALUES(10, 'alon', 10,
'APR 10 2020', 70, 5, 70)");
        db.execSQL("INSERT INTO " + TABLE_USEREXERCISES + " VALUES(11, 'alon', 11,
'MAR 11 2021', 120, 1, 110)");

    }

//Delete database
public static void deleteDatabase(Context mContext) {
    mContext.deleteDatabase(DATABASE_NAME);
}

@Override
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
    //Drop Existing Table
    db.execSQL("DROP TABLE IF EXISTS " + TABLE_USER);
    db.execSQL("DROP TABLE IF EXISTS " + TABLE_EXERCISES);
    db.execSQL("DROP TABLE IF EXISTS " + TABLE_USEREXERCISES);
    onCreate(db);
}

//method to add user
public boolean insertUser(User user) {
    //getWritableDatabase - method from the default properties of
SQLiteOpenHelper
    //getWritableDatabase for insert actions...getReadableDatabase for select
(read) actions.
    SQLiteDatabase db = this.getWritableDatabase();

    // User Table:
    // (userName TEXT PRIMARY KEY, userPassword TEXT, userLevel INTEGER,

```

```

    // userWeight INTEGER, userHeight INTEGER, userBirthDate TEXT, userGender
    TEXT, profilePic INTEGER)
    ContentValues cv = new ContentValues();
    cv.put("userName", user.getUserName());
    cv.put("userPassword", user.getUserPassword());
    cv.put("userLevel", 0);
    cv.put("userWeight", user.getUserWeight());
    cv.put("userHeight", user.getUserHeight());
    cv.put("userBirthDate", user.getUserBirthDate());
    cv.put("userGender", user.getUserGender());
    cv.put("profilePic", user.getProfilePic());

    // insert - success variable... 1)positive -> it was inserted...2) -1 ->
it was a fail
    long insert = db.insert(TABLE_USER, null, cv);

    if(insert == -1){
        return false;
    }else{
        return true;
    }
}

//method to add user
public boolean insertAdmin(Admin admin) {
    //getWritableDatabase - method from the default properties of
SQLiteOpenHelper
    //getWritableDatabase for insert actions...getReadableDatabase for select
(read) actions.
    SQLiteDatabase db = this.getWritableDatabase();

    int userLevel = 1;

    if (admin.isSuperAdmin() == false){
        userLevel = 1;
    }
    else {
        userLevel = 2;
    }

    // User Table:
    // (userName TEXT PRIMARY KEY, userPassword TEXT, userLevel INTEGER,
    // userWeight INTEGER, userHeight INTEGER, userBirthDate TEXT, userGender
    TEXT, profilePic INTEGER)
    ContentValues cv = new ContentValues();
    cv.put("userName", admin.getUserName());
    cv.put("userPassword", admin.getUserPassword());
    cv.put("userLevel", userLevel);
    cv.put("userWeight", admin.getUserWeight());
    cv.put("userHeight", admin.getUserHeight());
    cv.put("userBirthDate", admin.getUserBirthDate());
    cv.put("userGender", admin.getUserGender());
    cv.put("profilePic", admin.getProfilePic());

    // insert - success variable... 1)positive -> it was inserted...2) -1 ->
it was a fail

```

```

long insert = db.insert(TABLE_USER, null, cv);

if(insert == -1){
    return false;
}else{
    return true;
}

//method to update User
public boolean updateUser(User user){

    //getWritableDatabase - method from the default properties of
SQLiteOpenHelper
    //getWritableDatabase for insert actions...getReadableDatabase for select
(read) actions.
    SQLiteDatabase db = this.getWritableDatabase();

    // ContentValues - a special class that works like a associative array
(PHP)
    // can take pairs of values and associate with them (like the bundle in
intent)
    // The ID column is auto increment
    // User Table:
    // (userName TEXT PRIMARY KEY, userPassword TEXT, userLevel INTEGER,
    // userWeight INTEGER, userHeight INTEGER, userBirthDate TEXT, userGender
TEXT, profilePic INTEGER)
    ContentValues cv = new ContentValues();
    cv.put("userName", user.getUserName());
    cv.put("userPassword", user.getUserPassword());
    cv.put("userLevel", user.getUserLevel());
    cv.put("userWeight", user.getUserWeight());
    cv.put("userHeight", user.getUserHeight());
    cv.put("userBirthDate", user.getUserBirthDate());
    cv.put("userGender", user.getUserGender());
    cv.put("profilePic", user.getProfilePic());

    // insert - success variable... 1)positive -> it was inserted...2) -1 ->
it was a fail
    long result = db.update(TABLE_USER, cv, "userName=?", new String
[]{user.getUserName()});

    if(result == -1){
        return false;
    }else{
        return true;
    }
}

//method to update User
public boolean updateUserAsAdmin(Admin user){

    //getWritableDatabase - method from the default properties of
SQLiteOpenHelper
    //getWritableDatabase for insert actions...getReadableDatabase for select
(read) actions.
    SQLiteDatabase db = this.getWritableDatabase();

```

```

int userLevel = 1;

if (user.isSuperAdmin())
    userLevel = 2;

// ContentValues - a special class that works like a associative array
(PHP)
// can take pairs of values and associate with them (like the bundle in
intent)
// The ID column is auto increment
// User Table:
// (userName TEXT PRIMARY KEY, userPassword TEXT, userLevel INTEGER,
// userWeight INTEGER, userHeight INTEGER, userBirthDate TEXT, userGender
TEXT, profilePic INTEGER)
ContentValues cv = new ContentValues();
cv.put("userName", user.getUserName());
cv.put("userPassword", user.getUserPassword());
cv.put("userLevel", userLevel);
cv.put("userWeight", user.getUserWeight());
cv.put("userHeight", user.getUserHeight());
cv.put("userBirthDate", user.getUserBirthDate());
cv.put("userGender", user.getUserGender());
cv.put("profilePic", user.getProfilePic());

// insert - success variable... 1)positive -> it was inserted...2) -1 ->
it was a fail
long result = db.update(TABLE_USER, cv, "userName=?", new String
[] {user.getUserName()});

if(result == -1){
    return false;
} else{
    return true;
}

//method to delete user (by ID because its the primary key)
public void deleteUser(User user) {
    // find user in the database. if its found, delete it and return true.
    // if its not found, return false.

    // getWritableDatabase - we are going to delete from it
    SQLiteDatabase db = this.getWritableDatabase();

    db.delete(TABLE_USER, "userName=?", new String[] {user.getUserName()});
}

//method to get user's level
public int getUserLevelByUserName(String givenUserName){
    ArrayList<User> returnList = new ArrayList<>();

    int userLevel = 0;

    // User Table:
    // (userName TEXT PRIMARY KEY, userPassword TEXT, userLevel INTEGER,

```

```

    // userWeight INTEGER, userHeight INTEGER, userBirthDate TEXT, userGender
    TEXT, profilePic INTEGER)
    // get data from the database
    String queryString = ("SELECT " + "*" + " FROM " + TABLE_USER + " WHERE "
+ "userName" + " = '" + givenUserName + "'");

    // get a reference to the active database
    // getWritableDatabase - insert, update or delete records
    // getReadableDatabase - SELECT items from the database
    SQLiteDatabase db = this.getReadableDatabase();

    // Cursor is the result set from a SQL statement
    Cursor cursor = db.rawQuery(queryString, null);

    // moveToFirst returns a true if there were items selected
    if(cursor.moveToFirst()){

        userLevel = cursor.getInt(2);

    }else{
        // failure. do not add anything to the list.
    }

    // always clean up after yourself
    // Lets close the connection to the database so others can use it
    // close the cursor when done.
    cursor.close();

    return userLevel;
}

//method to get all users
public ArrayList<User> getUsers(){

    ArrayList<User> returnList = new ArrayList<>();

    // get data from the database
    String queryString = "SELECT * FROM " + TABLE_USER;

    // get a reference to the active database
    // getWritableDatabase - insert, update or delete records
    // getReadableDatabase - SELECT items from the database
    SQLiteDatabase db = this.getReadableDatabase();

    // Cursor is the result set from a SQL statement
    Cursor cursor = db.rawQuery(queryString, null);

    // moveToFirst returns a true if there were items selected
    if(cursor.moveToFirst()){

        // Loop through the cursor (result set) and create new customer
        // objects. Put them into the return list
        do{
            // User Table:
            // (userName TEXT PRIMARY KEY, userPassword TEXT, userLevel
            INTEGER,
            // userWeight INTEGER, userHeight INTEGER, userBirthDate TEXT,
            userGender TEXT, profilePic INTEGER)

```

```

String userName = cursor.getString(0);
String userPassword = cursor.getString(1);
int userLevel = cursor.getInt(2);
int userWeight = cursor.getInt(3);
int userHeight = cursor.getInt(4);
String userBirthDate = cursor.getString(5);
String userGender = cursor.getString(6);
String profilePic = cursor.getString(7);

User newUser = new User(userName, userPassword, userWeight,
userHeight, userBirthDate, userGender, profilePic);
returnList.add(newUser);

} while (cursor.moveToNext());

}else{
    // failure. do not add anything to the list.
}

// always clean up after yourself
// Lets close the connection to the database so others can use it
// close the cursor when done.
cursor.close();

return returnList;
}

//method to get all users
public ArrayList<User> getUsersAndAdmins(){

ArrayList<User> returnList = new ArrayList<>();

// get data from the database
String queryString = "SELECT * FROM " + TABLE_USER;

// get a reference to the active database
// getWritableDatabase - insert, update or delete records
// getReadableDatabase - SELECT items from the database
SQLiteDatabase db = this.getReadableDatabase();

// Cursor is the result set from a SQL statement
Cursor cursor = db.rawQuery(queryString, null);

// moveToFirst returns a true if there were items selected
if(cursor.moveToFirst()){

    // Loop through the cursor (result set) and create new customer
    // objects. Put them into the return list
    do{
        // User Table:
        // (userName TEXT PRIMARY KEY, userPassword TEXT, userLevel
        INTEGER,
        // userWeight INTEGER, userHeight INTEGER, userBirthDate TEXT,
        userGender TEXT, profilePic INTEGER)
        String userName = cursor.getString(0);
        String userPassword = cursor.getString(1);
        int userLevel = cursor.getInt(2);
}

```

```

        int userWeight = cursor.getInt(3);
        int userHeight = cursor.getInt(4);
        String userBirthDate = cursor.getString(5);
        String userGender = cursor.getString(6);
        String profilePic = cursor.getString(7);

        //user is Admin at minimum
        if (userLevel == 1 || userLevel == 2){
            //settings boolean of isSuperAdmin
            boolean isSuperAdmin = false;
            if (userLevel == 2)
                isSuperAdmin = true;

            User newUser = new Admin(userName, userPassword, isSuperAdmin,
userWeight, userHeight, userBirthDate, userGender, profilePic);
            returnList.add(newUser);
        }
        //user is normal
        else {
            User newUser = new User(userName, userPassword, userWeight,
userHeight, userBirthDate, userGender, profilePic);
            returnList.add(newUser);
        }

    } while (cursor.moveToNext());

}else{
    // failure. do not add anything to the list.
}

// always clean up after yourself
// lets close the connection to the database so others can use it
// close the cursor when done.
cursor.close();

return returnList;
}

//method to get user with specific name and password
//method to search for user (true - user exists | false - user doesn't exist)
public boolean searchUserByNameAndPass(String userName, String userPassword){
    //get all users from user table
    List<User> Users = this.getUsers();

    // if there are no users
    if(Users.isEmpty() || (userName.isEmpty() && userPassword.isEmpty()) )
        return false;

    for (int i=0; i<Users.size(); i++) {
        User user = Users.get(i);
        if(user.getUserName().contentEquals(userName) &&
user.getUserPassword() .contentEquals(userPassword)){
            return true;
        }
    }
}

```

```

        return false;
    }

    //method to search for user (true - user exists / false - user doesn't exist)
    public boolean searchUserByName(String userName){
        //get all users from user table
        List<User> Users = this.getUsersAndAdmins();

        // if there are no users
        if(Users.isEmpty())
            return false;

        for (int i=0; i<Users.size(); i++) {
            User user = Users.get(i);
            if(user.getUserName().contentEquals(userName)){
                return true;
            }
        }
        return false;
    }

    //method to get for user (true - user exists / false - user doesn't exist)
    public User getUserByName(String userName){
        //get all users from user table
        ArrayList<User> Users = this.getUsersAndAdmins();

        // if there are no users
        if(Users.isEmpty() || userName == null)
            return null;

        for (int i=0; i<Users.size(); i++) {
            User user = Users.get(i);
            if(user.getUserName().contentEquals(userName)){
                return user;
            }
        }
        return null;
    }

    //method to insert Exercise
    public boolean insertExercise(Exercise exercise){

        //getWritableDatabase - method from the default properties of
        SQLiteOpenHelper
        //getWritableDatabase for insert actions...getReadableDatabase for select
        (read) actions.
        SQLiteDatabase db = this.getWritableDatabase();

        // ContentValues - a special class that works like a associative array
        (PHP)
        // can take pairs of values and associate with them (like the bundle in
        intent)
        // The ID column is auto increment
    }
}

```

```

// Exercise Table;
// (exerciseID INTEGER PRIMARY KEY, exerciseName TEXT,
// exercisePic INTEGER, exerciseDetail TEXT)
ContentValues cv = new ContentValues();
cv.put("exerciseID", exercise.getExerciseID());
cv.put("exerciseName", exercise.getExerciseName());
cv.put("exercisePic", exercise.getExercisePic());
cv.put("exerciseDetail", exercise.getExerciseDetail());

// insert - success variable... 1)positive -> it was inserted...2) -1 ->
it was a fail
long insert = db.insert(TABLE_EXERCISES, null, cv);

if(insert == -1){
    return false;
}else{
    return true;
}

}

//method to get all exercises
public ArrayList<Exercise> getExercises(){

    ArrayList<Exercise> returnList = new ArrayList<>();

    // get data from the database
    String queryString = "SELECT * FROM " + TABLE_EXERCISES;

    // get a reference to the active database
    // getWritableDatabase - insert, update or delete records
    // getReadableDatabase - SELECT items from the database
    SQLiteDatabase db = this.getReadableDatabase();

    // Cursor is the result set from a SQL statement
    Cursor cursor = db.rawQuery(queryString, null);

    // moveToFirst returns a true if there were items selected
    if(cursor.moveToFirst()){

        // Loop through the cursor (result set) and create new customer
        // objects. Put them into the return list
        do{
            // Exercise Table;
            // (exerciseID INTEGER PRIMARY KEY, exerciseName TEXT,
            // exercisePic INTEGER, exerciseDetail TEXT)
            int exerciseID = cursor.getInt(0);
            String exerciseName = cursor.getString(1);
            int exercisePic = cursor.getInt(2);
            String exerciseDetail = cursor.getString(3);

            Exercise newExercise = new Exercise(exerciseID, exerciseName,
            exercisePic, exerciseDetail);
            returnList.add(newExercise);

        } while (cursor.moveToNext());
    }else{
}

```

```

        // failure. do not add anything to the list.
    }

    // always clean up after yourself
    // lets close the connection to the database so others can use it
    // close the cursor when done.
    cursor.close();

    return returnList;
}

//method to get all exercises by userNameGiven
public ArrayList<UserExercisesInnerJoinEx> getExercisesByUserNameGiven(String
userNameGiven){

    ArrayList<UserExercisesInnerJoinEx> returnList = new ArrayList<>();

    // get data from the database
    // First table: Exercises
    // Second table: UserExercises
    // Exercise Table;
    // (exerciseID INTEGER PRIMARY KEY, exerciseName TEXT,
    // exercisePic INTEGER, exerciseDetail TEXT)
    //table UserExercises
    //((userExerciseID INTEGER PRIMARY KEY AUTOINCREMENT,
    // userTextColor TEXT, exerciseID INTEGER, date TEXT, time INTEGER,
    // rating INTEGER, repetition INTEGER)
    String queryString = "SELECT Exercises.exerciseID, exerciseName,
exercisePic, exerciseDetail, userExerciseID FROM Exercises INNER JOIN
UserExercises ON Exercises.exerciseID = UserExercises.exerciseID WHERE
UserExercises.userName = '" + userNameGiven + "'";

    //String queryString = "SELECT Exercises.exerciseID,
UserExercises.exerciseID, exerciseName, exercisePic, exerciseDetail FROM Exercises
INNER JOIN UserExercises ON Exercises.exerciseID = UserExercises.exerciseID WHERE
UserExercises.userName = " + userNameGiven;
    // get a reference to the active database
    // getWritableDatabase - insert, update or delete records
    // getReadableDatabase - SELECT items from the database
    SQLiteDatabase db = this.getReadableDatabase();

    // Cursor is the result set from a SQL statement
    Cursor cursor = db.rawQuery(queryString, null);

    // moveToFirst returns a true if there were items selected
    if(cursor.moveToFirst()){

        // Loop through the cursor (result set) and create new customer
        objects. Put them into the return list
        do{
            //UserExercisesInnerJoinEx (int userExerciseID, int exerciseID,
            String exerciseName, int exercisePic, String exerciseDetail)
            int exerciseID = cursor.getInt(0);
            String exerciseName = cursor.getString(1);
            int exercisePic = cursor.getInt(2);
            String exerciseDetail = cursor.getString(3);
            int userExerciseID = cursor.getInt(4);

```

```

UserExercisesInnerJoinEx newUserExercisesInnerJoinEx = new
UserExercisesInnerJoinEx(userExerciseID, exerciseID, exerciseName, exercisePic,
exerciseDetail);
    returnList.add(newUserExercisesInnerJoinEx);

} while (cursor.moveToNext());

}else{
    // failure. do not add anything to the list.
}

// always clean up after yourself
// Lets close the connection to the database so others can use it
// close the cursor when done.
cursor.close();

return returnList;
}

//method to get all exercises by userNameGiven
//that have not been done by the user yet (by knowing that the date of doing
the exercise is null)
public ArrayList<UserExercisesInnerJoinEx>
getExercisesByUserNameGivenNotDone(String userNameGiven){

ArrayList<UserExercisesInnerJoinEx> returnList = new ArrayList<>();

// get data from the database
// First table: Exercises
// Second table: UserExercises
// Exercise Table;
// (exerciseID INTEGER PRIMARY KEY, exerciseName TEXT,
// exercisePic INTEGER, exerciseDetail TEXT)
//table UserExercises
//(userExerciseID INTEGER PRIMARY KEY AUTOINCREMENT,
// userName TEXT, exerciseID INTEGER, date TEXT, time INTEGER,
// rating INTEGER, repetition INTEGER)
String queryString = "SELECT Exercises.exerciseID, exerciseName,
exercisePic, exerciseDetail, userExerciseID FROM Exercises INNER JOIN
UserExercises ON Exercises.exerciseID = UserExercises.exerciseID WHERE
UserExercises.userName = '" +userNameGiven+ "' AND UserExercises.date IS NULL";
//String queryString = "SELECT Exercises.exerciseID,
UserExercises.exerciseID, exerciseName, exercisePic, exerciseDetail FROM Exercises
INNER JOIN UserExercises ON Exercises.exerciseID = UserExercises.exerciseID WHERE
UserExercises.userName = " + userNameGiven;
// get a reference to the active database
// getWritableDatabase - insert, update or delete records
// getReadableDatabase - SELECT items from the database
SQLiteDatabase db = this.getReadableDatabase();

// Cursor is the result set from a SQL statement
Cursor cursor = db.rawQuery(queryString, null);

// moveToFirst returns a true if there were items selected
if(cursor.moveToFirst()){

    // Loop through the cursor (result set) and create new customer
}

```

```

objects. Put them into the return list
    do{
        //UserExercisesInnerJoinEx (int userExerciseID, int exerciseID,
        String exerciseName, int exercisePic, String exerciseDetail)
        int exerciseID = cursor.getInt(0);
        String exerciseName = cursor.getString(1);
        int exercisePic = cursor.getInt(2);
        String exerciseDetail = cursor.getString(3);
        int userExerciseID = cursor.getInt(4);

        UserExercisesInnerJoinEx newUserExercisesInnerJoinEx = new
        UserExercisesInnerJoinEx(userExerciseID, exerciseID, exerciseName, exercisePic,
        exerciseDetail);
        returnList.add(newUserExercisesInnerJoinEx);

    } while (cursor.moveToNext());

}else{
    // failure. do not add anything to the list.
}

// always clean up after yourself
// Lets close the connection to the database so others can use it
// close the cursor when done.
cursor.close();

return returnList;
}

//method to search for exercise by ID
public ArrayList<Exercise> getExercisesByID(int exerciseID){
    //get all users from user table
    ArrayList<Exercise> exercises = this.getExercises();
    ArrayList<Exercise> returnList = new ArrayList<>();

    // if there are no users
    if(exercises.isEmpty())
        return null;

    for (int i=0; i<exercises.size(); i++) {
        Exercise exercise = exercises.get(i);
        if(exercise.getExerciseID() == exerciseID){
            returnList.add(exercise);
        }
    }
}

return returnList;
}

//method to get one exercise by its ID
public Exercise getExerciseByID(int id){

    // get a reference to the active database
    // getWritableDatabase - insert, update or delete records
    // getReadableDatabase - SELECT items from the database
    SQLiteDatabase db = getReadableDatabase();

```

```

// Table: Exercises
//(exerciseID INTEGER PRIMARY KEY, exerciseName TEXT,
// exercisePic INTEGER, exerciseDetail TEXT)
Cursor cursor = db.rawQuery("SELECT * FROM Exercises WHERE exerciseID = "
+ id, null);

cursor.moveToFirst();

int exerciseID = cursor.getInt(0);
String exerciseName = cursor.getString(1);
int exercisePic = cursor.getInt(2);
String exerciseDetail = cursor.getString(3);

Exercise exercise = new Exercise(exerciseID, exerciseName, exercisePic,
exerciseDetail);

// always clean up after yourself
// lets close the connection to the database so others can use it
// close the cursor when done.
cursor.close();

return exercise;
}

//method to insert UserExercise
public boolean insertUserExercise(UserExercise userExercise){

    //getWritableDatabase - method from the default properties of
SQLiteOpenHelper
    //getWritableDatabase for insert actions...getReadableDatabase for select
(read) actions.
    SQLiteDatabase db = this.getWritableDatabase();

    // ContentValues - a special class that works like a associative array
(PHP)
    // can take pairs of values and associate with them (like the bundle in
intent)
    // The ID column is auto increment
    // UserExercises Table
//(userExerciseID INTEGER PRIMARY KEY, userName TEXT,
// exerciseID INTEGER, date TEXT, time INTEGER, rating INTEGER, repetition
INTEGER)
    ContentValues cv = new ContentValues();
    cv.put("userName", userExercise.getUserName());
    cv.put("exerciseID", userExercise.getExerciseID());
    cv.put("date", userExercise.getDate());
    cv.put("time", userExercise.getTime());
    cv.put("rating", userExercise.getRating());
    cv.put("repetition", userExercise.getRepetition());

    // insert - success variable... 1)positive -> it was inserted...2) -1 ->
it was a fail
    long insert = db.insert(TABLE_USEREXERCISES, null, cv);

    if(insert == -1){

```

```

        return false;
    }else{
        return true;
    }
}

//method to delete userExercise
//using it to delete userExercise when in dialog to delete an userExercise
public void deleteUserExercise(String userName, String exerciseName, int
userExerciseID){

    // get a reference to the active database
    // getWritableDatabase - insert, update or delete records
    // getReadableDatabase - SELECT items from the database
    SQLiteDatabase db = this.getWritableDatabase();

    //table: Exercises
    //((exerciseID INTEGER PRIMARY KEY, exerciseName TEXT, exercisePic INTEGER,
    exerciseDetail TEXT)
    Cursor c = db.rawQuery("SELECT * FROM " + TABLE_EXERCISES + " WHERE
    exerciseName = '" + exerciseName + "'", null);

    c.moveToFirst();

    //taking the exerciseID from the selected exercise
    //there is supposed to be only one exercise with the name it has
    int exerciseID = c.getInt(0);
    //Log.d("exerciseID", String.valueOf(exerciseID));
    //table UserExercises
    //((userExerciseID INTEGER PRIMARY KEY AUTOINCREMENT, userName TEXT,
    // exerciseID INTEGER, date TEXT, time INTEGER, rating INTEGER, repetition
    INTEGER))
    String queryString = "DELETE FROM " + TABLE_USEREXERCISES + " WHERE
    userName = '" + userName + "' AND exerciseID = " + exerciseID + " AND userExerciseID =
    " + userExerciseID;

    Log.d("Delete filter", String.valueOf(userExerciseID));
    //Log.d("Delete filter2", String.valueOf(exerciseID));

    db.execSQL(queryString);

}

//method to update userExercise
//((userExerciseID INTEGER PRIMARY KEY AUTOINCREMENT, userName TEXT, exerciseID
INTEGER, date TEXT, time INTEGER, rating INTEGER, repetition INTEGER)
public void updateUserExercise(int userExerciseID, String date, int time, int
rating, int repetition){

    // get a reference to the active database
    // getWritableDatabase - insert, update or delete records
    // getReadableDatabase - SELECT items from the database
    SQLiteDatabase db = this.getWritableDatabase();

    // UserExercises Table
    //((userExerciseID INTEGER PRIMARY KEY, userName TEXT,
    // exerciseID INTEGER, date TEXT, time INTEGER, rating INTEGER, repetition

```

```

    INTEGER)
    ContentValues cv = new ContentValues();
    cv.put("date", date);
    cv.put("time", time);
    cv.put("rating", rating);
    cv.put("repetition", repetition);

    db.update(TABLE_USEREXERCISES, cv, "userExerciseID = " +
userExerciseID,null);

}

//method to get all UserExercises
public ArrayList<UserExercise> getUserExercises(){

    ArrayList<UserExercise> returnList = new ArrayList<>();

    // get data from the database
    String queryString = "SELECT * FROM " + TABLE_USEREXERCISES;

    // get a reference to the active database
    // getWritableDatabase - insert, update or delete records
    // getReadableDatabase - SELECT items from the database
    SQLiteDatabase db = this.getReadableDatabase();

    // Cursor is the result set from a SQL statement
    Cursor cursor = db.rawQuery(queryString, null);

    // moveToFirst returns a true if there were items selected
    if(cursor.moveToFirst()){

        // Loop through the cursor (result set) and create new customer
        objects. Put them into the return list
        do{
            // UserExercises Table
            // (userExerciseID INTEGER PRIMARY KEY AUTOINCREMENT, userName
TEXT,
            // exerciseID INTEGER, date TEXT, time INTEGER, rating INTEGER,
repetition INTEGER)
            int userExerciseID = cursor.getInt(0);
            String userName = cursor.getString(1);
            int exerciseID = cursor.getInt(2);
            String date = cursor.getString(3);
            int time = cursor.getInt(4);
            int rating = cursor.getInt(5);
            int repetition = cursor.getInt(6);

            UserExercise newUserExercise = new UserExercise(userExerciseID,
userName, exerciseID, date, time, rating, repetition);
            returnList.add(newUserExercise);

        } while (cursor.moveToNext());

    }else{
        // failure. do not add anything to the list.
    }

    // always clean up after yourself
}

```

```
// Lets close the connection to the database so others can use it
// close the cursor when done.
cursor.close();

return returnList;

}

//method to search for userExercise by User ID
public ArrayList<UserExercise> getUserExercisesByUserName(String userName){
    //get all users from user table
    ArrayList<UserExercise> userExercises = this.getUserExercises();
    ArrayList<UserExercise> returnList = new ArrayList<>();

    // if there are no users
    if(userExercises.isEmpty())
        return null;

    for (int i=0; i<userExercises.size(); i++) {

        UserExercise userExercise = userExercises.get(i);

        if(userExercise.getUserName() == userName){
            returnList.add(userExercise);
        }
    }

    return returnList;
}

//method to search for userExercise by User ID
//that have been done by the user (by knowing that the date of doing the
//exercise is not null)
public ArrayList<UserExercise> getUserExercisesDoneByUserName(String
userName){
    //get all users from user table
    ArrayList<UserExercise> userExercises = this.getUserExercises();
    ArrayList<UserExercise> returnList = new ArrayList<>();

    // if there are no users
    if(userExercises.isEmpty())
        return null;

    for (int i=0; i<userExercises.size(); i++) {

        UserExercise userExercise = userExercises.get(i);

        if(userExercise.getUserName().contentEquals(userName) &&
userExercise.getDate() != null){
            returnList.add(userExercise);
        }
    }

    if(returnList.isEmpty() == false){
        Log.d("Error","List is empty");
    }
}
```

```

        return returnList;
    }

    //method to search for userExercise by User ID
    //that have not been done by the user yet (by knowing that the date of doing
    the exercise is null)
    public ArrayList<UserExercise> getUserExercisesNotDoneByUserName(String
    userName){
        //get all users from user table
        ArrayList<UserExercise> userExercises = this.getUserExercises();
        ArrayList<UserExercise> returnList = new ArrayList<>();

        // if there are no users
        if(userExercises.isEmpty())
            return null;

        for (int i=0; i<userExercises.size(); i++) {

            UserExercise userExercise = userExercises.get(i);

            if(userExercise.getUserName() == userName && userExercise.getDate() ==
null){
                returnList.add(userExercise);
            }
        }

        return returnList;
    }

    //method to get all songs
    public ArrayList<Song> getSongs() {

        ArrayList<Song> returnList = new ArrayList<>();

        // get data from the database
        String queryString = "SELECT * FROM " + TABLE_SONGS;

        // get a reference to the active database
        // getWritableDatabase - insert, update or delete records
        // getReadableDatabase - SELECT items from the database
        SQLiteDatabase db = this.getReadableDatabase();

        // Cursor is the result set from a SQL statement
        Cursor cursor = db.rawQuery(queryString, null);

        // moveToFirst returns a true if there were items selected
        if (cursor.moveToFirst()) {

            // Loop through the cursor (result set) and create new customer
            objects. Put them into the return list
            do {
                // Song Table
                // (songID INTEGER PRIMARY KEY, songName TEXT, songMP3 INTEGER)
                int songID = cursor.getInt(0);
                String songName = cursor.getString(1);
                int songMP3 = cursor.getInt(2);

```

```
Song newSong = new Song(songID, songName, songMP3);
returnList.add(newSong);

} while (cursor.moveToNext());

} else {
    // failure. do not add anything to the list.
}

// always clean up after yourself
// Lets close the connection to the database so others can use it
// close the cursor when done.
cursor.close();

return returnList;
}

}
```

## דוגמא למתחם :(Adapter)

מתחם של תרגילי משתמש שלא בוצעו ברשימה בסיסי התרגילים של המשתמש  
קוד המחלקה :

```

package com.example.fitnote13022021;

import android.app.Activity;
import android.app.AlertDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.ListAdapter;
import android.widget.TextView;
import android.widget.Toast;

import java.util.ArrayList;

public class UserExerciseNotDoneAdapter extends ArrayAdapter implements ListAdapter {

    private ArrayList<UserExercisesInnerJoinEx> arrayList;
    public static Context context;
    private DataBaseHelper DataBaseHelper;
    private String activeUserName;

    public UserExerciseNotDoneAdapter(ArrayList<UserExercisesInnerJoinEx> arrayList, Context context, String userName) {
        this.arrayList = arrayList;
        this.context = context;
        DataBaseHelper = new DataBaseHelper(context);
        this.activeUserName = userName;
    }
    public UserExerciseNotDoneAdapter(ArrayList<UserExercisesInnerJoinEx> arrayList, Context context) {
        this.arrayList = arrayList;
        this.context = context;
        DataBaseHelper = new DataBaseHelper(context);
    }

    @Override
    public int getCount() {
        return arrayList.size();
    }

    @Override
    public Object getItem(int position) {
        return arrayList.get(position);
    }

    @Override
    public long getItemId(int position) {

```

```

        return 0;
    }

    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        UserExercisesInnerJoinEx userExercisesInnerJoinEx =
arrayList.get(position);
        convertView =
LayoutInflater.from(context).inflate(R.layout.user_exercise_layout, null);

        TextView txtMainTitle = convertView.findViewById(R.id.txtMainTitle);
        ImageView imageView = convertView.findViewById(R.id.imageView);
        Button btnDelete = convertView.findViewById(R.id.btnDelete);

        txtMainTitle.setText(userExercisesInnerJoinEx.getExerciseName());
        imageView.setImageResource(userExercisesInnerJoinEx.getExercisePic());

        btnDelete.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Toast.makeText(context,
txtMainTitle.getText().toString(),Toast.LENGTH_SHORT).show();

                //creating the dialog to delete the userExercise
                AlertDialog.Builder builder = new AlertDialog.Builder(context);

                builder.setTitle("delete exercise");

                builder.setMessage("Do you really want to delete?");

                //delete
                builder.setPositiveButton("Yes", new
DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {

dataBaseHelper.deleteUserExercise(activeUserName,userExercisesInnerJoinEx.getExerciseName(), userExercisesInnerJoinEx.getUserExerciseID());
                        arrayList.remove(position);
                        notifyDataSetChanged();
                    }
                });

                //cancel
                builder.setNegativeButton("No", new
DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {
                        dialog.cancel();
                    }
                });
            }

            //Creating and showing the dialog
            AlertDialog dialog = builder.create();

            dialog.show();
        });
    }
}

```

```
        }

    });

    imageView.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Toast.makeText(context,
userExercisesInnerJoinEx.getExerciseName(), Toast.LENGTH_SHORT).show();

            Intent intent1 = new Intent(context,
ExecuteExerciseActivity.class);

            intent1.putExtra("ExecuteUserExerciseID",
userExercisesInnerJoinEx.getUserExerciseID());
            intent1.putExtra("ExecuteExerciseID",
userExercisesInnerJoinEx.getExerciseID());
            intent1.putExtra("activeUserName",activeUserName);
            context.startActivity(intent1);
            ((Activity)context).finish();
        }
    });

    txtMainTitle.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Toast.makeText(context,
userExercisesInnerJoinEx.getExerciseName(), Toast.LENGTH_SHORT).show();

            Intent intent1 = new Intent(context,
ExecuteExerciseActivity.class);

            intent1.putExtra("ExecuteUserExerciseID",
userExercisesInnerJoinEx.getUserExerciseID());
            intent1.putExtra("ExecuteExerciseID",
userExercisesInnerJoinEx.getExerciseID());
            intent1.putExtra("activeUserName",activeUserName);
            context.startActivity(intent1);
            ((Activity)context).finish();
        }
    });

    return convertView;
}
}
```

## מחלקה עזר לשימירה ולקיחת קבצי תמונות (PictureFileHelper):

מחלקה עזר לשימירת תמונות בקובץ info ומציאת תמונה בעזרת שם המשתמש.  
קוד המחלקה :

```
package com.example.fitnote13022021;

import android.app.Activity;
import android.content.Context;
import android.content.SharedPreferences;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.util.ArrayList;

public class PictureFileHelper {

    public static void writeFileToInternalStorage(Context context, Bitmap bitmap,
String filename)
    {
        SharedPreferences sp= context.getSharedPreferences("info",0);
        int counter=sp.getInt("counter", 0);
        try {
            FileOutputStream os =
((Activity)context).openFileOutput(filename+counter, Context.MODE_PRIVATE);
            //Here compress 50%, store the compressed data in os.
            bitmap.compress(Bitmap.CompressFormat.PNG,100,os);
            counter++;
            SharedPreferences.Editor editor=sp.edit();
            editor.putInt("counter",counter);
            editor.commit();
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
    }

    public static Bitmap readFileFromInternalStorage(Context context,String
filename)
    {
        Bitmap b=null;
        try {
            InputStream in = ((Activity)context).openFileInput(filename);
            b= BitmapFactory.decodeStream(in);
            in.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return b;
    }

    public static Bitmap getPic(Context context, String name)
    {
        File mydir = context.getFilesDir();
```

```
File lister = mydir.getAbsoluteFile();
Bitmap bitmap=null;

for (String list : lister.list())
{
    if(list.toString().contains(name)) {
        //Toast.makeText(context, list, Toast.LENGTH_LONG).show();
        bitmap = readFileFromInternalStorage(context, list);

    }
}

return bitmap;
}
```

## מחלקה להשמעת התראה בעזרת BroadcastReceiver : (ReminderBroadcast)

המחלקה בונה התראה ברגע שהיא מקבלת הodata מה-AlarmManager שלו יש זמן מוגדר מראש במסך התרגילים של המתמש (ProgramUserActivity).

קוד המחלקה :

```
package com.example.fitnote13022021;

import android.app.PendingIntent;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.util.Log;

import androidx.core.app.NotificationCompat;
import androidx.core.app.NotificationManagerCompat;

import java.util.concurrent.atomic.AtomicInteger;

public class ReminderBroadcast extends BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent) {

        Log.d("RECEIVED", "Notification received!" + " UserName" +
        intent.getStringExtra("activeUserName"));

        // Get id & message from intent.
        String message = intent.getStringExtra("message");
        String activeUserName = intent.getStringExtra("activeUserName");

        // Call ProgramUserActivity when notification is tapped.
        Intent programUserAcIntent = new Intent(context,
        ProgramUserActivity.class);

        //we need to add this userName in order to take the user to his
        //ProgramUserActivity screen
        programUserAcIntent.putExtra("activeUserName", activeUserName);

        //Intent to go to user's exercise List when clicking on notification
        PendingIntent contentIntent = PendingIntent.getActivity(
            context, 0, programUserAcIntent, 0
        );

        NotificationCompat.Builder builder = new
        NotificationCompat.Builder(context, "notifyLemubit")
            .setSmallIcon(android.R.drawable.ic_dialog_info)
            .setContentTitle("Exercise Time!")
            .setContentText(message)
            .setContentIntent(contentIntent)
            .setAutoCancel(true)
            .setPriority(NotificationCompat.PRIORITY_DEFAULT);

        NotificationManagerCompat notificationManager =
        NotificationManagerCompat.from(context);

        //According to the official documentation
```

```
//starting in Android 8.0(API Level 26)
//all notifications must be assigned to a channel

//the notificationID is a unique int for each notification that must be
defined
    notificationManager.notify(NotificationID.getID(), builder.build());

}

public static class NotificationID {
    private final static AtomicInteger c = new AtomicInteger(0);
    public static int getID() {
        return c.incrementAndGet();
    }
}
```

## מחלקה להשמעת מוזיקה והפעלת טיימר בעזרת סרוויס CountDownTimer ו-MediaPlayer ,Service (MusicAndTimerService)

המחלקה בונה שירותים שקוראים לה פעולה onCreate(). כאשר מפעילים פקודה לשירות נשלח ל פעולה onStartCommand(intent, flags, startId). מטרת המבירה היא פקודה על השירות לבצע PLAY-לנגן ולהפעיל את הטיימר או PAUSE-להשנות את שיניהם). הפעולה onDestroy() נקראת כאשר כושרים את השירות והפעלה עוצרת את הטיימר והמוזיקה ומפסת את המונח של הזמן.

קוד המחלקה:

```
package com.example.fitnote13022021;

import android.app.Service;
import android.content.Intent;
import android.media.MediaPlayer;
import android.os.CountDownTimer;
import android.os.IBinder;
import android.widget.Toast;

import androidx.annotation.Nullable;

import java.util.ArrayList;
import java.util.Random;

public class MusicAndTimerService extends Service {

    private DataBaseHelper DataBaseHelper;
    private MediaPlayer mediaPlayer;
    private CountDownTimer countDownTimer;
    public static int count = 0;

    //this method does binds the service
    //with and activity
    //הפעילות עם השירות את קוشرת
    @Nullable
    @Override
    public IBinder onBind(Intent intent) {
        return null;
    }

    @Override
    public void onCreate() {
        super.onCreate();
        DataBaseHelper = new DataBaseHelper(this);
        ArrayList<Song> songs = DataBaseHelper.getSongs();

        int arrayLength = songs.size();

        Random random = new Random();
        //This gives a random integer number between 0 (inclusive) and (Length of
        array-1)
        int randomNumber = random.nextInt(arrayLength);
        Song chosenSong = songs.get(randomNumber);

        Toast.makeText(this, "chosen song: " + chosenSong.getSongName() + "N: " +

```

```

randomNumber, Toast.LENGTH_SHORT).show();

//create the mediaPlayer
mediaPlayer = MediaPlayer.create(this, chosenSong.getSongMP3());

mediaPlayer.setLooping(true);

if(countDownTimer != null){
    countDownTimer.cancel();
    countDownTimer = null;
}

//setting up the timer
//max: 2147483347millis = 35791.389116667minutes = 596.523151944449978
hours, sec to change: 1000
countDownTimer = new CountDownTimer(Integer.MAX_VALUE - 300, 1000) {
    @Override
    public void onTick(long millisUntilFinished) {
        count++;
        //setting text in play to be time count from timer in
MusicAndTimerService
        ExecuteExerciseActivity.txtTime.setText(getDurationString(count));
    }

    @Override
    public void onFinish() {
        count = 0;
        ExecuteExerciseActivity.txtTime.setText(getDurationString(0));
    }
};

}

//when we start the service the onStartCommand method will be called
@Override
public int onStartCommand(Intent intent, int flags, int startId) {

    String action = intent.getStringExtra("ACTION");
    if(action.contentEquals("PLAY")){
        Toast.makeText(this, "PLAY", Toast.LENGTH_SHORT).show();
        //play music and timer
        mediaPlayer.start();
        countDownTimer.start();
    }else if (action.contentEquals("PAUSE")){
        Toast.makeText(this, "PAUSE", Toast.LENGTH_SHORT).show();
        //pause music and timer
        mediaPlayer.pause();
        countDownTimer.cancel();
    }

    // this means this service will be explicitly started and stopped
    //במפורש ויפסיק יונען זה שהשירות הידבר פירומ
    return START_STICKY;
}

}

//when the service is stopped onDestroy method will be called
@Override

```

```
public void onDestroy() {  
    super.onDestroy();  
  
    //stop music and timer  
    mediaPlayer.stop();  
    countDownTimer.cancel();  
    count = 0;  
    ExecuteExerciseActivity.txtTime.setText(getDurationString(0));  
}  
  
private String getDurationString(int seconds) {  
  
    int hours = seconds / 3600;  
    int minutes = (seconds % 3600) / 60;  
    seconds = seconds % 60;  
  
    return twoDigitString(hours) + " : " + twoDigitString(minutes) + " : " +  
twoDigitString(seconds);  
}  
  
private String twoDigitString(int number) {  
  
    if (number == 0) {  
        return "00";  
    }  
  
    if (number / 10 == 0) {  
        return "0" + number;  
    }  
  
    return String.valueOf(number);  
}  
}
```

## מסמכים נוספים

כל מסמך שכתבתי, או מצאתי במקור מידע כל שהוא שעזר לי:

- **בגיבוש הרעיון.**
- **בפיתוח התוכנה.**
- **בתיבת הקוד.**
- **בבדיקות התקינות.**

לפני כתיבת הקוד ועיצוב ה-XML-ים עיצבתי סקיצה של תיק עבודה לאפליקציה.

זאת בהשראת מאפליקציות קיימות בשוק.

דף הבא מצורף תיק העבודה הראשון שלי שנכתב בתחילת שנת הלימודים הנוכחית:

## תיק עבודה: FitNote

תאריך: 25.01.2021

מגיש: אלון בן דוד

מורים: ניצן

המטרה היא לכתוב אפליקציה להכנת אימון כושרiesel וŁמיעקב אחריו האימון.

המשתמש מכניס לאפליקציה בעט רישום ראשון את שמו, הסיסמה שלו, תאריך לידתו, מינו ומשקלו. לאחר מכן, נתונים המשמשים נכנסים למאגר הנתונים של התוכנה והוא רשאי להיכנס.

האפליקציה משמשת ככלי לארגון תכניות תרגילים לשימוש.

יהיו תכניות הנמצאים בתוך שדה הנתונים שהתוכנה מציע לשימוש. אך הוא רשאי לבנות תכנית אישית עם שם מסוים, תיאור ואוסף של תרגילים. התרגילים באפליקציה מסויכים לחلكי גוף מסוימים כך שניתן לבנות אימון כוחiesel ומקיף על מספר איברים בגוף מבלי להבין לעומק את אנטומיות הגוף האדם.

המשתמש יכול לבצע את התרגילים באימון בכל סדר שירצה. סדר ביצוע התרגילים באימון דינמי כדי למנוע הגבלה על המתאמן. למשל, אם המתאמן נמצא בחדר כoshר וההילICON תפוס ע"י מתאמן אחר, הוא יוכל

לבחור תרגיל אחר באימון שלא דורש שימוש במתќון זה.

האפליקציה משמשת ככלי מעקב למataן אחריו הביצועים שלו בכל התרגילים.

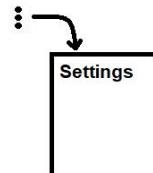
המשתמש יכול להבין את מכובו בחישוב תכניות הקודמות ולהבחן במוגמות של שיפור/האטה באופן ברור. זאת דרך גרפים, עמודות ופרמטרים הקשורים לתרגיל.

האפליקציה מציעה לשימוש בצע תרגילי כוח בהדגש על אזורים מסוימים בגוף.

יהיו המלצות לכמות הביצועים בתרגילים אך המשמש יכול להכניס את הנסיבות שעשה בפועל בזמן שלו. מספר החזרות של כל תרגיל מצוין בהמלצת האפליקציה.

המשתמש מאשר בלחיצת כפתור על ביצוע המטלה. התוכנה עוקבת אחרי הדיווחים של המתאמן וזוכרת את ביצועיו.

לאחר שימוש מסוים באפליקציה התוכנה תספק לשימוש עידודים המבוססים על הסטטיסטיקה שלו. אם ניכר שיפור בביצוע המתאמן בתרגיל מסוים, יינתן לו עידוד ורמת התרגילים עולה. אולם, במידה שהמתאמן מתקשה ביצוע התרגיל וביצועו חלקי, התוכנה תדע להקל עליו ובהדרגה תנסה לסייע לו להשתפר שוב.

**מסכים:****מסכים ראשונים – רישום ראשון ומסך כניסה:****מסך הפתיחה:**

מסך זה דורש שם משתמש וסיסמה של משתמש קיימים במאגר הנתונים כדי להיכנס לתוכנה. השלוש נקודות לעליה מובילות לתוך מסך קטן קטן שם יש אפשרות ל採取 למסך הנגדות המשתמש. למקרה יש שני כפתורים: כפתור רישום וכפתור רישום משתמש חדש. כפתור הרישום למשתמש החדש פותח תחת מסך בו הוא יידרש להכניס את כל הפרטים הבאים: שם משתמש, סיסמא, תאריך לידיה, מגוון, משקל. כפתור הרישום דורש שם משתמש וסיסמא כדי להיכנס לאפליקציה.

**מסך הרישום:**

במסך הרישום המשתמש בעת רישום ראשון מכניס את שמו, הסיסמא שלו, תאריך לידתו, מגוון ומשקלו.

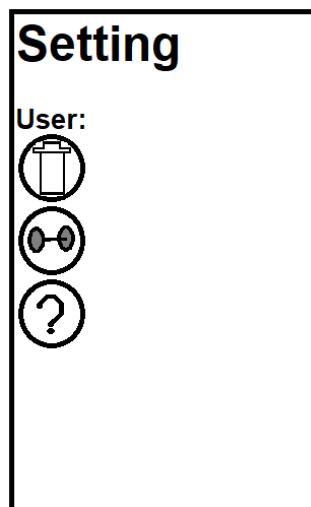
**מסך הגדרות המשתמש:**

במסך זה יש שלושה כפתורים.

כפתור הזבל מוחק את המשתמש הרשום עתה.

כפתור המשקלות נותן אפשרות לשינוי המשקל של המשתמש.

כפתור סימן השאלה נותן עזרה למשתמש באפליקציה. יש פרטיהם על האפליקציה ועל הנתונים שהיא לוחחת המשתמש במהלך תכניות התרגילים שלו.



ברגע שהמשתמש הכנס את פרטיו לאפליקציה הוא רשאי להיכנס עם שם משתמש וסיסמה לתכנית התרגילים שלו.

בהתחלת מופיעים תכניות שהתוכנה מציעה למשתמש להתנסות בהם. המשתמש יכול להכין לעצמו תכניות תרגילים המוצבבים בהdagש לשרירים אוטם הוא רוצה לחזק בתרגילים.

#### מסך ראשי – מסך תכניות האימון:

במסך הראשי מופיעים כל התרגילים במאגר הנתונים של האפליקציה. יש תרגילים שנמצאים בתוכנה עצמה מראש כמו אלו שמופיעים בשרטוט.

### Trainings:

ARM BEGINNER



⋮

LEG BEGINNER



⋮

CARDIO BEGINNER



⋮



Edit

Delete

Info

למעלה מופיעה כוורת סטטית עם השם "תכניות:".

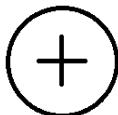
כפתור הפלוס למטה מימין לוקח את המשתמש למסך בו הוא יבנה את האימון בהתאם לשרירים שהוא רוצה לחזק.

כפottaר העמודות למטה משמאלי לוקח את המשתמש למסך בו יש סיכום של ניתוח המידע והסטטיסטיקה על כל התרגילים שהמשתמש עשה.

לחיצה על המלבן בו מופיע האימון לוקח את המשתמש למסך בו הוא רואה רשימה של כל התרגילים באימון והוא יכול גם לעשות את האימון ולראות נתוני על ביצועו האימוני.

לחיצה על שלוש הנקודות ליד כל מלבן של אימון נותן למשתמש שלושה אופציות: שינוי האימון, מחיקה וטיואר.

מיני מסך שמופיע באט לחיצה על שלוש הנקודות:



**Custom Training:**

Name: \_\_\_\_\_

Description: \_\_\_\_\_

**Exercises:**

ARM:  
 PushUp  
 Lifting

CARDIO:  
 ...  
 MIX:  
 ...

**FINISH**

**מסך חוספת אימון :**

מסך זה מופיע אחרי לחיצה על כפתור הפלוס במסך הראשי בו מופיעים כל התוכניות. למעלה מופיעות כותרת סטטיסטית עם השם "אימון מותאם אישית :". מתחת לכותרת מופיעים שני מקומות להכנסת טקסט שבאחד מופיע שם האימון ובשני תיאור של האימון.

לחיצה על כפתור התמונה נותנת אפשרות לשוחץ תמונה לאימון המותאם אישית. אפשר לא להכניס תמונה ואז האפליקציה תכניס למאגר הנתונים תמונה בירית Machol.

מתחת לכותרת "תרגילים": מופיע רשימה של כל התרגילים שיש במאגר הנתונים. כל התרגילים משוויכים לתת-כותרות שמתארות את חלק הגוף מהם העיקרי. עיקר מזקקים בעת האימון.

ליד כל תרגיל מופיע עיגול שלחיצה עליו מאשרת שהוא נכנס לאימון. אחרי לחיצה על העיגול מופיע ווי בתו אישור על הלחיצה.

במסך שמאל למעלה מופיע כפתור חוזה למסך תכניות התרגילים. לחיצה עליו מבטלת את תהליך הכתת האימון.

למטה מימין מופיע כפתור סיום "Finish" שלחיצה עליו מסיימת את תהליך הכתת האימון האישי ומכונישה את האימון לזכרון התוכנה. האימון החדש מופיע במסך הראשי בו מופיעים כל התכניות, ולשם יוחזר המשטמש ישר אחרי הלחיצה על כפתור הסיום.

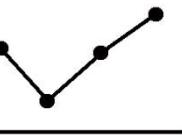
לחיצה על כפתור הסיום של הכנסת התוצאות תתאפשר אך ורק אחרי שככל הפרטים לעיל הוכנסו. לחיצה כל כפתור הסיום תיקח את המשטמש למסך דיווח הנתונים אחרי ביצוע התרגיל.



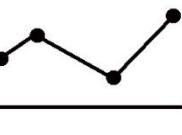
**Report:**

ARM BEGINNER:

LEG BEGINNER:



CARDIO:



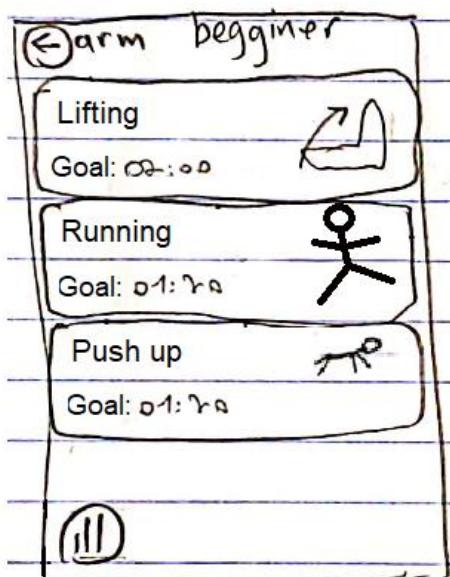
**מסך סטטיסטיות של כל התרגילים בתכניות :**

מסך זה מופיע בלחיצה על כפתור העמודות במסך הראשי בו מופיעים כל התכניות.

מסך זה מציג סיכום של ניתוח כל המידע על התכניות והתרגילים שביצעו משתמש בהם.

אפשר לראות מותחת לכל אחת את כל המדדי הסטטיסטי הקשור לאימון הזוג והתרגילים שבו.

במסך שמאל למעלה מופיע כפתור חוזה למסך תכניות האימון.

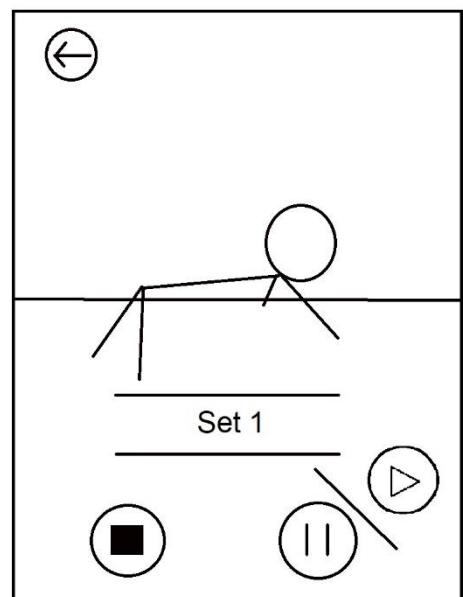


#### מסך התרגילים של האימון הנבחר במסך הראשי:

המשך זהה מופיע בלחיצה על אימון מסוים במסך תכניות האימון. במסך שמאל לעילו מופיע כפתור חוזה למסך תכניות האימון. שם האימון מופיע בគורתת לעילו. מאחורי הគורתת מופיע התמונה של האימון. ברשימה באמצע המשך מופיעים כל התרגילים שייכים לאותו אימון. בכל תרגיל מופיע שמו, תמונה שמתארת אותו והזמן המומלץ לביצוע התרגיל. סדר ביצוע התרגילים אינו משנה. משתמש חופשי לבצע את התרגילים באיזה סדר שהוא רוצה. סדר על תרגיל מסוים תיקח את המשתמש למסך בו הוא יתחל את תחילך ביצוע התרגיל. הכפטור למטה שמאל עם העמודות ייקח את המשתמש למסך בו מופיע סיכום של כל ניתוח המידע והסטטיסטיקה על האימון הנבחר ספציפית.

#### מסך ביצוע התרגיל:

מסך זה מופיע אחריו לחיצה על תרגיל במסך התרגילים של האימון. במסך שמאל לעילו מופיע כפתור חוזה למסך הקודם. לחיצה עליו מבטלת את עשיית התרגיל ומחזירה את המשתמש חוזה למסך התרגילים של האימון. במסך מופיע תמונה של התרגיל בשישיה. כאשר לוחצים על כפתור המשולש השוכב האנימציה פועלת והטיימר מתחילה. כשהאנימציה מתחילה סימן שאפשר להתחיל את התרגיל. למטה מופיע כוורתת עם מספר הסט שעושים.



הכפטור מימין למטה הוא כפתור אחד שלפני הначלה התרגיל הוא משולש שוכב, כפתור הפעלה, וברגע שלוחצים עליו התרגיל מתחילה – התמונה הגדולה הופכת לanimציה והמשתמש יכול להתחיל את התרגיל. ברגע שהכפטור נלחץ הוא הופך לשני קוים מקבילים – כפתור ההשניה. כSCPATOR ההשניה נלחץ הטימר עוצר. ההשניה המתבצעת ע"י לחיצה על כפתור ההשניה עוצרת את הטימר עד SCPATOR הפעלה נלחץ שוב. תחילת התרגיל והפעלתו ממשמע הפעלת טימר בתוך האפליקציה עצמה. זמן הביצוע הוא אחד הפרמטרים בו נעזר בעת ניתוח ביצוע התרגיל של המתאמן. הזמן אותו נתה הוא הזמן שהטימר מראה בפועל. הטימר מפסיק כשSCPATOR הריבוע השחור נלחץ.

לחיצה על הכפטור שמאל למטה, כפתור הריבוע השחור, עוצרת את ביצוע התרגיל. בלחיצה זו הטימר עוצר. אחורי לחיצה על כפתור הריבוע השחור ימשיך המשתמש לעשות את שאר הסטים של התרגיל. ברגע שישים המשתמש את הסט האחרון הוא ימשיך למסך הכנסת תוכנות ושביאות רצון. התוכנה מקבלת את הזמן שהתקיים למתאם לביצוע התרגיל. לכן

במסך הבא הוא יכנס את מספר הביצועים שהוא הספיק באותו זמן ואת רמת הקושי של התרגיל. לחיצה על הריבוע השחור מסיימת את ביצוע התרגיל. שהמשתמש לווחץ על הכפטור הזה הוא מאשר את סיום ביצוע הסט הזה של התרגיל. הטימר באותו הזמן גם עוצר והזמן שהטימר מראה הוא זמן ביצוע התרגיל בפועל. הזמן הזה ישמר בסיס הנתונים וישמש לתוכנה פרמטר לניתוח מצבו של המתאמן. ביצוע התרגיל מסתיים אחורי שהמתאמן עבר על כל הסטים שיש לאותו התרגיל.

נתון הזמן שהלקח לביצוע כל סט ישמיר בבסיס הנתונים. השוואת בין הזמנים בימים שונים תעזר לתוכנית קבועה אם יש מגמה של שיפור/האטה אצל המתאמן. התוכנה עצמה תעשה שינויים בהתאם למגמה. אם המשמש השתף בביוץ התרגילים וזמן הביצוע שלו קצר התוכנה תקטין את הזמן המומלץ לביצוע התרגילים. לעומת זאת, אם המשמש מתקשה בביוץ התרגילים וזמן הביצוע שלו גדול אז התוכנה תגדיל את הזמן המומלץ לביצוע התרגילים.

מסך הכנסת תוצאות לתוכנית ושביאות רצון :

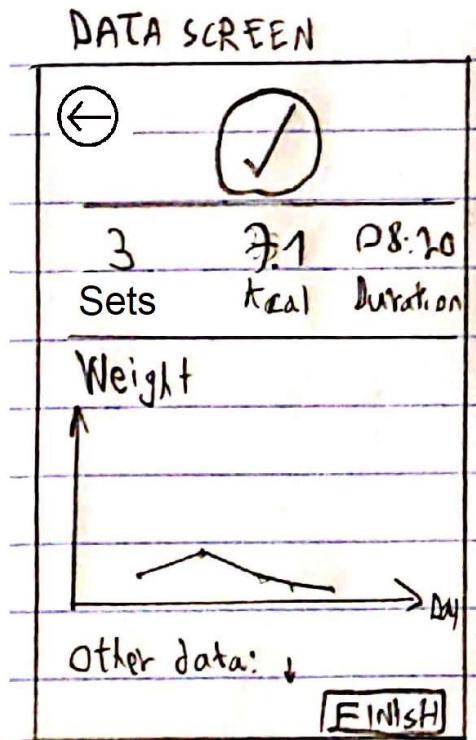
מסך זה מופיע לאחר סיום ביוץ התרגילים. המסך יבקש מהמתאמן להכניס בגלגולות את מספר הביצועים שביצע בכל סט בתרגיל. מספר הסטים תלוי בתרגיל שבוצע.

לחיצה על חץ החזרה יחויר את המשמש למסך התרגילים ויבטל את תהליך עשיית התרגיל. אחרי זה יש בקשה לדרג את הקושי שהיה בביוץ התרגילים מאחת ועד חמישה. כאשר אחת זה אומר שהቢוץ היה קשה מאד. בהדרגה הקושי של התרגיל יורד עד שיחמש זה אומר שהቢוץ היה קל למתאמן. לחיצה על כפתור הסיום של הכנסת התוצאות תאפשר לך אך ורק אחרי שכל הפרטים לעיל הוכנסו. לחיצה כל כפתור הסיום תיקח את המשמש למסך דיווח הנתונים אחורי ביוץ התרגיל.

דירוג הקושי של התרגיל, מספר הביצועים של המתאמן בכל סט והזמן שהלקח לפועל למתאמן לבצע את התרגיל הם הפרמטרים שייעזרו לנתח את מצבו של המתאמן. התוכנה תוכל לדעת האם להקשות או להקל על המתאמן באותו תרגיל.

The screenshot shows a results page with the following details:

- Results** (Section title)
- How much did you do in each set?**
- Set 1:** 25 (with a downward arrow icon)
- Set 2:** 25 (with a downward arrow icon)
- Set 3:** 25 (with a downward arrow icon)
- Rate the difficulty:(1 ->5)**
- A scale from 1 to 5 with the following marks: X, O, O, O, O, ✓
- FINISH** (Large button at the bottom)



מסך דיווח הנתונים אחורי ביצוע תרגיל:  
מסך זה מופיע אחורי ביצוע מלא של התרגיל והכנסת תוצאות ביצוע התרגיל.

למעלה מופיע ווי בתור אישור על סיום ביצוע התרגיל.  
באמצע מופיעים: מספר הסטים שביצע המשמש, מספר הקלוריות שירדו בעת ביצוע התרגיל והזמן הכלול לשחק לבצע את התרגיל.  
 מתחת לנtones אלה מופיעים גרפים המספקים ניתוח ויזואלי על מצבו של המתאמן באותו התרגיל שביצע כרגע ביחס לפעם קודמת שביצע את התרגיל.

לחיצה על כפתור הסיום למיטה מימין או על החץ מימין לעלה מחזירה את המתאמן למסך התרגילים בו מופיעים כל התרגילים של האימון.

#### תפקיך בסיס הנתונים במסכים:

ב חלק זה אני אסביר מה תפקידו של בסיס הנתונים בשירות המשתמש עברו כל מסך באפליקציה.  
קודם לנתח מהגרת הטבלאות שייהו בבסיס הנתונים.

שם הטבלה : User – טבלת המשתמשים :  
מטרה : שבירת פרטי אישיים של המשתמש (שדה=עמודה בטבלה).

הערות	סוג הנתונים	שם שדה באנגלית	שם שדה בעברית
PRIMARY KEY	TEXT	username	שם משתמש
	TEXT	password	סיסמא
	FLOAT	weight	משקל
	TEXT	birthyear	שנת לידה
	TEXT	gender	מין

שם הטבלה : BodyPart – טבלת איברי הגוף :  
מטרה : שבירת כל סוגי האיברים בגוף עליהם התרגילים עובדים.

הערות	סוג הנתונים	שם שדה באנגלית	שם שדה בעברית
PRIMARY KEY	INT	bodyPartCode	קוד האיבר
	TEXT	name	שם האיבר
	INT	bodyPartPic	תמונה של האיבר

שם הטבלה : Exercise – טבלת התרגילים :  
מטרה : שבירת כל סוגי התרגילים לאיימון הגוף.

הערות	סוג הנתונים	שם שדה באנגלית	שם שדה בעברית
PRIMARY KEY	INT	exerciseCode	קוד התרגיל
	TEXT	name	שם התרגיל
	INT	sets	סטים

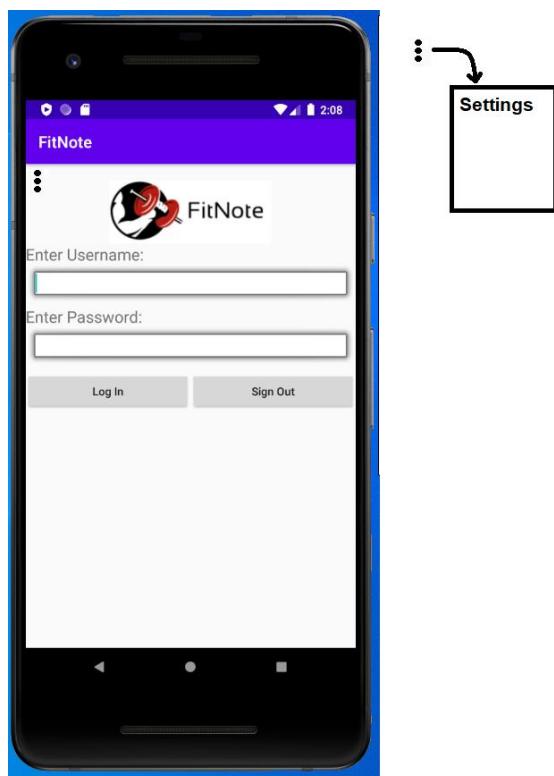
	INT	amount	מספר הביצועים
	TEXT	duration	זמן עשייה של תרגיל
	TEXT	difficulty	רמת הקושי
	TEXT	exerciseDate	תאריך ביצוע התרגיל
	INT	codeBodyPart	קוד איבר בגוף
	INT	exercisePic	תמונה של התרגיל
	TEXT	detail	תיאור ביצוע התרגיל
	TEXT	recommendation	המלצת

שם הטבלה : Program – טבלת תכנית התרגילים :  
מטרה : שימירת כל התכניות. אל כל תכנית משוכחים תרגילים מסוימים.

הערות	סוג הנתונים	שם שדה באנגלית	שם שדה בעברית
	TEXT	userName	שם המשתמש אליו האימון שידך
	TEXT	programName	שם האימון
	INT	programPic	תמונה של האימון
	INT	exerciseCode	קוד תרגיל
	TEXT	programDate	תאריך ביצוע האימון
	TEXT	description	מידע על האימון

#### מסך הפתיחה:

בכניסה ראשונה המשתמש חייב להכנס למסך הרישום דרך הכפתור מימין למיטה.  
אחריו שהמשתמש הלך למסך הרישום הראשוני והמידע עליו רשום בסיס הנתונים הוא יוכל להמשיך למסך תכניות האימון.





**מסך הרישום :**  
במסך הרישום המשתמש מכניס את שם המשתמש, הסיסמה, משקלו, תאריך לידה ומיןו.

כל המידע שמכנס עובר לרשותה של משתמש חדש שנשלח לטבלת המשתמשים כמשתמש חדש.

עם המשתמש החדש הרישום יוכל המתאם לחזור למסך הפתיחה ולהיכנס עם שם משתמש וסיסמה.

#### **מסך ראשי – מסך תכניות האימון :**

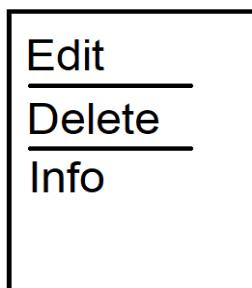
כל הרשומות של התכניות המשויכים לאותו המשתמש שנרשם כרגע מוצאים מבסיס הנתונים. במסך הראשי מופיעים ברשימה אחת נתונים מרשומות של טבלת התכניות שהוצעו כרגע.

כל אימון ברשימה המוצגת מופיע עם שם האימון ותמונה של האימון מהטבלה של התכניות.

שם אימון נלקח מהשדה "שם האימון" והתמונה של האימון מובאת מהשדה "תמונה של האימון".

לחיצה על שלוש הנקודות ליד כל אימון מופיע המסך הזה :  
לחיצה על כפטור העריכה נותן למשתמש לעורך את האימון הנבחר. העריכה המבוצעת עורכת את האימון הנבחר בטבלת התכניות.

לחיצה כל כפטור המחקה מוחק את האימון ואת הנתונים המשויכים לו מבסיס הנתונים.



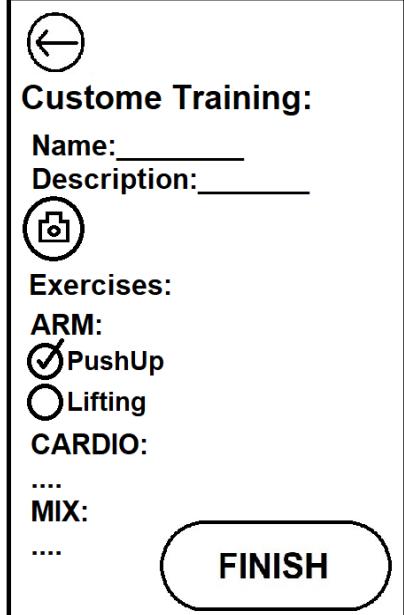
לחיצה על כפטור המודיע נותן את המידע על האימון משדה "מידע על התכנית" של תכנית זה מטבלת התכניות.

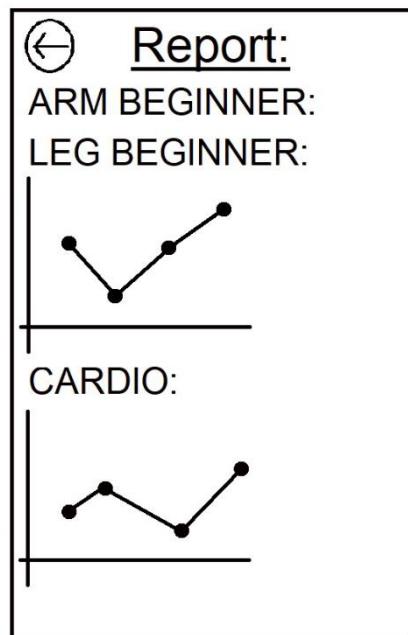
#### **מסך הוספה האימנו :**

מסך זה מופיע אחרי לחיצה על כפטור הפלס במסך הראשי בו מופיעים כל התכניות. במסך זה מתחילה תהליך של הוספה אימון מותאם אישית.

המשתמש מכניס במסך זה את הפרטים על האימון המותאם אישית שלו הכוללים : שם האימון, תיאור האימון, תמונה לאימון (לא חייב), יוכנס תמונה ברירת מחדל אם לא ישימו תמונה אחרת) ובחרית התרגילים המשויכים לאימון מטבלת התרגילים.

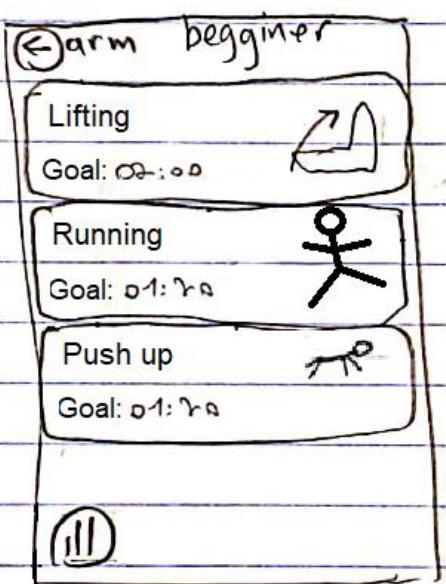
לחיצה על כפטור הסיום מכינה רשומה של אימון חדש המועברת לטבלת התכניות הכוללת את השם והתיאור והתמונה (אם שמו) של האימון המותאם אישית, וכל התרגילים ששסמננו בוויי יכנסו לרשותה של האימון החדש.





מסך הסטטיסטיקות של כל התכניות והתרגולים שלהם:  
מסך זה מופיע בלחיצה על כפתור העמודות במסך הראשי בו מופיעים כל התכניות.

מסך זה פורש בקטגוריות את הנתונים האישיים של המשתמש עבור כל התכניות שביצע והביצועים בכל התרגולים בהם.



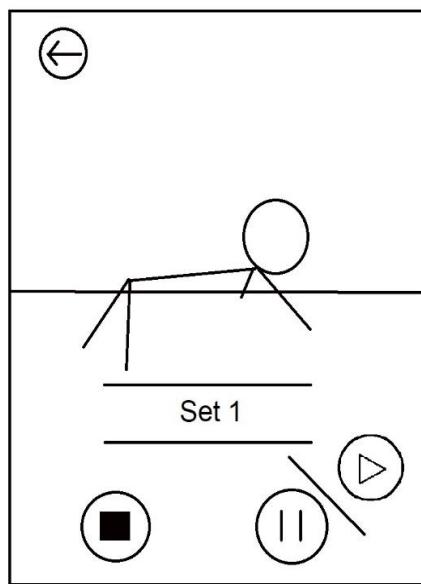
מסך התרגולים של האימון הנבחרת במסך הראשי:  
מסך זה מופיע בלחיצה על תכניות במסך הראשי בו מופיעים כל התכניות.

כל הרשומות של התרגולים המשויכים לאימון שנבחר מוצאים מבסיס הנתונים. במסך התרגולים מופיעים בראשימה אחת נתונים מרשותם של בטבת התרגולים שהוזנו כרגע.

כל תרגיל בראשימה המוצגת מופיע עם שם התרגיל, זמן ממולץ לעשיית התרגיל ותמונה של התרגיל.

שם תרגיל נלקח מהשדה "שם התרגיל", הזמן המומלץ לעשיית התרגיל נלקח מהשדה "המלצת" והתמונה של התרגיל מובאת מהשדה "תמונה של התרגיל" מטבלת התרגולים עבור כל תרגיל בראשימה.

הכפטור למטה משמאלי עם העמודות ייקח את המשטמש למסך בו מופיע סיכום של כל ניתוח המידע והסטטיסטיקה על האימון הנבחר ספציפית.



#### מסך ביצוע התרגיל:

במסך זה נלקחים התמונות של התרגיל, שתהייה אינטימית בהפעלת התרגיל והתרגיל ימשיך כמספר הסטים הכתובים בתרגיל הנבחר במסך התרגולים.

התמונה של התרגיל מובאת מהשדה "תמונה של התרגיל" ומספר הסטים של התרגיל נלקח מהשדה "סטים" מטבלת התרגולים.

לחיצה על הכפטור משמאלי למטה, כפתור הריבוע השחור, עוצרת את ביצוע התרגיל. בלחיצה זו הטימר עוצר. אחרי לחיצה על כפתור הריבוע השחור ימשיך המשטמש לעשות את שאר הסטים של התרגיל. ברגע שישים המשטמש את הסט האחרון הוא ימשיך למסך הכנסת תוכאות והسبיאות רצון.

התוכנה מקבלת את הזמן שלקח למתאמן לבצע את התרגיל. כולל ואת הזמן הזה היא מכניסה לשדה "זמן עשייה של תרגיל" בטבלת התרגולים.

**Results**

How much did you do in each set?

Set 1:	25	▼
Set 2:	25	▼
Set 3:	25	▼

Rate the difficulty: (1 -> 5)

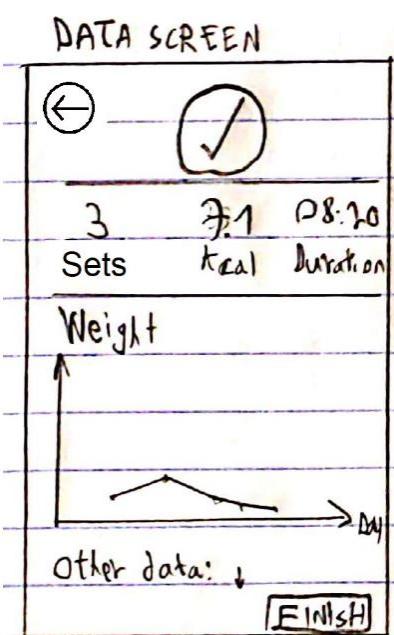
1	2	3	4	5
X	O	O	O	O

**FINISH**

**מסך חנשת תוצאות לתוכנית ושביאות רצון :**

במסך זה מכניס המתאיםן עברו כל סט את מספר הביצועים שהוא עשה ואת רמת הקושי האישית שלו ביצוע התרגיל.

מספר הביצועים הכלול יכנס לשדה "מספר הביצועים" ורמת הקושי שהכניס המשמש תיכנס לשדה "רמת הקושי" עבור תרגיל זה בטבלת התרגילים.

**מסך דיווח הנתונים אחרי ביצוע תרגיל :**

מסך זה מופיע אחרי ביצוע מלא של התרגיל והכנסת תוצאות ביצוע התרגיל. מעלה מופיע ווי בתור אישור על סיום ביצוע התרגיל.

באמצע מופיעים: מספר הסטים שביצע המשמש, מספר הקלוריות שירדו בעת ביצוע התרגיל והזמן הכלול שלקח לבצע את התרגיל.

מספר הסטים שביצע המשמש נלקח מהשדה "סטים" מהתרגיל הנעשה מטבלת התרגילים.

מספר הקלוריות שירדו בעת ביצוע התרגיל ייחס לפי זמן הביצוע של התרגיל מהשדה "זמן עשייה של תרגיל" מהתרגיל הנעשה מטבלת התרגילים.

זמן הכלול שלקח לבצע את התרגיל יילקח מהשדה "זמן עשייה של תרגיל" מהתרגיל הנעשה מטבלת התרגילים.

מתחת לנתונים אלה מופיעים גרפים המספקים ניתוח וייזואלי על מצבו של המתאיםן באותו התרגיל שביצע כרגע ביחס לפעמים קודמות שביצע את התרגיל.

הסטטיסטיקה מבוססת על מספר הביצועים עברו כל סט, זמן עשייה של כל

תרגיל, התאריך שבוצע בו התרגיל והמשוב שלו על רמת הקושי של התרגיל הנלקחים מהשדות "מספר הביצועים", "סטים", "תאריך ביצוע התרגיל",

"זמן עשייה של תרגיל"- "רמת קושי" מטבלת התרגילים.

לחיצה על כפתור הסיום למיטה מימין או על החץ מימין מעלה מוחזירה את המתאיםן למסך התרגילים בו מופיעים כל התרגילים של האימון.

## כל המחלקות והקבצים בפרויקט

### AndroidManifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.fitnote13022021">

    <uses-permission android:name="android.permission.READ_CONTACTS" />

    <uses-feature
        android:name="android.hardware.Camera"
        android:required="true" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/logo_only_pic"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.FitNote13022021">
        <activity android:name=".EditUserActivity"></activity>
        <activity android:name=".ManagerUsersActivity" />
        <activity
            android:name=".InformationActivity"
            android:label="@string/title_activity_information"
            android:theme="@style/Theme.FitNote13022021.NoActionBar" />
        <activity
            android:name=".ViewExercisesResultsActivity"
            android:label="Results" />
        <activity
            android:name=".SettingsActivity"
            android:label="Settings" />
        <activity
            android:name=".WhatsAppSendActivity"
            android:label="Whats App Send" />
        <activity
            android:name=".ShareActivity"
            android:label="Share With Friends" />
        <activity
            android:name=".StatisticsActivity"
            android:label="Statistics" />
        <activity
            android:name=".FeedbackActivity"
            android:label="feedback" />
        <activity
            android:name=".ExecuteExerciseActivity"
            android:label="execute-training" />

        <service android:name=".MusicAndTimerService"
            android:exported="false"/>

        <activity
            android:name=".AddExerciseActivity"
            android:label="AddExercise" />
        <activity
            android:name=".ProgramUserActivity"
```

```
        android:label="User-Program" />
<activity
    android:name=".RegisterActivity"
    android:label="Register" />
<activity
    android:name=".MainScreenActivity"
    android:label="FitNote">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity> <!-- Register the Reminder Receiver -->
<receiver
    android:name=".ReminderBroadcast"
    android:enabled="true" />
</application>

</manifest>
```

## Java Folder Classes

### AddExerciseActivity

```
package com.example.fitnote13022021;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.os.Handler;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.ListView;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.SearchView;
import android.widget.TextView;
import android.widget.Toast;

import java.util.ArrayList;
import java.util.Locale;

public class AddExerciseActivity extends AppCompatActivity implements
View.OnClickListener, AdapterView.OnItemClickListener {

    DataBaseHelper DataBaseHelper;
    ListView listViewExercises;
    ArrayList<Exercise> exercises;
    ArrayList<Exercise> filteredExercises;
    ExerciseAdaptor exerciseAdaptor;
    Button btCancel;
    SearchView searchViewExerciseList;
    String activeUserName;

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

        DataBaseHelper = new DataBaseHelper(this);

        listViewExercises = findViewById(R.id.listViewExercises);
        btCancel = findViewById(R.id.btCancel);
```

```
searchViewExerciseList = findViewById(R.id.searchViewExerciseList);

exercises = DataBaseHelper.getExercises();

//setting filteredExercises to be exercises
filteredExercises = exercises;

exerciseAdaptor = new ExerciseAdaptor(exercises, this);

listViewExercises.setAdapter(exerciseAdaptor);

//initialize the searchWidget in order to filter exercises
initSearchWidgets();

//settings onClick Listeners
listViewExercises.setOnItemClickListener(this);

btCancel.setOnClickListener(this);

//getting userName from intent
Intent intent = getIntent();

activeUserName = intent.getStringExtra("activeUserName");

}

//method to go back to ProgramUserActivity
@Override
public void onClick(View v) {

    if (v == btCancel){

        Intent intent = new Intent(this, ProgramUserActivity.class);

        intent.putExtra("activeUserName", activeUserName);

        startActivity(intent);

        finish();
    }
}

//method to add this specific exercise
@Override
public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

    //gets the clicked exercise
    Exercise chosenExercise = filteredExercises.get(position);

    int exerciseID = chosenExercise.getExerciseID();

    AlertDialog.Builder builder = new AlertDialog.Builder(this);

    builder.setMessage("Do you want to add this exercise?");
}
```

```

//Setting the builder's Layout inflater
View alertDialog =
getLayoutInflater().inflate(R.layout.exercise_with_description_layout, null);

//getting views from Layout
ImageView exerciseImage =
alertDialog.findViewById(R.id.imageViewOfExerciseInDescription);
TextView txtExerciseNameInDescription =
alertDialog.findViewById(R.id.txtExerciseNameInDescription);
TextView txtExerciseDescription =
alertDialog.findViewById(R.id.txtExerciseDescription);

//getting the whole chosen exercise information
Exercise exerciseChosen = null;

for (int i=0; i<exercises.size(); i++) {
    Exercise exercise = exercises.get(i);
    if(exercise.getExerciseID() == exerciseID){
        exerciseChosen = exercise;
    }
}

//Setting the imageView to show exercise image
exerciseImage.setImageResource(exerciseChosen.getExercisePic());

//Setting textViews to show exercise details
txtExerciseNameInDescription.setText(exerciseChosen.getExerciseName());
txtExerciseDescription.setText(exerciseChosen.getExerciseDetail());

//Setting the view to be in builder of alert dialog
builder.setView(alertDialog);

builder.setPositiveButton("Yes", new DialogInterface.OnClickListener() {
    public void onClick(DialogInterface dialog, int id) {

        UserExercise newUserExercise = new UserExercise(activeUserName,
        exerciseID);

        //method to insert UserExercise
        //public boolean insertUserExercise(UserExercise userExercise)
        //so we will turn this chosen exercise into a new userExercise
        //the builder - (String userName, int exerciseID)
        DataBaseHelper.insertUserExercise(newUserExercise);

        Intent intent = new Intent(AddExerciseActivity.this,
        ProgramUserActivity.class);

        intent.putExtra("activeUserName", activeUserName);

        startActivity(intent);

        finish();

    }
});

builder.setNegativeButton("Cancel", new DialogInterface.OnClickListener()
{
}
);

```

```

        public void onClick(DialogInterface dialog, int id) {
            dialog.cancel();
        });
    });

    //Creating and showing the dialog
    AlertDialog dialog = builder.create();

    dialog.show();
}

//initialize the searchWidget in order to filter exercises
public void initSearchWidgets(){

    SearchView searchView = (SearchView)searchViewExerciseList;

    searchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() {
        @Override
        public boolean onQueryTextSubmit(String query) {
            return false;
        }

        //this method is called every time a user
        //puts in any character into the search view
        //literally any chang :D
        @Override
        public boolean onQueryTextChange(String newText) {

            filteredExercises = new ArrayList<Exercise>();

            //same as a regular for loop
            for (Exercise exercise: exercises)
            {
                //if an exercise has one of the letter in the written text

                if(exercise.getExerciseName().toLowerCase().contains(newText.toLowerCase(Locale.ROOT))){
                    filteredExercises.add(exercise);
                }
            }

            ExerciseAdaptor exerciseAdaptor = new
            ExerciseAdaptor(filteredExercises, AddExerciseActivity.this);

            listViewExercises.setAdapter(exerciseAdaptor);

            return false;
        };
    });
}

@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    return super.onOptionsItemSelected(item);
}

```

```
//a method to go back to ProgramUserActivity when clicking back arrow in  
bottom  
    @Override  
    public void onBackPressed() {  
  
        Intent intent = new Intent(this, ProgramUserActivity.class);  
  
        intent.putExtra("activeUserName", activeUserName);  
  
        startActivity(intent);  
  
        finish();  
    }  
}
```

**Admin**

```

package com.example.fitnote13022021;

public class Admin extends User{

    private boolean isSuperAdmin;

    //constructors
    //constructor for all info
    public Admin(String userName, String userPassword,boolean isSuperAdmin , int
userWeight, int userHeight, String userBirthDate, String userGender, String
profilePic) {

        super(userName, userPassword, userWeight, userHeight, userBirthDate,
userGender, profilePic);

        this.isSuperAdmin = isSuperAdmin;

    }

    //constructor for user
    public Admin(User user, boolean isSuperAdmin ) {

        super(user.getUserName(), user.getUserPassword(), user.getUserWeight(),
user.getUserHeight(), user.getUserBirthDate(), user.getUserGender(),
user.getProfilePic());

        this.isSuperAdmin = isSuperAdmin;

    }

    // toString is necessary for printing the contents of a class object
    @Override
    public String toString() {
        return super.toString()+
            "\n" +
            "Admin{" +
            "isSuperAdmin=" + isSuperAdmin +
            '}';
    }

    //Getters and Setters
    public boolean isSuperAdmin() {
        return isSuperAdmin;
    }

    public void setSuperAdmin(boolean superAdmin) {
        isSuperAdmin = superAdmin;
    }

    public int getUserLevel(){
        if (isSuperAdmin())
            return 2;
        else
            return 1;
    }
}

```

ContactListAdapter

```

package com.example.fitnote13022021;

import android.app.Activity;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

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

import java.util.ArrayList;

public class ContactListAdapter extends
RecyclerView.Adapter<ContactListAdapter.ViewHolder> {

    //Initialize variable
    Activity activity;
    ArrayList<ContactModel> arrayList;
    RecycleViewClickListener listener;

    //Create constructor
    public ContactListAdapter(Activity activity, ArrayList<ContactModel>
arrayList, RecycleViewClickListener listener){
        this.activity = activity;
        this.arrayList = arrayList;
        this.listener = listener;
        notifyDataSetChanged();
    }

    @NonNull
    @Override
    public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType)
{
    //Initialize variable
    View view = LayoutInflater.from(parent.getContext())
        .inflate(R.layout.item_contact,parent,false);
    //Return view
    return new ViewHolder(view);
}

    @Override
    public void onBindViewHolder(@NonNull ViewHolder holder, int position) {
        //Initialize contact model
        ContactModel model = arrayList.get(position);

        //Set name
        holder.tv_nameContact.setText(model.getName());

        //Set number
        holder.tv_numberContact.setText(model.getNumber());
    }

    @Override
    public int getItemCount() {

```

```
//Return array list size
    return arrayList.size();
}

public interface RecycleViewClickListener {
    void onClick(View v, int position);
}

public class ViewHolder extends RecyclerView.ViewHolder implements
View.OnClickListener {

    //Initialize variable
    TextView tv_nameContact, tv_numberContact;

    public ViewHolder(@NonNull View itemView) {
        super(itemView);
        //Assign variable
        tv_nameContact = itemView.findViewById(R.id.tv_nameContact);
        tv_numberContact = itemView.findViewById(R.id.tv_numberContact);
        itemView.setOnClickListener(this);
    }

    // A method that is triggered when you click an item in the list
    @Override
    public void onClick(View v) {
        // we need to pass the view and the position in the array
        listener.onClick(v, getAdapterPosition());
    }
}
}
```

## ContactModel

```
package com.example.fitnote13022021;

public class ContactModel {

    //Initialize variables
    String name, number;
    //Generate getter and setter
    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getNumber() {
        return number;
    }

    public void setNumber(String number) {
        this.number = number;
    }

}
```

## DataBaseHelper

```
package com.example.fitnote13022021;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.util.Log;

import androidx.annotation.Nullable;

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

public class DataBaseHelper extends SQLiteOpenHelper {

    //Initialize variables
    private static final String DATABASE_NAME = "fitnote_database12345678";
    private static final int DATABASE_VERSION = 1;

    private static final String TABLE_USER = "Users";
    private static final String TABLE_EXERCISES = "Exercises";
    private static final String TABLE_USEREXERCISES = "UserExercises";
    private static final String TABLE_SONGS = "Songs";

    DataBaseHelper(Context context){
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }

    public DataBaseHelper(@Nullable Context context, @Nullable String name,
    @Nullable SQLiteDatabase.CursorFactory factory, int version) {
        super(context, name, null, version);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        //Create tables
        String tableUser = "CREATE TABLE " + TABLE_USER + " (userName TEXT PRIMARY
KEY, userPassword TEXT, userLevel INTEGER, userWeight INTEGER, userHeight INTEGER,
userBirthDate TEXT, userGender TEXT, profilePic INTEGER)";
        //userLevel 0-normal, 1-admin, 2-superAdmin
        String tableExercise = "CREATE TABLE " + TABLE_EXERCISES + " (exerciseID
INTEGER PRIMARY KEY, exerciseName TEXT, exercisePic INTEGER, exerciseDetail
TEXT)";
        String tableUserExercises = "CREATE TABLE " + TABLE_USEREXERCISES + "
(userExerciseID INTEGER PRIMARY KEY AUTOINCREMENT, userName TEXT, exerciseID
INTEGER, date TEXT, time INTEGER, rating INTEGER, repetition INTEGER)";
        String tableSongs = "CREATE TABLE " + TABLE_SONGS + " (songID INTEGER
PRIMARY KEY, songName TEXT, songMP3 INTEGER)";

        db.execSQL(tableUser);
        db.execSQL(tableExercise);
        db.execSQL(tableUserExercises);
        db.execSQL(tableSongs);
    }
}
```

```

// adding exercises
// Exercise Table;
// (exerciseID INT PRIMARY KEY, exerciseName TEXT,
// exercisePic INT, exerciseDetail TEXT)
db.execSQL("INSERT INTO " + TABLE_EXERCISES + " VALUES (1, 'Push Up',
"+R.drawable.pushup+", '1. Get down on all fours, placing your hands slightly
wider than your shoulders. " +
"\n" +
"2. Straighten your arms and legs. " +
"\n" +
"3. Lower your body until your chest nearly touches the floor. " +
"\n" +
"4. Pause, then push yourself back up. " +
"\n" +
"5. Repeat. ')");
db.execSQL("INSERT INTO " + TABLE_EXERCISES + " VALUES (2, 'Squat',
"+R.drawable.squat+, '1. Find a foot stance that feels best for you. Pointing
your toes slightly outwards helps some, but keeping them parallel is fine, too. If
you're not sure what's best, start by putting your feet shoulder-width apart and
pointed about 15 degrees outwards. " +
"\n" +
"2. Tense your abs like someone is about to punch you. " +
"\n" +
"3. Look straight ahead and stand tall!'");
db.execSQL("INSERT INTO " + TABLE_EXERCISES + " VALUES (3, 'Running',
"+R.drawable.running+, 'TIPS: When you first start out, try alternating between
running and walking during your session. As time goes on, make the running
intervals longer until you no longer feel the need to walk. " +
"Give yourself a few minutes to cool down after each run by
walking and doing a few stretches. Try our post-run stretch routine.'");
db.execSQL("INSERT INTO " + TABLE_EXERCISES + " VALUES (4, 'RopeJump',
"+R.drawable.ropejump+, 'Get a large rope that you can pass under your feet. Move
the rope under your feet while you jump and repeat.'");
db.execSQL("INSERT INTO " + TABLE_EXERCISES + " VALUES (5, 'JumpingJacks',
"+R.drawable.jumping_jacks+, 'JumpingJacks: Stand upright with your legs
together, arms at your sides. " +
"\n" +
"Bend your knees slightly, and jump into the air. " +
"\n" +
"As you jump, spread your legs to be about shoulder-width apart.
Stretch your arms out and over your head. " +
"\n" +
"Jump back to starting position. " +
"\n" +
"Repeat. '");
db.execSQL("INSERT INTO " + TABLE_EXERCISES + " VALUES (6, 'TricepsDips',
"+R.drawable.triceps_dips+, '1. You can do it from a chair or a bench, " +
"you can even do it on the floor. " +
"\n" +
"2. So just getting yourself into position to start by rolling
your shoulders down. " +
"You want to start from good posture. Sometimes we tend to slump
forward and then we're gonna potentially cause an extra strain on the shoulder

```

```

joint. "+  

        "\n" +  

        "3. So set yourself up by rolling the shoulders back, opening up  

the chest. Bringing your hands directly underneath your shoulders onto the bench  

or onto the ground. "+  

        "And youre gonna take your legs out keeping your knees bent." +  

        "\n" +  

        "4. If you wanted to make it harder you could extend the legs. "+  

        " So start with the easier option, see how you feel first. "+  

        "\n" +  

        "5. Youre gonna bend the elbows, lowering the hips down, and then  

exhale to extend the elbows and lift your body up. "+  

        "Inhale down, exhale up.'");  

        db.execSQL("INSERT INTO " + TABLE_EXERCISES + " VALUES (7,  

'InclinePushUp', "+R.drawable.incline_pushup+", '1. Place your hands on the edge  

of the elevated surface. ' +  

        "\n" +  

        "2. Step your feet back so your legs are straight and your arms  

are perpendicular to your body. "+  

        "\n" +  

        "3. Inhale as you slowly lower your chest to the edge of your  

platform. ' +  

        "\n" +  

        "4. Pause for a second. "+  

        "\n" +  

        "5. Exhale as you push back to your starting position with your  

arms fully extended.')");  

        db.execSQL("INSERT INTO " + TABLE_EXERCISES + " VALUES (8, 'JumpingSquat',  

"+R.drawable.jumping_squat+", 'Stand tall with your feet hip-width apart. '+  

        "\n" +  

        "Hinge at the hips to push your butt back and lower down until  

your thighs are parallel to the floor. "+  

        "\n" +  

        "Allow your knees to bend 45 degrees when you land, and then  

immediately drop back down into a squat, and jump again.')");  

        db.execSQL("INSERT INTO " + TABLE_EXERCISES + " VALUES (9, 'HammerCurls',  

"+R.drawable.hammer_curls+", 'Step 1 '+  

        "Stand up straight with a dumbbell in each hand, holding them  

alongside you. Your palms should face your body. Keep your feet hip-width apart  

and engage your core to stabilize the body. "+  

        "\n" +  

        "Step 2 "+  

        "Keep your biceps stationary and start bending at your elbows,  

lifting both dumbbells. "+  

        "\n" +  

        "Step 3 "+  

        "Lift until the dumbbells reach shoulder-level, but don't actually  

touch your shoulders. Hold this contraction briefly, then lower back to the  

starting position and repeat.')");  

        db.execSQL("INSERT INTO " + TABLE_EXERCISES + " VALUES (10,  

'ShoulderPress', "+R.drawable.shoulder_press+", 'Stand with feet shoulder-width  

apart and hold the dumbbells at shoulder height with your elbows at a 90-degree  

angle. "+  

        "\n" +

```

```

        "Slowly lift the dumbbells above your head without fully
straightening your arms. Pause at the top. " +
        "\n" +
        "Slowly return to the start position.'"));

        db.execSQL("INSERT INTO " + TABLE_EXERCISES + " VALUES (11, 'Swimming',
"+R.drawable.swimming+", '1. Float with your face in the water, your body straight
and horizontal. Stack your hands and keep your arms and legs long. " +
        "Point your thumbs down. " +
        "\n" +
        "2. Press your hands out and back in a circle, elbows high. Lift
your head slightly and inhale. " +
        "\n" +
        "3. Bring your hands together in front of your shoulders, thumbs
pointing up. Keep your elbows close to your body. Simultaneously bend your knees,
bringing your feet toward your butt and pointing your feet outward. " +
        "\n" +
        "4. Reach your arms forward. Kick out and back in a circle then
snap your feet together. Drop your head underwater and exhale. " +
        "\n" +
        "5. Glide forward and repeat.'"));

        // adding songs
        db.execSQL("INSERT INTO " + TABLE_SONGS + " VALUES (1, 'speedrun',
"+R.raw.speedrun+"));
        db.execSQL("INSERT INTO " + TABLE_SONGS + " VALUES (2, 'speedrun2',
"+R.raw.speedrun2+"));
        db.execSQL("INSERT INTO " + TABLE_SONGS + " VALUES (3, 'speedrun3',
"+R.raw.speedrun3+"));
        db.execSQL("INSERT INTO " + TABLE_SONGS + " VALUES (4, 'phantom',
"+R.raw.phantom+"));
        db.execSQL("INSERT INTO " + TABLE_SONGS + " VALUES (5, 'shoping',
"+R.raw.shoping+"));

        //Table Users:
        //((userName TEXT PRIMARY KEY, userPassword TEXT, userLevel INTEGER,
        // userWeight INTEGER, userHeight INTEGER, userBirthDate TEXT, userGender
        TEXT, profilePic INTEGER)
        //adding user alon - me the SUPER ADMIN
        db.execSQL("INSERT INTO " + TABLE_USER + " VALUES('alon', 123, 2, 90, 90,
'APR 2 2003', 'male', 'alon')");

        //adding userExercises
        //JAN FEB MAR APR MAY JUN JUL AUG SEP OCT NOV DEC
        //((userExerciseID INTEGER PRIMARY KEY AUTOINCREMENT, userName TEXT,
        // exerciseID INTEGER, date TEXT, time INTEGER, rating INTEGER, repetition
        INTEGER)

        db.execSQL("INSERT INTO " + TABLE_USEREXERCISES + " VALUES(1, 'alon', 1,
'MAR 7 2021', 100, 1, 100)");
        db.execSQL("INSERT INTO " + TABLE_USEREXERCISES + " VALUES(2, 'alon', 2,
'JAN 8 2021', 10, 2, 10)");
        db.execSQL("INSERT INTO " + TABLE_USEREXERCISES + " VALUES(3, 'alon', 3,

```

```

'FEB 9 2021', 60, 3, 60"));
        db.execSQL("INSERT INTO " + TABLE_USEREXERCISES + " VALUES(4, 'alon', 4,
'APR 10 2020', 30, 4, 30)");
        db.execSQL("INSERT INTO " + TABLE_USEREXERCISES + " VALUES(5, 'alon', 5,
'JUN 11 2020', 50, 5, 50)");
        db.execSQL("INSERT INTO " + TABLE_USEREXERCISES + " VALUES(6, 'alon', 6,
'SEP 25 2020', 100, 1, 100)");
        db.execSQL("INSERT INTO " + TABLE_USEREXERCISES + " VALUES(7, 'alon', 7,
'MAR 7 2021', 20, 2, 20)");
        db.execSQL("INSERT INTO " + TABLE_USEREXERCISES + " VALUES(8, 'alon', 8,
'JAN 8 2021', 56, 3, 56)");
        db.execSQL("INSERT INTO " + TABLE_USEREXERCISES + " VALUES(9, 'alon', 9,
'FEB 9 2021', 34, 4, 34)");
        db.execSQL("INSERT INTO " + TABLE_USEREXERCISES + " VALUES(10, 'alon', 10,
'APR 10 2020', 70, 5, 70)");
        db.execSQL("INSERT INTO " + TABLE_USEREXERCISES + " VALUES(11, 'alon', 11,
'MAR 11 2021', 120, 1, 110)");

    }

//Delete database
public static void deleteDatabase(Context mContext) {
    mContext.deleteDatabase(DATABASE_NAME);
}

@Override
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
    //Drop Existing Table
    db.execSQL("DROP TABLE IF EXISTS " + TABLE_USER);
    db.execSQL("DROP TABLE IF EXISTS " + TABLE_EXERCISES);
    db.execSQL("DROP TABLE IF EXISTS " + TABLE_USEREXERCISES);
    onCreate(db);
}

//method to add user
public boolean insertUser(User user) {
    //getWritableDatabase - method from the default properties of
SQLiteOpenHelper
    //getWritableDatabase for insert actions...getReadableDatabase for select
(read) actions.
    SQLiteDatabase db = this.getWritableDatabase();

    // User Table:
    // (userName TEXT PRIMARY KEY, userPassword TEXT, userLevel INTEGER,
    // userWeight INTEGER, userHeight INTEGER, userBirthDate TEXT, userGender
TEXT, profilePic INTEGER)
    ContentValues cv = new ContentValues();
    cv.put("userName", user.getUserName());
    cv.put("userPassword", user.getUserPassword());
    cv.put("userLevel", 0);
    cv.put("userWeight", user.getUserWeight());
    cv.put("userHeight", user.getUserHeight());
    cv.put("userBirthDate", user.getUserBirthDate());
}

```

```

cv.put("userGender", user.getUserGender());
cv.put("profilePic", user.getProfilePic());

// insert - success variable... 1)positive -> it was inserted...2) -1 ->
it was a fail
long insert = db.insert(TABLE_USER, null, cv);

if(insert == -1){
    return false;
}else{
    return true;
}

//method to add user
public boolean insertAdmin(Admin admin) {
    //getWritableDatabase - method from the default properties of
SQLiteOpenHelper
    //getWritableDatabase for insert actions...getReadableDatabase for select
(read) actions.
    SQLiteDatabase db = this.getWritableDatabase();

    int userLevel = 1;

    if (admin.isSuperAdmin() == false){
        userLevel = 1;
    }
    else {
        userLevel = 2;
    }

    // User Table:
    // (userName TEXT PRIMARY KEY, userPassword TEXT, userLevel INTEGER,
    // userWeight INTEGER, userHeight INTEGER, userBirthDate TEXT, userGender
TEXT, profilePic INTEGER)
    ContentValues cv = new ContentValues();
    cv.put("userName", admin.getUserName());
    cv.put("userPassword", admin.getUserPassword());
    cv.put("userLevel", userLevel);
    cv.put("userWeight", admin.getUserWeight());
    cv.put("userHeight", admin.getUserHeight());
    cv.put("userBirthDate", admin.getUserBirthDate());
    cv.put("userGender", admin.getUserGender());
    cv.put("profilePic", admin.getProfilePic());

    // insert - success variable... 1)positive -> it was inserted...2) -1 ->
it was a fail
    long insert = db.insert(TABLE_USER, null, cv);

    if(insert == -1){
        return false;
    }else{
        return true;
    }
}

```

```

//method to update User
public boolean updateUser(User user){

    //getWritableDatabase - method from the default properties of
SQLiteOpenHelper
    //getWritableDatabase for insert actions...getReadableDatabase for select
(read) actions.
    SQLiteDatabase db = this.getWritableDatabase();

    // ContentValues - a special class that works like a associative array
(PHP)
    // can take pairs of values and associate with them (like the bundle in
intent)
    // The ID column is auto increment
    // User Table:
    // (userName TEXT PRIMARY KEY, userPassword TEXT, userLevel INTEGER,
    // userWeight INTEGER, userHeight INTEGER, userBirthDate TEXT, userGender
TEXT, profilePic INTEGER)
    ContentValues cv = new ContentValues();
    cv.put("userName", user.getUserName());
    cv.put("userPassword", user.getUserPassword());
    cv.put("userLevel", user.getUserLevel());
    cv.put("userWeight", user.getUserWeight());
    cv.put("userHeight", user.getUserHeight());
    cv.put("userBirthDate", user.getUserBirthDate());
    cv.put("userGender", user.getUserGender());
    cv.put("profilePic", user.getProfilePic());

    // insert - success variable... 1)positive -> it was inserted...2) -1 ->
it was a fail
    long result = db.update(TABLE_USER, cv, "userName=?", new String
[]{user.getUserName()});

    if(result == -1){
        return false;
    }else{
        return true;
    }
}

//method to update User
public boolean updateUserAsAdmin(Admin user){

    //getWritableDatabase - method from the default properties of
SQLiteOpenHelper
    //getWritableDatabase for insert actions...getReadableDatabase for select
(read) actions.
    SQLiteDatabase db = this.getWritableDatabase();

    int userLevel = 1;

    if (user.isSuperAdmin())
        userLevel = 2;

    // ContentValues - a special class that works like a associative array
(PHP)
    // can take pairs of values and associate with them (like the bundle in

```

```

intent)
    // The ID column is auto increment
    // User Table:
    // (userName TEXT PRIMARY KEY, userPassword TEXT, userLevel INTEGER,
    // userWeight INTEGER, userHeight INTEGER, userBirthDate TEXT, userGender
TEXT, profilePic INTEGER)
    ContentValues cv = new ContentValues();
    cv.put("userName", user.getUserName());
    cv.put("userPassword", user.getUserPassword());
    cv.put("userLevel", userLevel);
    cv.put("userWeight", user.getUserWeight());
    cv.put("userHeight", user.getUserHeight());
    cv.put("userBirthDate", user.getUserBirthDate());
    cv.put("userGender", user.getUserGender());
    cv.put("profilePic", user.getProfilePic());

    // insert - success variable... 1)positive -> it was inserted...2) -1 ->
it was a fail
    long result = db.update(TABLE_USER, cv, "userName=?", new String
[]{user.getUserName()});

    if(result == -1){
        return false;
    }else{
        return true;
    }
}

//method to delete user (by ID because its the primary key)
public void deleteUser(User user) {
    // find user in the database. if its found, delete it and return true.
    // if its not found, return false.

    // getWritableDatabase - we are going to delete from it
    SQLiteDatabase db = this.getWritableDatabase();

    db.delete(TABLE_USER, "userName=?", new String[]{user.getUserName()});
}

//method to get user's level
public int getUserLevelByUserName(String givenUserName){
    ArrayList<User> returnList = new ArrayList<>();

    int userLevel = 0;

    // User Table:
    // (userName TEXT PRIMARY KEY, userPassword TEXT, userLevel INTEGER,
    // userWeight INTEGER, userHeight INTEGER, userBirthDate TEXT, userGender
TEXT, profilePic INTEGER)
    // get data from the database
    String queryString = ("SELECT " + "*" + " FROM " + TABLE_USER + " WHERE "
+ "userName" + " = '" + givenUserName + "'");

    // get a reference to the active database
    // getWritableDatabase - insert, update or delete records
    // getReadableDatabase - SELECT items from the database

```

```

SQLiteDatabase db = this.getReadableDatabase();

// Cursor is the result set from a SQL statement
Cursor cursor = db.rawQuery(queryString, null);

// moveToFirst returns a true if there were items selected
if(cursor.moveToFirst()){

    userLevel = cursor.getInt(2);

}else{
    // failure. do not add anything to the list.
}

// always clean up after yourself
// lets close the connection to the database so others can use it
// close the cursor when done.
cursor.close();

return userLevel;
}

//method to get all users
public ArrayList<User> getUsers(){

    ArrayList<User> returnList = new ArrayList<>();

    // get data from the database
    String queryString = "SELECT * FROM " + TABLE_USER;

    // get a reference to the active database
    // getWritableDatabase - insert, update or delete records
    // getReadableDatabase - SELECT items from the database
    SQLiteDatabase db = this.getReadableDatabase();

    // Cursor is the result set from a SQL statement
    Cursor cursor = db.rawQuery(queryString, null);

    // moveToFirst returns a true if there were items selected
    if(cursor.moveToFirst()){

        // Loop through the cursor (result set) and create new customer
        // objects. Put them into the return list
        do{
            // User Table:
            // (userName TEXT PRIMARY KEY, userPassword TEXT, userLevel
INTEGER,
            // userWeight INTEGER, userHeight INTEGER, userBirthDate TEXT,
userGender TEXT, profilePic INTEGER)
            String userName = cursor.getString(0);
            String userPassword = cursor.getString(1);
            int userLevel = cursor.getInt(2);
            int userWeight = cursor.getInt(3);
            int userHeight = cursor.getInt(4);
            String userBirthDate = cursor.getString(5);
            String userGender = cursor.getString(6);
            String profilePic = cursor.getString(7);
        }
    }
}

```

```
User newUser = new User(userName, userPassword, userWeight,
userHeight, userBirthDate, userGender, profilePic);
returnList.add(newUser);

} while (cursor.moveToNext());

}  

// failure. do not add anything to the list.
}

// always clean up after yourself
// Lets close the connection to the database so others can use it
// close the cursor when done.
cursor.close();

return returnList;
}

//method to get all users
public ArrayList<User> getUsersAndAdmins(){

ArrayList<User> returnList = new ArrayList<>();

// get data from the database
String queryString = "SELECT * FROM " + TABLE_USER;

// get a reference to the active database
// getWritableDatabase - insert, update or delete records
// getReadableDatabase - SELECT items from the database
SQLiteDatabase db = this.getReadableDatabase();

// Cursor is the result set from a SQL statement
Cursor cursor = db.rawQuery(queryString, null);

// moveToFirst returns a true if there were items selected
if(cursor.moveToFirst()){

    // Loop through the cursor (result set) and create new customer
    objects. Put them into the return list
    do{
        // User Table:
        // (userName TEXT PRIMARY KEY, userPassword TEXT, userLevel
        INTEGER,
        // userWeight INTEGER, userHeight INTEGER, userBirthDate TEXT,
        userGender TEXT, profilePic INTEGER)
        String userName = cursor.getString(0);
        String userPassword = cursor.getString(1);
        int userLevel = cursor.getInt(2);
        int userWeight = cursor.getInt(3);
        int userHeight = cursor.getInt(4);
        String userBirthDate = cursor.getString(5);
        String userGender = cursor.getString(6);
        String profilePic = cursor.getString(7);

        //user is Admin at minimum
        if (userLevel == 1 || userLevel == 2){
            //settings boolean of isSuperAdmin
```

```

boolean isSuperAdmin = false;
if (userLevel == 2)
    isSuperAdmin = true;

User newUser = new Admin(userName, userPassword, isSuperAdmin,
userWeight, userHeight, userBirthDate, userGender, profilePic);
returnList.add(newUser);
}

//user is normal
else {
    User newUser = new User(userName, userPassword, userWeight,
userHeight, userBirthDate, userGender, profilePic);
    returnList.add(newUser);
}

}

} while (cursor.moveToNext());

}else{
    // failure. do not add anything to the list.
}

// always clean up after yourself
// lets close the connection to the database so others can use it
// close the cursor when done.
cursor.close();

return returnList;
}

//method to get user with specific name and password
//method to search for user (true - user exists | false - user doesn't exist)
public boolean searchUserByNameAndPass(String userName, String userPassword){
    //get all users from user table
    List<User> Users = this.getUsers();

    // if there are no users
    if(Users.isEmpty() || (userName.isEmpty() && userPassword.isEmpty()) )
        return false;

    for (int i=0; i<Users.size(); i++) {
        User user = Users.get(i);
        if(user.getUserName().contentEquals(userName) &&
user.getUserPassword() .contentEquals(userPassword)){
            return true;
        }
    }

    return false;
}

//method to search for user (true - user exists | false - user doesn't exist)
public boolean searchUserByName(String userName){
    //get all users from user table
    List<User> Users = this.getUsersAndAdmins();
}

```

```

// if there are no users
if(Users.isEmpty())
    return false;

for (int i=0; i<Users.size(); i++) {
    User user = Users.get(i);
    if(user.getUserName().contentEquals(userName)){
        return true;
    }
}

return false;
}

//method to get for user (true - user exists / false - user doesn't exist)
public User getUserByName(String userName){
    //get all users from user table
    ArrayList<User> Users = this.getUsersAndAdmins();

    // if there are no users
    if(Users.isEmpty() || userName == null)
        return null;

    for (int i=0; i<Users.size(); i++) {
        User user = Users.get(i);
        if(user.getUserName().contentEquals(userName)){
            return user;
        }
    }

    return null;
}

//method to insert Exercise
public boolean insertExercise(Exercise exercise){

    //getWritableDatabase - method from the default properties of
    SQLiteOpenHelper
    //getWritableDatabase for insert actions...getReadableDatabase for select
    (read) actions.
    SQLiteDatabase db = this.getWritableDatabase();

    // ContentValues - a special class that works like a associative array
    (PHP)
    // can take pairs of values and associate with them (like the bundle in
    intent)
    // The ID column is auto increment
    // Exercise Table;
    // (exerciseID INTEGER PRIMARY KEY, exerciseName TEXT,
    // exercisePic INTEGER, exerciseDetail TEXT)
    ContentValues cv = new ContentValues();
    cv.put("exerciseID", exercise.getExerciseID());
    cv.put("exerciseName", exercise.getExerciseName());
    cv.put("exercisePic", exercise.getExercisePic());
    cv.put("exerciseDetail", exercise.getExerciseDetail());
}

```

```

    // insert - success variable... 1)positive -> it was inserted...2) -1 ->
it was a fail
    long insert = db.insert(TABLE_EXERCISES, null, cv);

    if(insert == -1){
        return false;
    }else{
        return true;
    }

}

//method to get all exercises
public ArrayList<Exercise> getExercises(){

    ArrayList<Exercise> returnList = new ArrayList<>();

    // get data from the database
    String queryString = "SELECT * FROM " + TABLE_EXERCISES;

    // get a reference to the active database
    // getWritableDatabase - insert, update or delete records
    // getReadableDatabase - SELECT items from the database
    SQLiteDatabase db = this.getReadableDatabase();

    // Cursor is the result set from a SQL statement
    Cursor cursor = db.rawQuery(queryString, null);

    // moveToFirst returns a true if there were items selected
    if(cursor.moveToFirst()){

        // Loop through the cursor (result set) and create new customer
        objects. Put them into the return list
        do{
            // Exercise Table;
            // (exerciseID INTEGER PRIMARY KEY, exerciseName TEXT,
            // exercisePic INTEGER, exerciseDetail TEXT)
            int exerciseID = cursor.getInt(0);
            String exerciseName = cursor.getString(1);
            int exercisePic = cursor.getInt(2);
            String exerciseDetail = cursor.getString(3);

            Exercise newExercise = new Exercise(exerciseID, exerciseName,
            exercisePic, exerciseDetail);
            returnList.add(newExercise);

        } while (cursor.moveToNext());

    }else{
        // failure. do not add anything to the list.
    }

    // always clean up after yourself
    // lets close the connection to the database so others can use it
    // close the cursor when done.
    cursor.close();

    return returnList;
}

```

```

}

//method to get all exercises by userNameGiven
public ArrayList<UserExercisesInnerJoinEx> getExercisesByUserNameGiven(String
userNameGiven){

    ArrayList<UserExercisesInnerJoinEx> returnList = new ArrayList<>();

    // get data from the database
    // First table: Exercises
    // Second table: UserExercises
    // Exercise Table;
    // (exerciseID INTEGER PRIMARY KEY, exerciseName TEXT,
    // exercisePic INTEGER, exerciseDetail TEXT)
    //table UserExercises
    //((userExerciseID INTEGER PRIMARY KEY AUTOINCREMENT,
    // userTextColor TEXT, exerciseID INTEGER, date TEXT, time INTEGER,
    // rating INTEGER, repetition INTEGER)
    String queryString = "SELECT Exercises.exerciseID, exerciseName,
exercisePic, exerciseDetail, userExerciseID FROM Exercises INNER JOIN
UserExercises ON Exercises.exerciseID = UserExercises.exerciseID WHERE
UserExercises.userName = '" + userNameGiven + "'";

    //String queryString = "SELECT Exercises.exerciseID,
UserExercises.exerciseID, exerciseName, exercisePic, exerciseDetail FROM Exercises
INNER JOIN UserExercises ON Exercises.exerciseID = UserExercises.exerciseID WHERE
UserExercises.userName = " + userNameGiven;

    // get a reference to the active database
    // getWritableDatabase - insert, update or delete records
    // getReadableDatabase - SELECT items from the database
    SQLiteDatabase db = this.getReadableDatabase();

    // Cursor is the result set from a SQL statement
    Cursor cursor = db.rawQuery(queryString, null);

    // moveToFirst returns a true if there were items selected
    if(cursor.moveToFirst()){

        // Loop through the cursor (result set) and create new customer
        // objects. Put them into the return list
        do{
            //UserExercisesInnerJoinEx (int userExerciseID, int exerciseID,
String exerciseName, int exercisePic, String exerciseDetail)
            int exerciseID = cursor.getInt(0);
            String exerciseName = cursor.getString(1);
            int exercisePic = cursor.getInt(2);
            String exerciseDetail = cursor.getString(3);
            int userExerciseID = cursor.getInt(4);

            UserExercisesInnerJoinEx newUserExercisesInnerJoinEx = new
UserExercisesInnerJoinEx(userExerciseID, exerciseID, exerciseName, exercisePic,
exerciseDetail);
            returnList.add(newUserExercisesInnerJoinEx);

        } while (cursor.moveToNext());

    }else{
        // failure. do not add anything to the list.
    }
}

```

```

        }

        // always clean up after yourself
        // Lets close the connection to the database so others can use it
        // close the cursor when done.
        cursor.close();

        return returnList;
    }

    //method to get all exercises by userNameGiven
    //that have not been done by the user yet (by knowing that the date of doing
    the exercise is null)
    public ArrayList<UserExercisesInnerJoinEx>
getExercisesByUserNameGivenNotDone(String userNameGiven){

    ArrayList<UserExercisesInnerJoinEx> returnList = new ArrayList<>();

    // get data from the database
    // First table: Exercises
    // Second table: UserExercises
    // Exercise Table;
    // (exerciseID INTEGER PRIMARY KEY, exerciseName TEXT,
    // exercisePic INTEGER, exerciseDetail TEXT)
    //table UserExercises
    // (userExerciseID INTEGER PRIMARY KEY AUTOINCREMENT,
    // userName TEXT, exerciseID INTEGER, date TEXT, time INTEGER,
    // rating INTEGER, repetition INTEGER)
    String queryString = "SELECT Exercises.exerciseID, exerciseName,
exercisePic, exerciseDetail, userExerciseID FROM Exercises INNER JOIN
UserExercises ON Exercises.exerciseID = UserExercises.exerciseID WHERE
UserExercises.userName = '" +userNameGiven+ "' AND UserExercises.date IS NULL";
    //String queryString = "SELECT Exercises.exerciseID,
UserExercises.exerciseID, exerciseName, exercisePic, exerciseDetail FROM Exercises
INNER JOIN UserExercises ON Exercises.exerciseID = UserExercises.exerciseID WHERE
UserExercises.userName = " + userNameGiven;
    // get a reference to the active database
    // getWritableDatabase - insert, update or delete records
    // getReadableDatabase - SELECT items from the database
    SQLiteDatabase db = this.getReadableDatabase();

    // Cursor is the result set from a SQL statement
    Cursor cursor = db.rawQuery(queryString, null);

    // moveToFirst returns a true if there were items selected
    if(cursor.moveToFirst()){

        // Loop through the cursor (result set) and create new customer
        objects. Put them into the return list
        do{
            //UserExercisesInnerJoinEx (int userExerciseID, int exerciseID,
String exerciseName, int exercisePic, String exerciseDetail)
            int exerciseID = cursor.getInt(0);
            String exerciseName = cursor.getString(1);
            int exercisePic = cursor.getInt(2);
            String exerciseDetail = cursor.getString(3);
            int userExerciseID = cursor.getInt(4);

```

```
UserExercisesInnerJoinEx newUserExercisesInnerJoinEx = new
UserExercisesInnerJoinEx(userExerciseID, exerciseID, exerciseName, exercisePic,
exerciseDetail);
    returnList.add(newUserExercisesInnerJoinEx);

} while (cursor.moveToNext());

}  
else{
    // failure. do not add anything to the list.
}

// always clean up after yourself
// lets close the connection to the database so others can use it
// close the cursor when done.
cursor.close();

return returnList;
}

//method to search for exercise by ID
public ArrayList<Exercise> getExercisesByID(int exerciseID){
    //get all users from user table
    ArrayList<Exercise> exercises = this.getExercises();
    ArrayList<Exercise> returnList = new ArrayList<>();

    // if there are no users
    if(exercises.isEmpty())
        return null;

    for (int i=0; i<exercises.size(); i++) {
        Exercise exercise = exercises.get(i);
        if(exercise.getExerciseID() == exerciseID){
            returnList.add(exercise);
        }
    }
    return returnList;
}

//method to get one exercise by its ID
public Exercise getExerciseByID(int id){

    // get a reference to the active database
    // getWritableDatabase - insert, update or delete records
    // getReadableDatabase - SELECT items from the database
    SQLiteDatabase db = getReadableDatabase();

    // Table: Exercises
    // (exerciseID INTEGER PRIMARY KEY, exerciseName TEXT,
    // exercisePic INTEGER, exerciseDetail TEXT)
    Cursor cursor = db.rawQuery("SELECT * FROM Exercises WHERE exerciseID = "
+ id, null);

    cursor.moveToFirst();

}
```

```

int exerciseID = cursor.getInt(0);
String exerciseName = cursor.getString(1);
int exercisePic = cursor.getInt(2);
String exerciseDetail = cursor.getString(3);

Exercise exercise = new Exercise(exerciseID, exerciseName, exercisePic,
exerciseDetail);

// always clean up after yourself
// lets close the connection to the database so others can use it
// close the cursor when done.
cursor.close();

return exercise;
}

//method to insert UserExercise
public boolean insertUserExercise(UserExercise userExercise){

    //getWritableDatabase - method from the default properties of
SQLiteOpenHelper
    //getWritableDatabase for insert actions...getReadableDatabase for select
(read) actions.
    SQLiteDatabase db = this.getWritableDatabase();

    // ContentValues - a special class that works like a associative array
(PHP)
    // can take pairs of values and associate with them (like the bundle in
intent)
    // The ID column is auto increment
    // UserExercises Table
    //((userExerciseID INTEGER PRIMARY KEY, userName TEXT,
    // exerciseID INTEGER, date TEXT, time INTEGER, rating INTEGER, repetition
    INTEGER))
    ContentValues cv = new ContentValues();
    cv.put("userName", userExercise.getUserName());
    cv.put("exerciseID", userExercise.getExerciseID());
    cv.put("date", userExercise.getDate());
    cv.put("time", userExercise.getTime());
    cv.put("rating", userExercise.getRating());
    cv.put("repetition", userExercise.getRepetition());

    // insert - success variable... 1)positive -> it was inserted...2) -1 ->
it was a fail
    long insert = db.insert(TABLE_USEREXERCISES, null, cv);

    if(insert == -1){
        return false;
    }else{
        return true;
    }
}

//method to delete userExercise
//using it to delete userExercise when in dialog to delete an userExercise

```

```

public void deleteUserExercise(String userName, String exerciseName, int
userExerciseID){

    // get a reference to the active database
    // getWritableDatabase - insert, update or delete records
    // getReadableDatabase - SELECT items from the database
    SQLiteDatabase db = this.getWritableDatabase();

    //table: Exercises
    //((exerciseID INTEGER PRIMARY KEY, exerciseName TEXT, exercisePic INTEGER,
    exerciseDetail TEXT)
    Cursor c = db.rawQuery("SELECT * FROM " + TABLE_EXERCISES + " WHERE
    exerciseName = '" + exerciseName + "'", null);

    c.moveToFirst();

    //taking the exerciseID from the selected exercise
    //there is supposed to be only one exercise with the name it has
    int exerciseID = c.getInt(0);
    //Log.d("exerciseID", String.valueOf(exerciseID));
    //table UserExercises
    //((userExerciseID INTEGER PRIMARY KEY AUTOINCREMENT, userName TEXT,
    // exerciseID INTEGER, date TEXT, time INTEGER, rating INTEGER, repetition
    INTEGER)
    String queryString = "DELETE FROM " + TABLE_USEREXERCISES + " WHERE
    userName = '" + userName + "' AND exerciseID = " + exerciseID + " AND userExerciseID =
    " + userExerciseID;

    Log.d("Delete filter", String.valueOf(userExerciseID));
    //Log.d("Delete filter2", String.valueOf(exerciseID));

    db.execSQL(queryString);

}

//method to update userExercise
//((userExerciseID INTEGER PRIMARY KEY AUTOINCREMENT, userName TEXT, exerciseID
INTEGER, date TEXT, time INTEGER, rating INTEGER, repetition INTEGER)
public void updateUserExercise(int userExerciseID, String date, int time, int
rating, int repetition){

    // get a reference to the active database
    // getWritableDatabase - insert, update or delete records
    // getReadableDatabase - SELECT items from the database
    SQLiteDatabase db = this.getWritableDatabase();

    // UserExercises Table
    //((userExerciseID INTEGER PRIMARY KEY, userName TEXT,
    // exerciseID INTEGER, date TEXT, time INTEGER, rating INTEGER, repetition
    INTEGER)
    ContentValues cv = new ContentValues();
    cv.put("date", date);
    cv.put("time", time);
    cv.put("rating", rating);
    cv.put("repetition", repetition);

    db.update(TABLE_USEREXERCISES, cv, "userExerciseID = " +
    userExerciseID, null);
}

```

```
}

//method to get all UserExercises
public ArrayList<UserExercise> getUserExercises(){

    ArrayList<UserExercise> returnList = new ArrayList<>();

    // get data from the database
    String queryString = "SELECT * FROM " + TABLE_USEREXERCISES;

    // get a reference to the active database
    // getWritableDatabase - insert, update or delete records
    // getReadableDatabase - SELECT items from the database
    SQLiteDatabase db = this.getReadableDatabase();

    // Cursor is the result set from a SQL statement
    Cursor cursor = db.rawQuery(queryString, null);

    // moveToFirst returns a true if there were items selected
    if(cursor.moveToFirst()){

        // Loop through the cursor (result set) and create new customer
        // objects. Put them into the return list
        do{
            // UserExercises Table
            // (userExerciseID INTEGER PRIMARY KEY AUTOINCREMENT, userName
            TEXT,
            // exerciseID INTEGER, date TEXT, time INTEGER, rating INTEGER,
            repetition INTEGER)
            int userExerciseID = cursor.getInt(0);
            String userName = cursor.getString(1);
            int exerciseID = cursor.getInt(2);
            String date = cursor.getString(3);
            int time = cursor.getInt(4);
            int rating = cursor.getInt(5);
            int repetition = cursor.getInt(6);

            UserExercise newUserExercise = new UserExercise(userExerciseID,
            userName, exerciseID, date, time, rating, repetition);
            returnList.add(newUserExercise);

        } while (cursor.moveToNext());

    }else{
        // failure. do not add anything to the list.
    }

    // always clean up after yourself
    // lets close the connection to the database so others can use it
    // close the cursor when done.
    cursor.close();

    return returnList;
}

//method to search for userExercise by User ID
```

```

public ArrayList<UserExercise> getUserExercisesByUserName(String userName){
    //get all users from user table
    ArrayList<UserExercise> userExercises = this.getUserExercises();
    ArrayList<UserExercise> returnList = new ArrayList<>();

    // if there are no users
    if(userExercises.isEmpty())
        return null;

    for (int i=0; i<userExercises.size(); i++) {

        UserExercise userExercise = userExercises.get(i);

        if(userExercise.getUserName() == userName){
            returnList.add(userExercise);
        }
    }

    return returnList;
}

//method to search for userExercise by User ID
//that have been done by the user (by knowing that the date of doing the
exercise is not null)
public ArrayList<UserExercise> getUserExercisesDoneByUserName(String
userName){
    //get all users from user table
    ArrayList<UserExercise> userExercises = this.getUserExercises();
    ArrayList<UserExercise> returnList = new ArrayList<>();

    // if there are no users
    if(userExercises.isEmpty())
        return null;

    for (int i=0; i<userExercises.size(); i++) {

        UserExercise userExercise = userExercises.get(i);

        if(userExercise.getUserName().contentEquals(userName) &&
userExercise.getDate() != null){
            returnList.add(userExercise);
        }
    }

    if(returnList.isEmpty() == false){
        Log.d("Error","List is empty");
    }

    return returnList;
}

//method to search for userExercise by User ID
//that have not been done by the user yet (by knowing that the date of doing
the exercise is null)
public ArrayList<UserExercise> getUserExercisesNotDoneByUserName(String
userName){

```

```

//get all users from user table
ArrayList<UserExercise> userExercises = this.getUserExercises();
ArrayList<UserExercise> returnList = new ArrayList<>();

// if there are no users
if(userExercises.isEmpty())
    return null;

for (int i=0; i<userExercises.size(); i++) {

    UserExercise userExercise = userExercises.get(i);

    if(userExercise.getUserName() == userName && userExercise.getDate() ==
null){
        returnList.add(userExercise);
    }
}

return returnList;
}

//method to get all songs
public ArrayList<Song> getSongs() {

    ArrayList<Song> returnList = new ArrayList<>();

    // get data from the database
    String queryString = "SELECT * FROM " + TABLE_SONGS;

    // get a reference to the active database
    // getWritableDatabase - insert, update or delete records
    // getReadableDatabase - SELECT items from the database
    SQLiteDatabase db = this.getReadableDatabase();

    // Cursor is the result set from a SQL statement
    Cursor cursor = db.rawQuery(queryString, null);

    // moveToFirst returns a true if there were items selected
    if (cursor.moveToFirst()) {

        // Loop through the cursor (result set) and create new customer
        objects. Put them into the return list
        do {
            // Song Table
            // (songID INTEGER PRIMARY KEY, songName TEXT, songMP3 INTEGER)
            int songID = cursor.getInt(0);
            String songName = cursor.getString(1);
            int songMP3 = cursor.getInt(2);

            Song newSong = new Song(songID, songName, songMP3);
            returnList.add(newSong);

        } while (cursor.moveToNext());
    } else {
        // failure. do not add anything to the list.
    }
}

```

```
// always clean up after yourself  
// Lets close the connection to the database so others can use it  
// close the cursor when done.  
cursor.close();  
  
return returnList;  
}  
}
```

## EditUserActivity

```
package com.example.fitnote13022021;

import androidx.appcompat.app.AppCompatActivity;

import android.app.AlertDialog;
import android.app.DatePickerDialog;
import android.content.Intent;
import android.os.Bundle;
import android.text.Editable;
import android.text.TextWatcher;
import android.view.View;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.EditText;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.SeekBar;
import android.widget.TextView;
import android.widget.Toast;

import java.util.Calendar;

public class EditUserActivity extends AppCompatActivity implements TextWatcher,
View.OnClickListener {

    //Initialize variables
    DataBaseHelper DataBaseHelper;

    User userChosen;

    String activeUserName;

    int activeUserLevel;

    EditText etPasswordEditUser;

    TextView txtTitleEditUser, txtSeekBarWeightEditUser, txtSeekBarHeightEditUser;

    SeekBar seekBarWeightEditUser, seekBarHeightEditUser;

    DatePickerDialog datePickerDialog;
    Button btDatePickerButtonEditUser, btUpdateEditUser, btCancelEditUser,
btDeleteEditUser;

    RadioGroup radioGroupGenderEditUser, radioGroupUserLevelEditUser;
    RadioButton radioButtonGender, radioButtonUserLevel;

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

        //setting database
        DataBaseHelper = new DataBaseHelper(this);

        //Assign btDatePickerButtonEditUser
```

```

btDatePickerButtonEditUser =
findViewById(R.id.btDatePickerButtonEditUser);
//initialize date picker
initDatePicker();
btDatePickerButtonEditUser.setText(getTodaysDate());

Intent intent = getIntent();

String userNameChosen = intent.getStringExtra("userNameChosen");
activeUserName = intent.getStringExtra("activeUserName");
activeUserLevel = intent.getIntExtra("activeUserLevel", 0);

//getting chosen user from database
userChosen = DataBaseHelper.getUserByName(userNameChosen);

//Assign variables
etPasswordEditUser = findViewById(R.id.etPasswordEditUser);

txtTitleEditUser = findViewById(R.id.txtTitleEditUser);
txtSeekBarWeightEditUser = findViewById(R.id.txtSeekBarWeightEditUser);
txtSeekBarHeightEditUser = findViewById(R.id.txtSeekBarHeightEditUser);

seekBarWeightEditUser = findViewById(R.id.seekBarWeightEditUser);
seekBarHeightEditUser = findViewById(R.id.seekBarHeightEditUser);

btUpdateEditUser = findViewById(R.id.btUpdateEditUser);
btCancelEditUser = findViewById(R.id.btCancelEditUser);
btDeleteEditUser = findViewById(R.id.btDeleteEditUser);

radioGroupGenderEditUser = findViewById(R.id.radioGroupGenderEditUser);
radioGroupUserLevelEditUser =
findViewById(R.id.radioGroupUserLevelEditUser);

//setting title to show userNameChosen:
txtTitleEditUser.setText("Edit " + userChosen.getUserName() + " :");

//setting seekBar and txt to show progress from user chosen
seekBarWeightEditUser.setProgress(userChosen.getUserWeight());
txtSeekBarWeightEditUser.setText(""+userChosen.getUserWeight());
seekBarHeightEditUser.setProgress(userChosen.getUserHeight());
txtSeekBarHeightEditUser.setText(""+userChosen.getUserHeight());

seekBarWeightEditUser.setOnSeekBarChangeListener(new
SeekBar.OnSeekBarChangeListener() {
    @Override
    public void onProgressChanged(SeekBar seekBar, int progress, boolean
fromUser) {
        txtSeekBarWeightEditUser.setText(""+progress);
    }

    @Override
    public void onStartTrackingTouch(SeekBar seekBar) {

    }

    @Override
    public void onStopTrackingTouch(SeekBar seekBar) {

```

```

        }

    });

    seekBarHeightEditUser.setOnSeekBarChangeListener(new
SeekBar.OnSeekBarChangeListener() {
    @Override
    public void onProgressChanged(SeekBar seekBar, int progress, boolean
fromUser) {
        txtSeekBarHeightEditUser.setText(""+progress);
    }

    @Override
    public void onStartTrackingTouch(SeekBar seekBar) {

    }

    @Override
    public void onStopTrackingTouch(SeekBar seekBar) {

    }
});

//setting the edit text to show password
etPasswordEditUser.setText(userChosen.getUserPassword());

//setting textChange Listener
etPasswordEditUser.addTextChangedListener(this);

//setting click Listener
btUpdateEditUser.setOnClickListener(this);
btCancelEditUser.setOnClickListener(this);
btDeleteEditUser.setOnClickListener(this);

}

@Override
public void onClick(View v) {
    if(v == btUpdateEditUser)
    {
        //gets the id of the checked radio button in radio group
        int radioIdGender =
radioGroupGenderEditUser.getCheckedRadioButtonId();

        //sets radioButton variable to whatever is checked
        radioButtonGender = findViewById(radioIdGender);

        //sets String gender to be checked gender
        String gender = radioButtonGender.getText().toString();

        //gets the id of the checked radio button in radio group
        int radioIdUserLevel =
radioGroupUserLevelEditUser.getCheckedRadioButtonId();

        //sets radioButton variable to whatever is checked
        radioButtonUserLevel = findViewById(radioIdUserLevel);

        //sets String gender to be checked gender
        String userLevelTXT = radioButtonUserLevel.getText().toString();
    }
}
}

```

```

int chosenUserLevel = 0;

if (userLevelTXT.equals("Normal")){
    chosenUserLevel = 0;
}else if (userLevelTXT.equals("Admin")){
    chosenUserLevel = 1;
}else if (userLevelTXT.equals("Super Admin")){
    chosenUserLevel = 2;
}

/// User (username TEXT PRIMARY KEY, pass TEXT, birthYear INT, gender
TEXT)

// User Table:
// (userName TEXT PRIMARY KEY, userPassword TEXT, userWeight INTEGER,
// userHeight INTEGER, userBirthDate TEXT, userGender TEXT, profilePic
INTEGER)
String userName = userChosen.getUserName();
String userPassword = etPasswordEditUser.getText().toString();
Integer userWeight =
Integer.parseInt(txtSeekBarWeightEditUser.getText().toString());
Integer userHeight=
Integer.parseInt(txtSeekBarHeightEditUser.getText().toString());
String userBirthDate =
btDatePickerButtonEditUser.getText().toString();
String userGender = gender;
String profilePic = userName;

User user = new User(userName, userPassword, userWeight, userHeight,
userBirthDate, userGender,profilePic);

//putting user info in User table
if(chosenUserLevel == 0){
    User user1 = new User(userName, userPassword, userWeight,
userHeight, userBirthDate, userGender,profilePic);
    //update chosen user info
    DataBaseHelper.updateUser(user1);
}

//if user is not normal
if (chosenUserLevel == 1 || chosenUserLevel == 2){
    Admin user2;

    if (chosenUserLevel == 1)
        user2 = new Admin(userName, userPassword, false,userWeight,
userHeight, userBirthDate, userGender,profilePic);
    else
        user2 = new Admin(userName, userPassword, true,userWeight,
userHeight, userBirthDate, userGender,profilePic);

    //update chosen user info
    DataBaseHelper.updateUserAsAdmin(user2);
}

```

```
//going to ManagerUsersActivity
Intent intent = new Intent(this, ManagerUsersActivity.class);
intent.putExtra("activeAdminName", activeUserName);
intent.putExtra("activeAdminLevel", activeUserLevel);
startActivity(intent);
finish();
```

```
}
```

```
if(v == btCancelEditUser)
{
    //going to ManagerUsersActivity
    Intent intent = new Intent(this, ManagerUsersActivity.class);
    intent.putExtra("activeAdminName", activeUserName);
    intent.putExtra("activeAdminLevel", activeUserLevel);
    startActivity(intent);
    finish();
}
```

```
if (v == btDeleteEditUser){

    DataBaseHelper.deleteUser(userChosen);

    //going to ManagerUsersActivity
    Intent intent = new Intent(this, ManagerUsersActivity.class);
    intent.putExtra("activeAdminName", activeUserName);
    intent.putExtra("activeAdminLevel", activeUserLevel);
    startActivity(intent);
    finish();
}
```

```
}
```

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

```
}
```

```
@Override
public void onTextChanged(CharSequence s, int start, int before, int count) {
    //gets editTexts inputs, trim() -> removing spaces between strings
    String passwordInput = etPasswordEditUser.getText().toString().trim();

    boolean allWritten = !passwordInput.isEmpty();

    // true - only if all buttons have text
    btUpdateEditUser.setEnabled(allWritten);
}
```

```
@Override
public void afterTextChanged(Editable s) {
```

```
}
```

```
//initialize date picker
private void initDatePicker() {
```

```

//setting date listener
DatePickerDialog.OnDateSetListener dateSetListener = new
DatePickerDialog.OnDateSetListener()
{
    @Override
    public void onDateSet(DatePicker datePicker, int year, int month, int
day)
    {
        //adding 1 to month because its default is 0
        month = month + 1;
        String date = makeDateString(day, month, year);
        btDatePickerButtonEditUser.setText(date);

    }
};

Calendar cal = Calendar.getInstance();
//setting default date to be current date
int year = cal.get(Calendar.YEAR);
int month = cal.get(Calendar.MONTH);
int day = cal.get(Calendar.DAY_OF_MONTH);

int style = AlertDialog.THEME_HOLO_LIGHT;

// (context to datePickerDialog, style, date Listener, year, month, day)
datePickerDialog = new DatePickerDialog(this, style, dateSetListener,
year, month, day);
//sets max date to be today
datePickerDialog.getDatePicker().setMaxDate(System.currentTimeMillis());

}

//creates date as a string
private String makeDateString(int day, int month, int year) {
    return getMonthFormat(month) + " " + day + " " + year;
}

//makes the number month a real month (1 - JAN)
private String getMonthFormat(int month) {
    if(month == 1)
        return "JAN";
    if(month == 2)
        return "FEB";
    if(month == 3)
        return "MAR";
    if(month == 4)
        return "APR";
    if(month == 5)
        return "MAY";
    if(month == 6)
        return "JUN";
    if(month == 7)
        return "JUL";
    if(month == 8)
        return "AUG";
    if(month == 9)
        return "SEP";
    if(month == 10)
        return "OCT";
}

```

```
        return "OCT";
    if(month == 11)
        return "NOV";
    if(month == 12)
        return "DEC";

    //default should never happen
    return "JAN";
}

public void openDatePicker(View view)
{
    datePickerDialog.show();
}

//gets today's date
private String getTodaysDate() {

    Calendar cal = Calendar.getInstance();
    //setting default date to be current date
    int year = cal.get(Calendar.YEAR);
    int month = cal.get(Calendar.MONTH);
    //adding 1 to month because its defualt is 0
    month = month + 1;
    int day = cal.get(Calendar.DAY_OF_MONTH);
    return makeDateString(day, month, year);
}

//a method to go back to SettingsActivity when clicking back arrow in bottom
@Override
public void onBackPressed() {

    //going to ManagerUsersActivity
    Intent intent = new Intent(this, ManagerUsersActivity.class);
    intent.putExtra("activeAdminName", activeUserName);
    intent.putExtra("activeAdminLevel", activeUserLevel);
    startActivity(intent);
    finish();
}

}
```

## ExecuteExerciseActivity

```
package com.example.fitnote13022021;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.os.CountDownTimer;
import android.view.View;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import java.util.Calendar;

public class ExecuteExerciseActivity extends AppCompatActivity implements
View.OnClickListener {

    DataBaseHelper DataBaseHelper;
    ImageView btnFinish, btnPlay;
    public static TextView txtTime;
    ImageView imgExercise;
    public static boolean playTheTimer = true;
    int userExerciseID;
    String activeUserName;

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

        DataBaseHelper = new DataBaseHelper(this);

        //finding XML parts
        btnFinish = findViewById(R.id.btnFinish);
        btnPlay = findViewById(R.id.btnPlay);
        txtTime = findViewById(R.id.txtTime);
        imgExercise = findViewById(R.id.imgExercise);

        //refresh boolean for new screen
        playTheTimer = true;

        Intent intent = getIntent();

        userExerciseID = intent.getIntExtra("ExecuteUserExerciseID", 0);
        int exerciseID = intent.getIntExtra("ExecuteExerciseID", 0);
        activeUserName = intent.getStringExtra("activeUserName");

        Exercise exercise = DataBaseHelper.getExerciseByID(exerciseID);
```

```
imgExercise.setImageResource(exercise.getExercisePic());  
  
btnPlay.setOnClickListener(this);  
btnFinish.setOnClickListener(this);  
  
}  
  
//on button click  
@Override  
public void onClick(View v) {  
  
    if (v == btnPlay) {  
  
        //Playing the clock count and starting music  
        if (playTheTimer == true) {  
  
            Intent startService = new Intent(this,  
MusicAndTimerService.class);  
            startService.putExtra("ACTION", "PLAY");  
            startService(startService);  
  
            //setting the image to be pause button  
            btnPlay.setImageResource(R.drawable.pause_button);  
  
        }  
  
        //Pausing the clock count and the music  
        if (playTheTimer == false) {  
  
            Intent startService = new Intent(this,  
MusicAndTimerService.class);  
            startService.putExtra("ACTION", "PAUSE");  
            startService(startService);  
  
            //setting the image to be play button  
            btnPlay.setImageResource(R.drawable.play_button);  
  
        }  
  
        //Switching modes (Play/Pause) to switch between them  
        if (playTheTimer == true) {  
            playTheTimer = false;  
        } else {  
            playTheTimer = true;  
        }  
  
    }  
  
    if (v == btnFinish){  
  
        //Pausing the clock count  
        if(!playTheTimer){  
            btnPlay.setImageResource(R.drawable.play_button);  
        }  
  
        int countFromService = MusicAndTimerService.count;  
  
        //stop button to stop music service
```

```
Intent startService = new Intent(this, MusicAndTimerService.class);
stopService(startService);

//going to SignIn activity
Intent intent = new Intent(this, FeedbackActivity.class);

String date = getTodaysDate();

//adding extra in order to update the userExercise later
intent.putExtra("userExerciseID", userExerciseID);
intent.putExtra("countOfExercise", countFromService);
intent.putExtra("date", date);
intent.putExtra("activeUserName", activeUserName);

startActivity(intent);

//finishing the activity to go to onCreate when coming back
finish();
}

}

//a method to go back to ProgramUserActivity
@Override
public void onBackPressed() {
    super.onBackPressed();

    //stop button to stop music service
    Intent startService = new Intent(this, MusicAndTimerService.class);
    stopService(startService);

    //setting an intent to go back to the ProgramUserActivity after finishing
    //an exercise
    Intent intent = new Intent(ExecuteExerciseActivity.this,
    ProgramUserActivity.class);

    intent.putExtra("activeUserName", activeUserName);

    startActivity(intent);

    finish();
}

@Override
protected void onDestroy() {
    super.onDestroy();

    //stop button to stop music service
    Intent startService = new Intent(this, MusicAndTimerService.class);
    stopService(startService);

    finish();
}

//gets today's date
private String getTodaysDate() {
```

```
Calendar cal = Calendar.getInstance();
//setting default date to be current date
int year = cal.get(Calendar.YEAR);
int month = cal.get(Calendar.MONTH);
//adding 1 to month because its defualt is 0
month = month + 1;
int day = cal.get(Calendar.DAY_OF_MONTH);
return makeDateString(day, month, year);
}

//creates date as a string
private String makeDateString(int day, int month, int year) {
    return getMonthFormat(month) + " " + day + " " + year;
}

//makes the number month a real month (1 - JAN)
private String getMonthFormat(int month) {
    if(month == 1)
        return "JAN";
    if(month == 2)
        return "FEB";
    if(month == 3)
        return "MAR";
    if(month == 4)
        return "APR";
    if(month == 5)
        return "MAY";
    if(month == 6)
        return "JUN";
    if(month == 7)
        return "JUL";
    if(month == 8)
        return "AUG";
    if(month == 9)
        return "SEP";
    if(month == 10)
        return "OCT";
    if(month == 11)
        return "NOV";
    if(month == 12)
        return "DEC";

    //default should never happen
    return "JAN";
}
}
```

## Exercise

```
package com.example.fitnote13022021;

public class Exercise {

    // Exercise Table;
    // (exerciseID INT PRIMARY KEY, exerciseName TEXT,
    // exercisePic INT, exerciseDetail TEXT)

    private int exerciseID;
    private String exerciseName;
    private int exercisePic;
    private String exerciseDetail;

    //constructors
    public Exercise(int exerciseID, String exerciseName, int exercisePic, String
exerciseDetail) {
        this.exerciseID = exerciseID;
        this.exerciseName = exerciseName;
        this.exercisePic = exercisePic;
        this.exerciseDetail = exerciseDetail;
    }

    // toString is necessary for printing the contents of a class object
    @Override
    public String toString() {
        return "Exercise{" +
            "exerciseID=" + exerciseID +
            ", exerciseName='" + exerciseName + '\'' +
            ", exercisePic=" + exercisePic +
            ", exerciseDetail='" + exerciseDetail + '\'' +
            '}';
    }

    //Getters and Setters
    public int getExerciseID() {
        return exerciseID;
    }

    public void setExerciseID(int exerciseID) {
        this.exerciseID = exerciseID;
    }

    public String getExerciseName() {
        return exerciseName;
    }

    public void setExerciseName(String exerciseName) {
        this.exerciseName = exerciseName;
    }

    public int getExercisePic() {
        return exercisePic;
    }
}
```

```
public void setExercisePic(int exercisePic) {  
    this.exercisePic = exercisePic;  
}  
  
public String getExerciseDetail() {  
    return exerciseDetail;  
}  
  
public void setExerciseDetail(String exerciseDetail) {  
    this.exerciseDetail = exerciseDetail;  
}  
}
```

## ExerciseAdaptor

```
package com.example.fitnote13022021;

import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import java.util.ArrayList;

public class ExerciseAdaptor extends BaseAdapter {

    private ArrayList<Exercise> arrayList;
    private Context context;

    public ExerciseAdaptor(ArrayList<Exercise> arrayList, Context context) {
        this.arrayList = arrayList;
        this.context = context;
    }

    @Override
    public int getCount() {
        return arrayList.size();
    }

    @Override
    public Object getItem(int position) {
        return arrayList.get(position);
    }

    @Override
    public long getItemId(int position) {
        return 0;
    }

    @Override
    public View getView(int position, View convertView, ViewGroup parent) {

        Exercise exercise = arrayList.get(position);

        convertView =
LayoutInflater.from(context).inflate(R.layout.exercise_layout, null);

        TextView txtMainTitle = convertView.findViewById(R.id.txtMainTitle);
        ImageView imageView = convertView.findViewById(R.id.imageView);

        txtMainTitle.setText(exercise.getExerciseName());
        imageView.setImageResource(exercise.getExercisePic());

        return convertView;
    }
}
```

## FeedbackActivity

```
package com.example.fitnote13022021;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.NumberPicker;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.TextView;

public class FeedbackActivity extends AppCompatActivity {

    DataBaseHelper DataBaseHelper;
    int userExerciseID;
    int timeCount = 0;
    int repetition = 0;
    String date;

    TextView txtTestInput, txtFirstSentence, txtSecondSentence;
    NumberPicker numberPicker;
    Button btnFinish;
    RadioGroup radioGroup;
    RadioButton radioButton;

    String activeUserName;

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

        DataBaseHelper = new DataBaseHelper(this);

        //finding XML parts
        txtTestInput = findViewById(R.id.txtTestInput);
        txtFirstSentence = findViewById(R.id.txtFirstSentence);
        txtSecondSentence = findViewById(R.id.txtSecondSentence);
        numberPicker = findViewById(R.id.numberPicker);
        btnFinish = findViewById(R.id.btnFinish);
        radioGroup = findViewById(R.id.radioGroup);

        //setting txtInput
        txtTestInput.setText("Repetition: " + repetition);

        //getting the userExerciseId and seconds count from
        ExecuteExerciseActivity
        Intent intent = getIntent();

        userExerciseID = intent.getIntExtra("userExerciseID", 0);
```

```

timeCount = intent.getIntExtra("countOfExercise", 0);
date = intent.getStringExtra("date");
activeUserName = intent.getStringExtra("activeUserName");

numberPicker.setMinValue(0);
numberPicker.setMaxValue(150);

//setting a listener to repetition choose
numberPicker.setOnValueChangedListener(new
NumberPicker.OnValueChangeListener() {
    @Override
    public void onValueChange(NumberPicker picker, int oldVal, int newVal)
{
    repetition = newVal;
    txtTestInput.setText("Repetition: " + repetition);
}
});

//setting a listener to click on finishing reporting feedback
btnFinish.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {

        //gets the id of the checked radio button in radio group
        int radioId = radioGroup.getCheckedRadioButtonId();

        //sets radioButton variable to whatever is checked
        radioButton = findViewById(radioId);

        //sets String gender to be checked gender
        int rating = Integer.parseInt(radioButton.getText().toString());

        //method to update userExercise
        //changeUserExercise(int userExerciseID, String date, int time,
        int rating, int repetition)
        DataBaseHelper.updateUserExercise(userExerciseID, date, timeCount,
rating, repetition);

        //setting an intent to go back to the ProgramUserActivity after
finishing an exercise
        Intent intent = new Intent(FeedbackActivity.this,
ProgramUserActivity.class);

        intent.putExtra("activeUserName", activeUserName);

        startActivity(intent);

        finish();
    }
});

}

//a method to go back to ProgramUserActivity when clicking back arrow in
bottom
@Override

```

```
public void onBackPressed() {  
    Intent intent = new Intent(this, ProgramUserActivity.class);  
    intent.putExtra("activeUserName", activeUserName);  
    startActivity(intent);  
    finish();  
}  
}
```

## InformationActivity

```
package com.example.fitnote13022021;

import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;

import com.google.android.material.floatingactionbutton.FloatingActionButton;
import com.google.android.material.snackbar.Snackbar;

import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;

import android.view.View;
import android.widget.AdapterView;
import android.widget.ImageView;
import android.widget.ListView;
import android.widget.TextView;

import java.util.ArrayList;

public class InformationActivity extends AppCompatActivity implements
AdapterView.OnItemClickListener{

    //Initialize variables
    DataBaseHelper DataBaseHelper;

    ListView listViewExercisesInfo;

    ArrayList<Exercise> exercises;

    ExerciseAdaptor exerciseAdaptor;

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

        //setting database
        DataBaseHelper = new DataBaseHelper(this);

        //finding XML parts
        //Assign variables
        listViewExercisesInfo = findViewById(R.id.listViewExercisesInfo);

        exercises = DataBaseHelper.getExercises();

        exerciseAdaptor = new ExerciseAdaptor(exercises, this);

        listViewExercisesInfo.setAdapter(exerciseAdaptor);

        //settings onClick Listener
        listViewExercisesInfo.setOnItemClickListener(this);
    }
}
```

```

//method to show a specific exercise
@Override
public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

    //gets the clicked exercise
    Exercise chosenExercise = exercises.get(position);

    int exerciseID = chosenExercise.getExerciseID();

    AlertDialog.Builder builder = new AlertDialog.Builder(this);

    builder.setMessage(chosenExercise.getExerciseName() + ":");

    //Setting the builder's layout inflater
    View alertView =
getLayoutInflater().inflate(R.layout.exercise_with_description_layout, null);

    //getting views from layout
    ImageView exerciseImage =
alertView.findViewById(R.id.imageViewOfExerciseInDescription);
    TextView txtExerciseNameInDescription =
alertView.findViewById(R.id.txtExerciseNameInDescription);
    TextView txtExerciseDescription =
alertView.findViewById(R.id.txtExerciseDescription);

    //getting the whole chosen exercise information
    Exercise exerciseChosen = null;

    for (int i=0; i<exercises.size(); i++) {
        Exercise exercise = exercises.get(i);
        if(exercise.getExerciseID() == exerciseID){
            exerciseChosen = exercise;
        }
    }

    //Setting the imageView to show exercise image
    exerciseImage.setImageResource(exerciseChosen.getExercisePic());

    //Setting textViews to show exercise details
    txtExerciseNameInDescription.setText(exerciseChosen.getExerciseName());
    txtExerciseDescription.setText(exerciseChosen.getExerciseDetail());


    //Setting the view to be in builder of alert dialog
    builder.setView(alertView);

    builder.setPositiveButton("Got it!", new DialogInterface.OnClickListener()
{
    public void onClick(DialogInterface dialog, int id) {

        dialog.cancel();

    }
});
    //builder.setNegativeButton("Cancel", new
DialogInterface.OnClickListener() {
        // public void onClick(DialogInterface dialog, int id) {

```

```
// dialog.cancel();
// }
// });

//Creating and showing the dialog
AlertDialog dialog = builder.create();

dialog.show();

}

//a method to go back to MainScreenActivity when clicking back arrow in bottom
@Override
public void onBackPressed() {

    Intent intent = new Intent(this, MainScreenActivity.class);

    startActivity(intent);

    finish();

}

}
```

## MainScreenActivity

```
package com.example.fitnote13022021;

import androidx.appcompat.app.AppCompatActivity;

import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class MainScreenActivity extends AppCompatActivity implements
View.OnClickListener {

    //Initialize variables
    DataBaseHelper DataBaseHelper;

    EditText etUsername, etPassword;

    Button btRegister, btLogIn;

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

        //setting database
        DataBaseHelper = new DataBaseHelper(this);

        //toolbar = findViewById(R.id.toolbar);
        //setSupportActionBar(toolbar);
        //https://www.youtube.com/watch?v=DMkzIOLppf4
        //3:50
        //** https://www.youtube.com/watch?v=IrJ8Hzuz2LU

        //finding XML parts
        //Assign variables
        etUsername = findViewById(R.id.etUserName);
        etPassword = findViewById(R.id.etPassword);
        btLogIn = findViewById(R.id.btLogIn);
        btRegister = findViewById(R.id.btRegister);

        //setting click listener
        btLogIn.setOnClickListener(this);
        btRegister.setOnClickListener(this);

        //adding exercises (one time)
        //dataBaseHelper.deleteDatabase(this);
        //dataBaseHelper.addExercises();

    }
}
```

```

@Override
public void onClick(View v) {
    // User Table:
    // (userName TEXT PRIMARY KEY, userPassword TEXT,
    // userWeight INTEGER, userHeight INTEGER, userBirthDate TEXT,
    // userGender TEXT)
    String userName = etUsername.getText().toString();
    String userPassword = etPassword.getText().toString();

    if (v == btRegister) {

        //clear text on editTexts
        etUsername.setText("");
        etPassword.setText("");

        //going to SignIn activity
        Intent intent = new Intent(this, RegisterActivity.class);
        startActivity(intent);
        finish();
    }

    if (v == btLogIn) {

        if (dataBaseHelper.searchUserByNameAndPass(userName, userPassword)) {
            Toast.makeText(this, "Welcome: " + userName,
Toast.LENGTH_SHORT).show();

            //clear text on editTexts
            etUsername.setText("");
            etPassword.setText("");

            //going to programs activity
            Intent intent = new Intent(this, ProgramUserActivity.class);
            intent.putExtra("activeUserName", userName);
            startActivity(intent);
            finish();
        } else {
            Toast.makeText(this, "No user with that info",
Toast.LENGTH_SHORT).show();
        }
    }

    //a method that sets an AlertDialog if the user wants to leave
    @Override
    public void onBackPressed() {
        AlertDialog.Builder builder = new AlertDialog.Builder(this);

        builder.setMessage("Do you want to leave? ");
        builder.setPositiveButton("OK", new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int id) {
                MainScreenActivity.super.onBackPressed();
            }
        });
        builder.setNegativeButton("Cancel", new DialogInterface.OnClickListener()
{
            public void onClick(DialogInterface dialog, int id) {
                dialog.cancel();
            }
        });
    }
}

```

```

        }

    });

    //Creating and showing the dialog
    AlertDialog dialog = builder.create();

    dialog.show();
}

//on create menu method (inflating) the Layout in menu
@Override
public boolean onCreateOptionsMenu(Menu menu) {

    getMenuInflater().inflate(R.menu.main_menu, menu);

    return true;
}

//on item click of menu
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    int id = item.getItemId();

    if (id == R.id.aboutItem){
        //going to InformationActivity activity
        Intent intent = new Intent(this, InformationActivity.class);

        startActivity(intent);

        finish();
    }
    //shows dialog if user wants to exit
    if (id == R.id.exitItem){
        AlertDialog.Builder builder = new AlertDialog.Builder(this);

        builder.setMessage("Do you want to leave? ");
        builder.setPositiveButton("OK", new DialogInterface.OnClickListener()
{
            public void onClick(DialogInterface dialog, int id) {
                MainScreenActivity.super.onBackPressed();
            }
        });
        builder.setNegativeButton("Cancel", new
DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int id) {
                dialog.cancel();
            }
        });
    }
    //Creating and showing the dialog
    AlertDialog dialog = builder.create();

    dialog.show();
}

return true;}}

```

## ManagerUsersActivity

```
package com.example.fitnote13022021;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ListView;
import android.widget.SearchView;
import android.widget.TextView;
import android.widget.Toast;

import java.util.ArrayList;
import java.util.Locale;

public class ManagerUsersActivity extends AppCompatActivity {

    //Initialize variables
    DataBaseHelper DataBaseHelper;

    String activeUserName;

    User activeUser;

    TextView txtWelcomeManageUsers, txtExplainManageUser;

    ListView listViewManagerUsers;

    SearchView searchViewManageUsersList;

    ArrayList<User> users;

    ArrayList<User> filteredUsers;

    UsersAdapter usersAdaptor;

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

        //setting database
        DataBaseHelper = new DataBaseHelper(this);

        Intent intent = getIntent();

        activeUserName = intent.getStringExtra("activeAdminName");

        //getting user from database
        activeUser = DataBaseHelper.getUserByName(activeUserName);

        //Assign variables
        txtWelcomeManageUsers = findViewById(R.id.txtWelcomeManageUsers);
        txtExplainManageUser = findViewById(R.id.txtExplainManageUser);
```

```

listViewManagerUsers = findViewById(R.id.listViewManagerUsers);

searchViewManageUsersList = findViewById(R.id.searchViewManageUsersList);

users = DataBaseHelper.getUsersAndAdmins();

filteredUsers = users;

usersAdaptor = new UsersAdapter(users, this, activeUserName,
activeUser.getUserLevel());

listViewManagerUsers.setAdapter(usersAdaptor);

//initialize the searchWidget in order to filter exercises
initSearchWidgets();

String userLevel = "Admin";

if (activeUser.getUserLevel() == 2){
    userLevel = "Super-Admin";
}

Toast.makeText(this, activeUserName + " Level: " +
activeUser.getUserLevel(), Toast.LENGTH_LONG).show();

//settings text to say hello userName
txtWelcomeManageUsers.setText("Welcome " + activeUserName + " a known " +
userLevel);

//settings click listener to list
listViewManagerUsers.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int
position, long id) {

        String userNameChosen = users.get(position).getUserName();

        //avoid user changing his own info and prevent non SUPER ADMIN
people to do it
        if(!userNameChosen.equals(activeUserName) &&
activeUser.getUserLevel() == 2){
            //going to EditUserActivity
            Intent intent = new Intent(ManagerUsersActivity.this,
EditUserActivity.class);
            intent.putExtra("activeUserName", activeUserName);
            intent.putExtra("activeUserLevel", activeUser.getUserLevel());
            intent.putExtra("userNameChosen",
users.get(position).getUserName());
            startActivity(intent);
        }else {
            if(userNameChosen.equals(activeUserName))
                Toast.makeText(ManagerUsersActivity.this, "YOU CANT EDIT YOURSELF!!!!!!",
Toast.LENGTH_SHORT).show();
        else if (activeUser.getUserLevel() != 2)
            Toast.makeText(ManagerUsersActivity.this, "Your'e level doesn't allow you to
edit users", Toast.LENGTH_SHORT).show();
    }
}

```

```

        }

    });

}

//initialize the searchWidget in order to filter exercises
public void initSearchWidgets(){

    SearchView searchView = (SearchView)searchViewManageUsersList;

    searchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() {
        @Override
        public boolean onQueryTextSubmit(String query) {
            return false;
        }

        //this method is called every time a user
        //puts in any character into the search view
        //literally any chang :D
        @Override
        public boolean onQueryTextChange(String newText) {

            filteredUsers = new ArrayList<User>();

            //same as a regular for lop
            for (User user: users)
            {

                int userLevel = user.getUserLevel();
                String txtLevel = "Normal";
                if (userLevel == 1){
                    txtLevel = "Admin";
                }else if(userLevel == 2){
                    txtLevel = "Super-Admin";
                }

                //if the user's name has one of the letters in the written
                text

                if(user.getUserName().toLowerCase().contains(newText.toLowerCase(Locale.ROOT))){
                    filteredUsers.add(user);
                }

                //if the user's password has one of the letters in the written
                text
                else
                if(user.getPassword().toLowerCase().contains(newText.toLowerCase(Locale.ROOT)))
                {
                    filteredUsers.add(user);
                }

                //if the user's weight has one of the letters in the written
                text
                else
                if(Integer.toString(user.getUserWeight()).contains(newText.toLowerCase(Locale.ROOT

```

```

        ))){
            filteredUsers.add(user);
        }

        //if the user's height has one of the letters in the written
text
        else
if(Integer.toString(user.getUserHeight()).contains(newText.toLowerCase(Locale.ROOT
))){
            filteredUsers.add(user);
        }

        //if the user's birth date has one of the letters in the
written text
        else
if(user.getUserBirthDate().toLowerCase().contains(newText.toLowerCase(Locale.ROOT)
)){
            filteredUsers.add(user);
        }

        //if the user's gender has one of the letters in the written
text
        else
if(user.getUserGender().toLowerCase().contains(newText.toLowerCase(Locale.ROOT))){
            filteredUsers.add(user);
        }

        //if the user's Level has one of the letters in the written
text
        else
if(txtLevel.toLowerCase().contains(newText.toLowerCase(Locale.ROOT))){
            filteredUsers.add(user);
        }

    }

    UsersAdapter usersAdaptorFiltered = new
UsersAdapter(filteredUsers, ManagerUsersActivity.this, activeUserName,
activeUser.getUserLevel());

    listViewManagerUsers.setAdapter(usersAdaptorFiltered);

    return false;
};

});;
});

//a method to go back to SettingsActivity when clicking back arrow in bottom
@Override
public void onBackPressed() {

    Intent intent = new Intent(this, SettingsActivity.class);

    intent.putExtra("activeUserName", activeUserName);

    startActivity(intent);
}

```

```
    finish();  
}  
}
```

**MusicAndTimerService**

```

package com.example.fitnote13022021;

import android.app.Service;
import android.content.Intent;
import android.media.MediaPlayer;
import android.os.CountDownTimer;
import android.os.IBinder;
import android.widget.Toast;

import androidx.annotation.Nullable;

import java.util.ArrayList;
import java.util.Random;

public class MusicAndTimerService extends Service {

    private DataBaseHelper DataBaseHelper;
    private MediaPlayer mediaPlayer;
    private CountDownTimer countDownTimer;
    public static int count = 0;

    //this method does binds the service
    //with and activity
    הפעילות עם השירות ואן קונשנרט
    @Nullable
    @Override
    public IBinder onBind(Intent intent) {
        return null;
    }

    @Override
    public void onCreate() {
        super.onCreate();
        DataBaseHelper = new DataBaseHelper(this);
        ArrayList<Song> songs = DataBaseHelper.getSongs();

        int arrayLength = songs.size();

        Random random = new Random();
        //This gives a random integer number between 0 (inclusive) and (Length of
        array-1)
        int randomNumber = random.nextInt(arrayLength);
        Song chosenSong = songs.get(randomNumber);

        Toast.makeText(this, "chosen song: " + chosenSong.getSongName() + "N: " +
randomNumber, Toast.LENGTH_SHORT).show();

        //create the mediaPlayer
        mediaPlayer = MediaPlayer.create(this, chosenSong.getSongMP3());

        mediaPlayer.setLooping(true);

        if(countDownTimer != null){
    
```

```

        countDownTimer.cancel();
        countDownTimer = null;
    }

    //setting up the timer
    //max: 2147483347millis = 35791.389116667minutes = 596.523151944449978
hours, sec to change: 1000
    countDownTimer = new CountDownTimer(Integer.MAX_VALUE - 300, 1000) {
        @Override
        public void onTick(long millisUntilFinished) {
            count++;
            //setting text in play to be time count from timer in
MusicAndTimerService
            ExecuteExerciseActivity.txtTime.setText(getDurationString(count));
        }

        @Override
        public void onFinish() {
            count = 0;
            ExecuteExerciseActivity.txtTime.setText(getDurationString(0));
        }
    };
}

//when we start the service the onStartCommand method will be called
@Override
public int onStartCommand(Intent intent, int flags, int startId) {

    String action = intent.getStringExtra("ACTION");
    if(action.contentEquals("PLAY")){
        Toast.makeText(this, "PLAY", Toast.LENGTH_SHORT).show();
        //play music and timer
        mediaPlayer.start();
        countDownTimer.start();
    }else if (action.contentEquals("PAUSE")){
        Toast.makeText(this, "PAUSE", Toast.LENGTH_SHORT).show();
        //pause music and timer
        mediaPlayer.pause();
        countDownTimer.cancel();
    }

    // this means this service will be explicitly started and stopped
    // במנוע שווי פסיק וונען זה ישירות הדבר פירש
    return START_STICKY;
}

//when the service is stopped onDestroy method will be called
@Override
public void onDestroy() {
    super.onDestroy();

    //stop music and timer
    mediaPlayer.stop();
    countDownTimer.cancel();
    count = 0;
    ExecuteExerciseActivity.txtTime.setText(getDurationString(0));
}

```

```
}

private String getDurationString(int seconds) {

    int hours = seconds / 3600;
    int minutes = (seconds % 3600) / 60;
    seconds = seconds % 60;

    return twoDigitString(hours) + " : " + twoDigitString(minutes) + " : " +
twoDigitString(seconds);
}

private String twoDigitString(int number) {

    if (number == 0) {
        return "00";
    }

    if (number / 10 == 0) {
        return "0" + number;
    }

    return String.valueOf(number);
}

}
```

**PictureFileHelper**

```

package com.example.fitnote13022021;

import android.app.Activity;
import android.content.Context;
import android.content.SharedPreferences;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.util.ArrayList;

public class PictureFileHelper {

    public static void writeFileToInternalStorage(Context context, Bitmap bitmap,
String filename)
    {
        SharedPreferences sp= context.getSharedPreferences("info",0);
        int counter=sp.getInt("counter", 0);
        try {
            FileOutputStream os =
((Activity)context).openFileOutput(filename+counter, Context.MODE_PRIVATE);
            //Here compress 50%, store the compressed data in os.
            bitmap.compress(Bitmap.CompressFormat.PNG,100,os);
            counter++;
            SharedPreferences.Editor editor=sp.edit();
            editor.putInt("counter",counter);
            editor.commit();
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
    }

    public static Bitmap readFileFromInternalStorage(Context context,String
filename)
    {
        Bitmap b=null;
        try {
            InputStream in = ((Activity)context).openFileInput(filename);
            b= BitmapFactory.decodeStream(in);
            in.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return b;
    }

    public static Bitmap getPic(Context context, String name)
    {
        File mydir = context.getFilesDir();
        File lister = mydir.getAbsoluteFile();
        Bitmap bitmap=null;
    }
}

```

```
for (String list : lister.list())
{
    if(list.toString().contains(name)) {
        //Toast.makeText(context, list, Toast.LENGTH_LONG).show();
        bitmap = readFromFileInternalStorage(context, list);

    }
}

return bitmap;
}
```

## ProgramUserActivity

```
package com.example.fitnote13022021;

import androidx.appcompat.app.AppCompatActivity;

import android.app.AlarmManager;
import android.app.NotificationChannel;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.app.TimePickerDialog;
import android.content.Intent;
import android.graphics.Bitmap;
import android.os.Build;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.ListView;
import android.widget.SearchView;
import android.widget.TextView;
import android.widget.TimePicker;
import android.widget.Toast;

import java.util.ArrayList;
import java.util.Calendar;
import java.util.Locale;

public class ProgramUserActivity extends AppCompatActivity implements
View.OnClickListener {

    //Initialize variables
    public static String activeUserName = null;

    DataBaseHelper DataBaseHelper;
    Button btnSetAlarm, btnCancelAlarm;
    ImageView imgProgramUserPic;
    TextView txtMainTitle, txtEmptyListMessage, tv_timer;
    ListView listViewUserExercises;
    ArrayList<UserExercisesInnerJoinEx> activeUsersExercises;
    UserExerciseNotDoneAdapter userExerciseAdaptor;
    ImageView btnAddExercise;
    SearchView searchViewUserExerciseList;
    private int tvHour=0, tvMinute=0;

    @Override
```

```

protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_program_user);

    //creating the notification channel (if needed)
    createNotificationChannel();

    DataBaseHelper = new DataBaseHelper(this);

    //finding XML parts / Assign variables
    imgProgramUserPic = findViewById(R.id.imgProgramUserPic);
    listViewUserExercises = findViewById(R.id.listViewUserExercises);
    txtMainTitle = findViewById(R.id.txtMainTitle);
    txtEmptyListMessage = findViewById(R.id.txtEmptyListMessage);
    tv_timer = findViewById(R.id.tv_timer);
    btnAddExercise = findViewById(R.id.btnAddExercise);
    searchViewUserExerciseList =
    findViewById(R.id.searchViewUserExerciseList);
    btnSetAlarm = findViewById(R.id.btnSetAlarm);
    btnCancelAlarm = findViewById(R.id.btnCancelAlarm);

    Intent intent = getIntent();

    //getting the active userName from getting the user who entered with Log
    in with his name given from the intent
    String userName = intent.getStringExtra("activeUserName");

    activeUserName = userName;

    //changing textView to show userName's name as well
    String welcomeText = txtMainTitle.getText().toString();
    txtMainTitle.setText(welcomeText + " " + activeUserName);

    //changing the userPicture to show the user's picture

    Bitmap userPic = null;

    if(userName != null)
        userPic = PictureFileHelper.getPic(this, userName);

    if(userPic != null && imgProgramUserPic != null){
        imgProgramUserPic.setImageBitmap(userPic);
    }

    activeUsersExercises =
    DataBaseHelper.getExercisesByUserNameGivenNotDone(activeUserName);

    //setting up a new adaptor
    userExerciseAdaptor = new UserExerciseNotDoneAdapter(activeUsersExercises,
    this, activeUserName);

    listViewUserExercises.setAdapter(userExerciseAdaptor);

    //setting the txtEmptyListMessage visible if the ListView is empty
    if(activeUsersExercises.isEmpty()){
        txtEmptyListMessage.setVisibility(View.VISIBLE);
    }
}

```

```
        }

        //goes to ExecuteExerciseActivity
        listViewUserExercises.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int
position, long id) {

        //Toast.makeText(ProgramUserActivity.this, "1",
        Toast.LENGTH_SHORT).show();

    }
});

//initialize the searchWidget in order to filter exercises
initSearchWidgets();

btnAddExercise.setOnClickListener(this);

tv_timer.setOnClickListener(this);

btnSetAlarm.setOnClickListener(this);

btnCancelAlarm.setOnClickListener(this);

}

@Override
public void onClick(View v) {

    //if the user wants to add an exercise
    if(v == btnAddExercise){

        //going to SignIn activity
        Intent intent = new Intent(this, AddExerciseActivity.class);

        //adding extra
        intent.putExtra("activeUserName", activeUserName);

        startActivity(intent);

        //finishing the activity to go to onCreate when coming back
        finish();
    }

    //if the user want to set himself a timer
    if(v == tv_timer){
        //Initialize time picked dialog
        TimePickerDialog timePickerDialog = new TimePickerDialog(
            ProgramUserActivity.this,
            new TimePickerDialog.OnTimeSetListener() {
                @Override
                public void onTimeSet(TimePicker view, int hourOfDay, int
minute) {
                    //Initialize hour and minute

```

```

tvHour = hourOfDay;
tvMinute = minute;
//Initialize calendar
Calendar calendar = Calendar.getInstance();
//Set hour and minute
calendar.set(0,0,0,tvHour,tvMinute);
//Set selected time on text view

tv_timer.setText(android.text.format.DateFormat.format("h:mm a", calendar));
}
},12,0,true
);
//Displayed previous selected time
timePickerDialog.updateTime(tvHour,tvMinute);
//Show dialog
timePickerDialog.show();
}

if(v == btnSetAlarm || v == btnCancelAlarm){

//Set notificationID & message
Intent notificationIntent = new Intent(
    ProgramUserActivity.this, ReminderBroadcast.class);

notificationIntent.putExtra("message", "Its time to train " +
activeUserName);
notificationIntent.putExtra("activeUserName", activeUserName);

//PendingIntent
PendingIntent pendingIntent = PendingIntent.getBroadcast(
    ProgramUserActivity.this, 0, notificationIntent, 0
);
//FLAG_CANCEL_CURRENT-if pending already exists, the current one will
be canceled.
//FLAG_UPDATE_CURRENT-indicates that the pendingIntent
//which we created now can be updated in the future

//AlarmManager
//Use with getSystemService to retrieve an AlarmManager
//for receiving intents at a time of your choosing
AlarmManager alarmManager = (AlarmManager)
getSystemService(ALARM_SERVICE);

switch (v.getId()){
case R.id.btnSetAlarm:
    // Set Alarm - we got hour and minutes from: tvHour,tvMinute

    // Create time
    Calendar startTime = Calendar.getInstance();
    startTime.set(Calendar.HOUR_OF_DAY, tvHour);
    startTime.set(Calendar.MINUTE, tvMinute);
    startTime.set(Calendar.SECOND, 1);
    long alarmStartTime = startTime.getTimeInMillis();

    //checking that the time given is not before now
    if (alarmStartTime <= System.currentTimeMillis()){


```

```

        Toast.makeText(this, "This time is before now! : " +
startTime.getTime(), Toast.LENGTH_LONG).show();

    }else{
        // Set Alarm
        //AlarmManager.RTC_WAKEUP - wakes up the device to fire
        //at the specified time
        alarmManager.set(AlarmManager.RTC_WAKEUP, alarmStartTime,
pendingIntent);

        Toast.makeText(this, "Set Done! : " + startTime.getTime(),
Toast.LENGTH_LONG).show();
    }

    break;

case R.id.btnCancelAlarm:
    // Cancel Alarm
    alarmManager.cancel(pendingIntent);
    Toast.makeText(this, "Canceled", Toast.LENGTH_SHORT).show();
    break;
}

}

private void createNotificationChannel(){
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O){
        CharSequence name = "LembuitReminderChannel";
        String description = "Channel for Lembuit Remider";
        int importance = NotificationManager.IMPORTANCE_DEFAULT;
        NotificationChannel channel = new NotificationChannel("notifyLemubit",
name, importance);
        channel.setDescription(description);

        NotificationManager notificationManager =
getSystemService(NotificationManager.class);
        notificationManager.createNotificationChannel(channel);
    }
}

//initialize the searchWidget in order to filter exercises
public void initSearchWidgets(){

    SearchView searchView = (SearchView)searchViewUserExerciseList;

    searchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() {
        @Override
        public boolean onQueryTextSubmit(String query) {
            return false;
        }

        //this method is called every time a user
        //puts in any character into the search view
        //literally any change :D
    });
}

```

```

@Override
public boolean onQueryTextChange(String newText) {

    ArrayList<UserExercisesInnerJoinEx> filteredExercises = new
ArrayList<UserExercisesInnerJoinEx>();

    //same as a regular for loop
    for (UserExercisesInnerJoinEx userExercisesInnerJoinEx:
activeUsersExercises)
    {
        //if an exercise has one of the letters in the written text

if(userExercisesInnerJoinEx.getExerciseName().toLowerCase().contains(newText.toLowerCase(Locale.ROOT))){
            filteredExercises.add(userExercisesInnerJoinEx);
        }
    }

    UserExerciseNotDoneAdapter userExerciseAdaptor = new
UserExerciseNotDoneAdapter(filteredExercises, ProgramUserActivity.this);

    listViewUserExercises.setAdapter(userExerciseAdaptor);

    return false;
};

});

}

//on create menu method (inflating) the Layout in menu
@Override
public boolean onCreateOptionsMenu(Menu menu) {

    getMenuInflater().inflate(R.menu.program_user_menu, menu);

    return true;
}

//on item click of menu
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    int id = item.getItemId();

    //if the user wants to see graphs of his performances
    if (id == R.id.graphsItem){

        //going to StatisticsActivity activity
        Intent intent = new Intent(this, StatisticsActivity.class);

        //adding extra
        intent.putExtra("activeUserName", activeUserName);

        startActivity(intent);

        //finishing the activity to go to onCreate when coming back
        finish();
    }
}

```

```
//if the user wants to change his settings
if (id == R.id.settingsItem){
    //going to SettingsActivity activity
    Intent intent = new Intent(this, SettingsActivity.class);

    //adding extra
    intent.putExtra("activeUserName", activeUserName);

    startActivity(intent);

    //finishing the activity to go to onCreate when coming back
    finish();
}

//if the user wants to share his data with others
if(id == R.id.shareItem){

    //going to ShareActivity
    Intent intent = new Intent(this, ShareActivity.class);
    intent.putExtra("activeUserName", activeUserName);
    startActivity(intent);
    finish();

}

//if the user wants to see his results on a list
if (id == R.id.exercisesResultsItem){
    //going to ViewExercisesResultsActivity
    Intent intent = new Intent(this, ViewExercisesResultsActivity.class);
    intent.putExtra("activeUserName", activeUserName);
    startActivity(intent);
    finish();
}

return true;
}

//a method to go back to SettingsActivity when clicking back arrow in bottom
@Override
public void onBackPressed() {

    //going to MainScreenActivity
    Intent intent = new Intent(this, MainScreenActivity.class);
    startActivity(intent);
    finish();

}
}
```

## RegisterActivity

```
package com.example.fitnote13022021;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

import android.Manifest;
import android.app.Activity;
import android.app.AlertDialog;
import android.app.DatePickerDialog;
import android.content.ContextWrapper;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.drawable.BitmapDrawable;
import android.graphics.drawable.Drawable;
import android.media.MediaPlayer;
import android.os.Bundle;
import android.os.Environment;
import android.provider.MediaStore;
import android.text.Editable;
import android.text.TextWatcher;
import android.view.View;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.SeekBar;
import android.widget.TextView;
import android.widget.Toast;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;
import java.util.Calendar;

public class RegisterActivity extends AppCompatActivity implements TextWatcher,
View.OnClickListener {

    DataBaseHelper DataBaseHelper;

    ImageView imgProfilePic, imgCemraButton;

    Bitmap bitmapToSave;

    private static final int REQUEST_IMAGE_CAPTURE = 100;

    EditText etUsername, etPassword;
```

```
TextView txtSeekBarWeight, txtSeekBarHeight;
SeekBar seekBarWeight, seekBarHeight;

DatePickerDialog datePickerDialog;
Button dateButton;

Button btFinishReg, btCancel;

RadioGroup radioGroup;
RadioButton radioButton;

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

    //initialize DataBaseHelper
    DataBaseHelper = new DataBaseHelper(this);

    //initialize date picker
    initDatePicker();
    dateButton = findViewById(R.id.btDatePickerButton);
    dateButton.setText(getTodaysDate());

    //finding XML parts
    imgProfilePic = findViewById(R.id.imgProfilePic);
    imgCemraButton = findViewById(R.id.imgCemraButton);
    etUsername = findViewById(R.id.etUserName);
    etPassword = findViewById(R.id.etPassword);

    txtSeekBarWeight = findViewById(R.id.txtSeekBarWeight);
    txtSeekBarHeight = findViewById(R.id.txtSeekBarHeight);

    seekBarWeight = findViewById(R.id.seekBarWeight);
    seekBarHeight = findViewById(R.id.seekBarHeight);

    radioGroup = findViewById(R.id.radioGroup);

    btFinishReg = findViewById(R.id.btFinishReg);
    btCancel = findViewById(R.id.btCancel);

    //setting a tag to the imgProfilePic to be the default image:
    imgProfilePic.setTag(R.drawable.default_user);

    seekBarWeight.setOnSeekBarChangeListener(new
SeekBar.OnSeekBarChangeListener() {
        @Override
        public void onProgressChanged(SeekBar seekBar, int progress, boolean
fromUser) {
            txtSeekBarWeight.setText(""+progress);
        }

        @Override
        public void onStartTrackingTouch(SeekBar seekBar) {

        }

        @Override
    });
}
```

```

        public void onStopTrackingTouch(SeekBar seekBar) {
    }

});}

seekBarHeight.setOnSeekBarChangeListener(new
SeekBar.OnSeekBarChangeListener() {
    @Override
    public void onProgressChanged(SeekBar seekBar, int progress, boolean
fromUser) {
        txtSeekBarHeight.setText(""+progress);
    }

    @Override
    public void onStartTrackingTouch(SeekBar seekBar) {
    }

    @Override
    public void onStopTrackingTouch(SeekBar seekBar) {
    }
});

//setting textChange listener
etUsername.addTextChangedListener(this);
etPassword.addTextChangedListener(this);

//setting click listener
btFinishReg.setOnClickListener(this);
btCancel.setOnClickListener(this);
imgCemraButton.setOnClickListener(this);

}

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

@Override
public void onTextChanged(CharSequence s, int start, int before, int count) {
    //gets editTexts inputs, trim() -> removing spaces between strings
    String usernameInput = etUsername.getText().toString().trim();
    String passwordInput = etPassword.getText().toString().trim();

    boolean allWritten = !usernameInput.isEmpty() && !passwordInput.isEmpty();

    // true - only if all buttons have text
    btFinishReg.setEnabled(allWritten);
}

@Override
public void afterTextChanged(Editable s) {
}

```

```

@Override
public void onClick(View v) {
    if(v == btFinishReg)
    {
        //gets the id of the checked radio button in radio group
        int radioId = radioGroup.getCheckedRadioButtonId();

        //sets radioButton variable to whatever is checked
        radioButton = findViewById(radioId);

        //sets String gender to be checked gender
        String gender = radioButton.getText().toString();

        /// User (username TEXT PRIMARY KEY, pass TEXT, birthYear INT, gender
TEXT)

        // User Table:
        // (userName TEXT PRIMARY KEY, userPassword TEXT, userWeight INTEGER,
        // userHeight INTEGER, userBirthDate TEXT, userGender TEXT, profilePic
INTEGER)
        String userName = etUsername.getText().toString();
        String userPassword = etPassword.getText().toString();
        Integer userWeight =
Integer.parseInt(txtSeekBarWeight.getText().toString());
        Integer userHeight=
Integer.parseInt(txtSeekBarHeight.getText().toString());
        String userBirthDate = dateButton.getText().toString();
        String userGender = gender;
        String profilePic = userName;

        //putting user info in User table
        User user = new User(userName, userPassword, userWeight, userHeight,
userBirthDate, userGender,profilePic);

        //method to search for user with the same userName (true - user exists
/ false - user doesn't exist)
        if(dataBaseHelper.searchUserByName(user.getUserName()) == false) {
            dataBaseHelper.insertUser(user);

            saveToInternalStorage(userName);

            Toast.makeText(this, user + " is inserted",
Toast.LENGTH_SHORT).show();

            //going to MainScreenActivity
            Intent intent = new Intent(this, MainScreenActivity.class);
            startActivity(intent);
            finish();
        } else{
            Toast.makeText(this, "user with the name " + user.getUserName() +
" already exists", Toast.LENGTH_SHORT).show();
        }
    }
}

```

```

if(v == btCancel)
{
    Toast.makeText(this, "Canceling...", Toast.LENGTH_SHORT).show();

    //going to MainScreenActivity
    Intent intent = new Intent(this, MainScreenActivity.class);
    startActivity(intent);
    finish();
}

if(v == imgCemraButton)
{
    takePicture(imgCemraButton);
}

//method to start another application using and intent object
public void takePicture(View view){

    //the intents intention is to capture an image
    //we have to specify the action for the intent
    Intent imageTakeIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);

    //now we are able to check if there is any application
    //capable of handling this intent
    //otherwise your'e application will crush
    if(imageTakeIntent.resolveActivity(getApplicationContext()) != null)
    {
        //we have to user startActivityForResult method
        //second parameter is the requestCode
        startActivityForResult(imageTakeIntent, REQUEST_IMAGE_CAPTURE);
    }
}

//in order to receive the result from the other application we
//have to override a method called onActivityResult
@Override
protected void onActivityResult(int requestCode, int resultCode, @Nullable
Intent data) {

    //we have to check if the requestCode is the same
    //AND if the requestCode is ok
    super.onActivityResult(requestCode, resultCode, data);
    if (requestCode == REQUEST_IMAGE_CAPTURE && resultCode == RESULT_OK) {
        //here we have the data available on this intent
        //now we can retrieve the data
        Bundle extras = data.getExtras();
        Bitmap imageBitmap = (Bitmap) extras.get("data");

        //now we can display the image on the image view
        imgProfilePic.setImageBitmap(imageBitmap);

        //getting the image to save
}

```

```
        bitmapToSave = imageBitmap;
    }

}

//save image to phone
private void saveToInternalStorage(String userName){
    //bitmapToSave

    if(bitmapToSave != null && userName != null){
        PictureFileHelper.writeFileToInternalStorage(this, bitmapToSave,
userName);
    }else {
        //default pic
        Bitmap bitmap_default_user =
BitmapFactory.decodeResource(getResources(),R.drawable.default_user);
        PictureFileHelper.writeFileToInternalStorage(this,
bitmap_default_user, userName);
    }
}

//gets today's date
private String getTodaysDate() {

    Calendar cal = Calendar.getInstance();
    //setting default date to be current date
    int year = cal.get(Calendar.YEAR);
    int month = cal.get(Calendar.MONTH);
    //adding 1 to month because its defualt is 0
    month = month + 1;
    int day = cal.get(Calendar.DAY_OF_MONTH);
    return makeDateString(day, month, year);
}

//initialize date picker
private void initDatePicker() {
    //setting date Listener
    DatePickerDialog.OnDateSetListener dateSetListener = new
DatePickerDialog.OnDateSetListener()
{
    @Override
    public void onDateSet(DatePicker datePicker, int year, int month, int
day)
    {
        //adding 1 to month because its default is 0
        month = month + 1;
        String date = makeDateString(day, month, year);
        dateButton.setText(date);

    }
};

Calendar cal = Calendar.getInstance();
//setting default date to be current date
int year = cal.get(Calendar.YEAR);
```

```

int month = cal.get(Calendar.MONTH);
int day = cal.get(Calendar.DAY_OF_MONTH);

int style = AlertDialog.THEME_HOLO_LIGHT;

// (context to datePickerDialog, style, date Listener, year, month, day)
datePickerDialog = new DatePickerDialog(this, style, dateSetListener,
year, month, day);
//sets max date to be today
datePickerDialog.getDatePicker().setMaxDate(System.currentTimeMillis());

}

//creates date as a string
private String makeDateString(int day, int month, int year) {
    return getMonthFormat(month) + " " + day + " " + year;
}

//makes the number month a real month (1 - JAN)
private String getMonthFormat(int month) {
    if(month == 1)
        return "JAN";
    if(month == 2)
        return "FEB";
    if(month == 3)
        return "MAR";
    if(month == 4)
        return "APR";
    if(month == 5)
        return "MAY";
    if(month == 6)
        return "JUN";
    if(month == 7)
        return "JUL";
    if(month == 8)
        return "AUG";
    if(month == 9)
        return "SEP";
    if(month == 10)
        return "OCT";
    if(month == 11)
        return "NOV";
    if(month == 12)
        return "DEC";

    //default should never happen
    return "JAN";
}

public void openDatePicker(View view)
{
    datePickerDialog.show();
}

@Override
public void onBackPressed() {
    //going to MainScreenActivity
    Intent intent = new Intent(this, MainScreenActivity.class);
}

```

```
        startActivity(intent);
        finish();
    }
}
```

## ReminderBroadcast

```
package com.example.fitnote13022021;

import android.app.PendingIntent;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.util.Log;

import androidx.core.app.NotificationCompat;
import androidx.core.app.NotificationManagerCompat;

import java.util.concurrent.atomic.AtomicInteger;

public class ReminderBroadcast extends BroadcastReceiver {
    @Override
    public void onReceive(Context context, Intent intent) {

        Log.d("RECEIVED", "Notification received!" + " UserName" +
        intent.getStringExtra("activeUserName"));

        // Get id & message from intent.
        String message = intent.getStringExtra("message");
        String activeUserName = intent.getStringExtra("activeUserName");

        // Call ProgramUserActivity when notification is tapped.
        Intent programUserAcIntent = new Intent(context,
        ProgramUserActivity.class);

        //we need to add this userName in order to take the user to his
        //ProgramUserActivity screen
        programUserAcIntent.putExtra("activeUserName", activeUserName);

        //Intent to go to user's exercise list when clicking on notification
        PendingIntent contentIntent = PendingIntent.getActivity(
            context, 0, programUserAcIntent, 0
        );

        NotificationCompat.Builder builder = new
        NotificationCompat.Builder(context, "notifyLemubit")
            .setSmallIcon(android.R.drawable.ic_dialog_info)
            .setContentTitle("Exercise Time!")
            .setContentText(message)
            .setContentIntent(contentIntent)
            .setAutoCancel(true)
            .setPriority(NotificationCompat.PRIORITY_DEFAULT);

        NotificationManagerCompat notificationManager =
        NotificationManagerCompat.from(context);

        //According to the official documentation
        //starting in Android 8.0(API Level 26)
        //all notifications must be assigned to a channel

        //the notificationID is a unique int for each notification that must be
        defined
        notificationManager.notify(NotificationID.getID(), builder.build());
    }
}
```

```
    }

    public static class NotificationID {
        private final static AtomicInteger c = new AtomicInteger(0);
        public static int getID() {
            return c.incrementAndGet();
        }
    }

}
```

## SettingsActivity

```
package com.example.fitnote13022021;

import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;

import android.app.AlertDialog;
import android.app.DatePickerDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Color;
import android.os.Bundle;
import android.provider.MediaStore;
import android.text.Layout;
import android.view.View;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.SeekBar;
import android.widget.TextView;
import android.widget.Toast;

import java.util.Calendar;

public class SettingsActivity extends AppCompatActivity implements
View.OnClickListener{

    //Initialize variables
    DataBaseHelper DataBaseHelper;

    String activeUserName;

    int activeUserLevel;

    User activeUser;

    ImageView imgSettingUserPic;

    TextView txtSettingsUserNameShow, txtSettingsUserGenderShow
        , txtSettingsPassword, txtSettingsWeightHeight
        , txtSettingsGenderEdit, txtSettingsPictureEdit
        ,txtAdminOPTitleSettings;

    View layoutSettingsPassword, layoutSettingsWeightAndHeight
        ,layoutSettingBirthDate, layoutSettingsGender, layoutSettingPic
        ,layoutSettingManagerUsers;

    DatePickerDialog datePickerDialog;
    Button dateButton;

    private static final int REQUEST_IMAGE_CAPTURE = 100;
    Bitmap bitmapToSave;
```

```

ImageView imgSettingsEditProfilePic;

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

    //setting database
    DataBaseHelper = new DataBaseHelper(this);

    Intent intent = getIntent();

    activeUserName = intent.getStringExtra("activeUserName");

    activeUser = DataBaseHelper.getUserByName(activeUserName);

    //Assign variables
    imgSettingUserPic = findViewById(R.id.imgSettingUserPic);

    txtSettingsUserNameShow = findViewById(R.id.txtSettingsUserNameShow);
    txtSettingsUserGenderShow = findViewById(R.id.txtSettingsUserGenderShow);
    txtSettingsPassword = findViewById(R.id.txtSettingsPassword);
    txtSettingsWeightHeight = findViewById(R.id.txtSettingsWeightHeight);
    txtSettingsGenderEdit = findViewById(R.id.txtSettingsGenderEdit);
    txtSettingsPictureEdit = findViewById(R.id.txtSettingsPictureEdit);
    txtAdminOPTitleSettings = findViewById(R.id.txtAdminOPTitleSettings);

    layoutSettingsPassword = findViewById(R.id.LayoutSettingsPassword);
    layoutSettingsWeightAndHeight =
    findViewById(R.id.LayoutSettingsWeightAndHeight);
    layoutSettingBirthDate = findViewById(R.id.LayoutSettingBirthDate);
    layoutSettingsGender = findViewById(R.id.LayoutSettingsGender);
    layoutSettingPic = findViewById(R.id.layoutSettingPic);
    layoutSettingManagerUsers = findViewById(R.id.layoutSettingManagerUsers);

    //Setting the userPic to be the activeUser's pic
    //changing the userPicture to show the user's picture
    Bitmap userPic = PictureFileHelper.getPic(this, activeUserName);
    if(userPic != null && imgSettingUserPic != null){
        imgSettingUserPic.setImageBitmap(userPic);
    }

    //changing LayoutSettingManagerUsers to show if user is admin at Least (1
    or 2 in userLevel)
    activeUserLevel = DataBaseHelper.getUserLevelByUserName(activeUserName);

    //if user is admin at Least, show LayoutSettingManagerUsers:
    if (activeUserLevel == 1 || activeUserLevel == 2){
        layoutSettingManagerUsers.setVisibility(View.VISIBLE);

        //txt: option: manage users
        String textFromAdminTitle =
        txtAdminOPTitleSettings.getText().toString();

        if(activeUserLevel == 1){
            txtAdminOPTitleSettings.setText("Admin " + textFromAdminTitle);
        }else {
            txtAdminOPTitleSettings.setText("Super-Admin " +

```

```

textFromAdminTitle);
}

//Setting text to be user's info from database
txtSettingsUserNameShow.setText(activeUser.getUserName());
txtSettingsUserGenderShow.setText(activeUser.getUserGender());
txtSettingsPassword.setText(activeUser.getUserPassword());
txtSettingsWeightHeight.setText("Weight: " + activeUser.getUserWeight() +
" Height: " + activeUser.getUserHeight());
txtSettingsGenderEdit.setText(activeUser.getUserGender());

layoutSettingsPassword.setOnClickListener(this);
layoutSettingsWeightAndHeight.setOnClickListener(this);
layoutSettingBirthDate.setOnClickListener(this);
layoutSettingsGender.setOnClickListener(this);
layoutSettingPic.setOnClickListener(this);
layoutSettingManagerUsers.setOnClickListener(this);

}

@Override
public void onClick(View v) {

//If user wants to change his password
if (v == layoutSettingsPassword){
    AlertDialog.Builder builder = new AlertDialog.Builder(this);

    builder.setMessage("Do you want to change the password? ");

    //Setting the builder's editTexts
    final EditText passwordInput = new EditText(SettingsActivity.this);
    passwordInput.setText(activeUser.getUserPassword());

    //Setting the builder's editTexts in the builder
    builder.setView(passwordInput);

    builder.setPositiveButton("Yes", new DialogInterface.OnClickListener()
{
    public void onClick(DialogInterface dialog, int id) {

        //Setting the updatedUser
        User updatedUser = activeUser;

        //Setting the password to be the new one

updatedUser.setUserPassword(passwordInput.getText().toString());

        DataBaseHelper.updateUser(updatedUser);

        activeUser = updatedUser;

        txtSettingsPassword.setText(activeUser.getUserPassword());

    }
});
builder.setNegativeButton("Cancel", new

```

```

DialogInterface.OnClickListener() {
    public void onClick(DialogInterface dialog, int id) {
        dialog.cancel();
    }
});

//Creating and showing the dialog
AlertDialog dialog = builder.create();

dialog.show();
}

//If user wants to change his weight and/or height
if (v == layoutSettingsWeightAndHeight){
    AlertDialog.Builder builder = new AlertDialog.Builder(this);

    builder.setMessage("Do you want to change the weight and/or height?
");

    //Setting the builder's Layout inflater
    View alertView =
getLayoutInflater().inflate(R.layout.weight_height_settings_alert, null);

    //Setting the view to be in builder of alert dialog
    builder.setView(alertView);

    //Getting the xml parts from alert view
    SeekBar seekBarSettingsWeight, seekBarSettingsHeight;
    TextView txtSeekBarSettingsWeight, txtSeekBarSettingsHeight;

    seekBarSettingsWeight =
alertView.findViewById(R.id.seekBarSettingsWeight);
    seekBarSettingsHeight =
alertView.findViewById(R.id.seekBarSettingsHeight);
    txtSeekBarSettingsWeight =
alertView.findViewById(R.id.txtSeekBarSettingsWeight);
    txtSeekBarSettingsHeight =
alertView.findViewById(R.id.txtSeekBarSettingsHeight);

    //Setting for both seekers change listeners
    seekBarSettingsWeight.setOnSeekBarChangeListener(new
SeekBar.OnSeekBarChangeListener() {
        @Override
        public void onProgressChanged(SeekBar seekBar, int progress,
boolean fromUser) {
            txtSeekBarSettingsWeight.setText(""+progress);
        }

        @Override
        public void onStartTrackingTouch(SeekBar seekBar) {

        }

        @Override
        public void onStopTrackingTouch(SeekBar seekBar) {
    }
});
}

```

```

seekBarSettingsHeight.setOnSeekBarChangeListener(new
SeekBar.OnSeekBarChangeListener() {
    @Override
    public void onProgressChanged(SeekBar seekBar, int progress,
boolean fromUser) {
        txtSeekBarSettingsHeight.setText(""+progress);
    }

    @Override
    public void onStartTrackingTouch(SeekBar seekBar) {

    }

    @Override
    public void onStopTrackingTouch(SeekBar seekBar) {

}
});
```

builder.setPositiveButton("Yes", new DialogInterface.OnClickListener()
{
 public void onClick(DialogInterface dialog, int id) {

 //Setting the updatedUser
 User updatedUser = activeUser;

 //Setting the weight and/or height to be the new one
updatedUser.setUserWeight(Integer.parseInt(txtSeekBarSettingsWeight.getText().toString()));

updatedUser.setUserHeight(Integer.parseInt(txtSeekBarSettingsHeight.getText().toString()));

 DataBaseHelper.updateUser(updatedUser);

 activeUser = updatedUser;

 txtSettingsWeightHeight.setText("Weight: " +
activeUser.getUserWeight() + " Height: " + activeUser.getUserHeight());

 }
});

builder.setNegativeButton("Cancel", new
DialogInterface.OnClickListener() {
 public void onClick(DialogInterface dialog, int id) {
 dialog.cancel();
 }
});

//Creating and showing the dialog
AlertDialog dialog = builder.create();

dialog.show();
}

```
//If user wants to change his birth date
if (v == layoutSettingBirthDate){
    AlertDialog.Builder builder = new AlertDialog.Builder(this);

    builder.setMessage("Do you want to change your birth date? ");

    //Setting the builder's datePickerDialog
    //initialize date picker
    dateButton = new Button(SettingsActivity.this);

    dateButton.setWidth(250);
    dateButton.setHeight(200);
    dateButton.setText("JAN 01 2020");
    dateButton.setTextColor(Color.BLACK);
    dateButton.setTextSize(30);

//dateButton.setScrollBarStyle(Integer.parseInt("?android:spinnerStyle"));
    dateButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            datePickerDialog.show();
        }
    });
}

initDatePicker();
dateButton.setText(getTodaysDate());

//Setting the builder's editTexts in the builder
builder.setView(dateButton);

builder.setPositiveButton("Yes", new DialogInterface.OnClickListener()
{
    public void onClick(DialogInterface dialog, int id) {

        //Setting the updatedUser
        User updatedUser = activeUser;

        //Setting the birth date to be the new one
        updatedUser.setUserBirthDate(dateButton.getText().toString());

        DataBaseHelper.updateUser(updatedUser);

        activeUser = updatedUser;

    }
});

builder.setNegativeButton("Cancel", new
DialogInterface.OnClickListener() {
    public void onClick(DialogInterface dialog, int id) {
        dialog.cancel();
    }
});

//Creating and showing the dialog
AlertDialog dialog = builder.create();

dialog.show();
}
```

```
//If user wants to change his gender
if (v == layoutSettingsGender){
    AlertDialog.Builder builder = new AlertDialog.Builder(this);

    builder.setMessage("Do you want to change your gender? ");

    //Setting the builder's Layout inflater
    View alertView =
getLayoutInflater().inflate(R.layout.gender_settings_alert, null);

    //Setting the view to be in builder of alert dialog
    builder.setView(alertView);

    builder.setPositiveButton("Yes", new DialogInterface.OnClickListener()
{
    public void onClick(DialogInterface dialog, int id) {

        // Setting the updatedUser
        User updatedUser = activeUser;

        // Setting the gender to be the new one
        RadioButton radioButton;

        final RadioGroup radioGroupSettingsGender =
(RadioGroup)alertView.findViewById(R.id.radioGroupSettingsGender);

        // get selected radioButton from radioGroup //
        radioGroupSettingsGender.getCheckedRadioButtonId()
        int selectedId =
radioGroupSettingsGender.getCheckedRadioButtonId();

        // find the radioButton by returned id
        radioButton = (RadioButton)
alertView.findViewById(selectedId);

        // radioButton text
        String radiovalue = String.valueOf(radioButton.getText());

        updatedUser.setUserGender(radiovalue);

        DataBaseHelper.updateUser(updatedUser);

        activeUser = updatedUser;

        txtSettingsGenderEdit.setText(activeUser.getUserGender());
        txtSettingsUserGenderShow.setText(activeUser.getUserGender());

    }
});

    builder.setNegativeButton("Cancel", new
DialogInterface.OnClickListener() {
    public void onClick(DialogInterface dialog, int id) {
        dialog.cancel();
    }
});
}
```

```
//Creating and showing the dialog
AlertDialog dialog = builder.create();

        dialog.show();
    }

//If user wants to change his profile pic
if (v == layoutSettingPic) {
    AlertDialog.Builder builder = new AlertDialog.Builder(this);

    builder.setMessage("Do you want to change your profile pic? ");

    //Setting the builder's Layout inflater
    View alertView =
getLayoutInflater().inflate(R.layout.profilepic_settings_alert, null);

    //Setting the view to be in builder of alert dialog
    builder.setView(alertView);

    //Setting the user picture to be the new one
    imgSettingsEditProfilePic =
alertView.findViewById(R.id.imgSettingsEditProfilePic);
    final ImageView imgSettingsEditCemraButton =
alertView.findViewById(R.id.imgSettingsEditCemraButton);

    imgSettingsEditCemraButton.setOnClickListener(new
View.OnClickListener() {
        @Override
        public void onClick(View v) {
            takePicture(imgSettingsEditCemraButton);
        }
    });
}

builder.setPositiveButton("Yes", new DialogInterface.OnClickListener()
{
    public void onClick(DialogInterface dialog, int id) {

        // Setting the updatedUser
        User updatedUser = activeUser;

        updatedUser.setProfilePic(activeUser.getUserName());

        DataBaseHelper.updateUser(updatedUser);

        activeUser = updatedUser;

        // Setting the imageView of user pic
        // from the settings Layout to show new pic
        imgSettingUserPic.setImageBitmap(bitmapToSave);

        //save image to internal storage
        saveToInternalStorage(activeUser.getUserName());

    }
});
builder.setNegativeButton("Cancel", new
DialogInterface.OnClickListener() {
    public void onClick(DialogInterface dialog, int id) {
```

```

        dialog.cancel();
    }
});

//Creating and showing the dialog
AlertDialog dialog = builder.create();

dialog.show();
}

//if the user(admin/superAdmin) wants to go to manager users
if (v == layoutSettingManagerUsers){

    //going to ManagerUsersActivity
    Intent intent = new Intent(this, ManagerUsersActivity.class);
    intent.putExtra("activeAdminName", activeUser.getUserName());
    intent.putExtra("activeAdminLevel", activeUserLevel);
    startActivity(intent);
}

//gets today's date
private String getTodaysDate() {

    Calendar cal = Calendar.getInstance();
    //setting default date to be current date
    int year = cal.get(Calendar.YEAR);
    int month = cal.get(Calendar.MONTH);
    //adding 1 to month because its defualt is 0
    month = month + 1;
    int day = cal.get(Calendar.DAY_OF_MONTH);
    return makeDateString(day, month, year);
}

//initialize date picker
private void initDatePicker() {
    //setting date listener
    DatePickerDialog.OnDateSetListener dateSetListener = new
DatePickerDialog.OnDateSetListener()
{
    @Override
    public void onDateSet(DatePicker datePicker, int year, int month, int
day)
    {
        //adding 1 to month because its default is 0
        month = month + 1;
        String date = makeDateString(day, month, year);
        dateButton.setText(date);

    }
};

Calendar cal = Calendar.getInstance();
//setting default date to be current date
int year = cal.get(Calendar.YEAR);
int month = cal.get(Calendar.MONTH);
int day = cal.get(Calendar.DAY_OF_MONTH);

```

```

int style = AlertDialog.THEME_HOLO_LIGHT;

// (context to datePickerDialog, style, date Listener, year, month, day)
datePickerDialog = new DatePickerDialog(this, style, dateSetListener,
year, month, day);
//sets max date to be today
datePickerDialog.getDatePicker().setMaxDate(System.currentTimeMillis());

}

//creates date as a string
private String makeDateString(int day, int month, int year) {
    return getMonthFormat(month) + " " + day + " " + year;
}

//makes the number month a real month (1 - JAN)
private String getMonthFormat(int month) {
    if(month == 1)
        return "JAN";
    if(month == 2)
        return "FEB";
    if(month == 3)
        return "MAR";
    if(month == 4)
        return "APR";
    if(month == 5)
        return "MAY";
    if(month == 6)
        return "JUN";
    if(month == 7)
        return "JUL";
    if(month == 8)
        return "AUG";
    if(month == 9)
        return "SEP";
    if(month == 10)
        return "OCT";
    if(month == 11)
        return "NOV";
    if(month == 12)
        return "DEC";

    //default should never happen
    return "JAN";
}

//method to start another application using and intent object
public void takePicture(View view){

    //the intents intention is to capture an image
    //we have to specify the action for the intent
    Intent imageTakeIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);

    //now we are able to check if there is any application
    //capable of handling this intent
    //otherwise your'e application will crush
    if(imageTakeIntent.resolveActivity(getApplicationContext()) != null)

```

```

    {
        //we have to user startActivityForResult method
        //second parameter is the requestCode
        startActivityForResult(imageTakeIntent, REQUEST_IMAGE_CAPTURE);
    }

}

//in order to receive the result from the other application we
//have to override a method called
@Override
protected void onActivityResult(int requestCode, int resultCode, @Nullable
Intent data) {

    //we have to check if the requestCode is the same
    //AND if the requestCode is ok
    super.onActivityResult(requestCode, resultCode, data);
    if (requestCode == REQUEST_IMAGE_CAPTURE && resultCode == RESULT_OK) {
        //here we have the data available on this intent
        //now we can retrieve the data
        Bundle extras = data.getExtras();
        Bitmap imageBitmap = (Bitmap) extras.get("data");

        //now we can display the image on the image view
        //imgSettingsEditProfilePic is the profile pic on the alert
        //and on the imageView from the settings layout
        imgSettingsEditProfilePic.setImageBitmap(imageBitmap);

        //getting the image to save
        bitmapToSave = imageBitmap;
    }
}

//save image to phone
private void saveToInternalStorage(String userName){
    //bitmapToSave

    if(bitmapToSave != null){
        PictureFileHelper.writeFileToInternalStorage(this, bitmapToSave,
userName);
    }else {
        //default pic
        Bitmap bitmap_default_user =
        BitmapFactory.decodeResource(getResources(), R.drawable.default_user);
        PictureFileHelper.writeFileToInternalStorage(this,
        bitmap_default_user, userName);
    }
}

//a method to go back to ProgramUserActivity when clicking back arrow in
bottom
@Override
public void onBackPressed() {

```

```
Intent intent = new Intent(this, ProgramUserActivity.class);
intent.putExtra("activeUserName", activeUserName);
startActivity(intent);
finish();
}
```

## ShareActivity

```
package com.example.fitnote13022021;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import android.Manifest;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.database.Cursor;
import android.net.Uri;
import android.os.Bundle;
import android.provider.ContactsContract;
import android.view.View;
import android.widget.SearchView;
import android.widget.TextView;
import android.widget.Toast;

import java.util.ArrayList;

public class ShareActivity extends AppCompatActivity {

    //Initialize variable
    String activeUserName;

    TextView txtShareWithContactScreen;

    RecyclerView recyclerViewContacts;

    ArrayList<ContactModel> contactList = new ArrayList<ContactModel>();

    ArrayList<ContactModel> filteredContacts;

    ContactListAdapter adapter;

    ContactListAdapter.RecycleViewClickListener listener;

    SearchView searchViewContactList;

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

        Intent intent = getIntent();

        activeUserName = intent.getStringExtra("activeUserName");

        //Assign variables
        txtShareWithContactScreen = findViewById(R.id.txtShareWithContactScreen);
        recyclerViewContacts = findViewById(R.id.recyclerViewContacts);
        searchViewContactList = findViewById(R.id.searchViewContactList);
```

```

//edit TextView
String text = txtShareWithContactScreen.getText().toString();
txtShareWithContactScreen.setText(text + " " + activeUserName);

//Check permission
checkPermission();

//initialize the searchWidget in order to filter exercises
initSearchWidgets();

}

private void checkPermission() {
    //Check condition
    if (ContextCompat.checkSelfPermission(ShareActivity.this
            , Manifest.permission.READ_CONTACTS)
        != PackageManager.PERMISSION_GRANTED){
        //When permission is not granted
        //Request permission
        ActivityCompat.requestPermissions(ShareActivity.this
                , new String[]{Manifest.permission.READ_CONTACTS}, 100);
    }else {
        //When permission is granted
        //Create method
        getContactList();
    }
}

private void getContactList() {
    //Initialize uri
    //uniform resource identifier
    Uri uri = ContactsContract.Contacts.CONTENT_URI;

    //Sort by ascending (ASC)
    String sort = ContactsContract.CommonDataKinds.Phone.DISPLAY_NAME+" ASC";

    //Initialize cursos
    Cursor cursor = getContentResolver().query(
        uri, null, null, null, sort
    );

    //Check condition
    if(cursor.getCount() > 0){
        //When count is greater than 0
        //Use while loop
        while (cursor.moveToNext()){
            //Cursor move to next
            //Get contact id
            String id = cursor.getString(cursor.getColumnIndex(
                ContactsContract.Contacts._ID
            ));
            //Get contact name
            String name = cursor.getString(cursor.getColumnIndex(
                ContactsContract.Contacts.DISPLAY_NAME
            ));
            //Initialize phone uri
            Uri uriPhone = ContactsContract.CommonDataKinds.Phone.CONTENT_URI;

```

```

        //Initialize selection
        String selection =
ContactsContract.CommonDataKinds.Phone.CONTACT_ID
        +"=?";
        //Initialize phone cursor
        Cursor phoneCursor = getContentResolver().query(
            uriPhone, null, selection
            , new String[]{id}, null
        );
        //Check condition
        if (phoneCursor.moveToFirst()) {
            //When phone cursor move to next
            String number =
phoneCursor.getString(phoneCursor.getColumnIndex(
                ContactsContract.CommonDataKinds.Phone.NUMBER
));
            //Initialize contact model
            ContactModel model = new ContactModel();
            //Set name
            model.setName(name);
            //Set number
            if(number.charAt(0) == '0')
            {
                number = "+972 "+number.substring(1);
            }
            model.setNumber(number);
            //Add model in array list
            contactList.add(model);
            //Close number cursor
            phoneCursor.close();
        }
    }
    //Close cursor
    cursor.close();
}

//setting filtered list to be contactList
filteredContacts = contactList;

//Set Layout manager
recyclerViewContacts.setLayoutManager(new LinearLayoutManager(this));

//setOnItemClickListener
setOnItemClickListener();

//Initialize adapter
adapter = new ContactListAdapter(this, contactList, listener);
//Set adapter
recyclerViewContacts.setAdapter(adapter);
}

private void setOnItemClickListener() {
    listener = new ContactListAdapter.RecycleViewClickListener() {
        @Override
        public void onClick(View v, int position) {
            //now we are creating an intent to go to the WhatsAppSendActivity
            Intent intent = new Intent(getApplicationContext(),
WhatsAppSendActivity.class);

```

```

//now we want to get the contact ID, name and number:
ContactModel contactChosen = filteredContacts.get(position);
String name = contactChosen.getName();
String number = contactChosen.getNumber();

intent.putExtra("contactName", name);
intent.putExtra("contactNumber", number);
intent.putExtra("activeUserName", activeUserName);

startActivity(intent);

finish();

}

};

}

@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults) {
    super.onRequestPermissionsResult(requestCode, permissions, grantResults);
    //Check condition
    if (requestCode == 100 & grantResults.length > 0 && grantResults[0]
    == PackageManager.PERMISSION_GRANTED){
        //When permission is granted
        //Call method
        getContactList();
    }else {
        //When permission is denied
        //Display toast
        Toast.makeText(this, "Permission Denied.", Toast.LENGTH_SHORT).show();
        //Call check permission method
        checkPermission();
    }
}

//initialize the searchWidget in order to filter contacts
public void initSearchWidgets(){

    SearchView searchView = (SearchView)searchViewContactList;

    searchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() {
        @Override
        public boolean onQueryTextSubmit(String query) {
            return false;
        }

        //this method is called every time a user
        //puts in any character into the search view
        //literally any chang :D
        @Override
        public boolean onQueryTextChange(String newText) {

            filteredContacts = new ArrayList<ContactModel>();

            //same as a regular for loop
            for (ContactModel contactModel: contactList)

```

```
{  
    //if a contact has one of the letter in the written text  
    //OR a contact has one of the numbers in the written number  
  
    if(contactModel.getName().toLowerCase().contains(newText.toLowerCase())  
        || contactModel.getNumber().contains(newText)){  
        filteredContacts.add(contactModel);  
    }  
}  
  
ContactListAdapter contactListAdapter = new  
ContactListAdapter(ShareActivity.this, filteredContacts, listener);  
  
recyclerViewContacts.setAdapter(contactListAdapter);  
  
return false;  
};  
});  
}  
  
//a method to go back to ProgramUserActivity when clicking back arrow in  
bottom  
@Override  
public void onBackPressed() {  
  
    Intent intent = new Intent(this, ProgramUserActivity.class);  
  
    intent.putExtra("activeUserName", activeUserName);  
  
    startActivity(intent);  
  
    finish();  
}  
}
```

**Song**

```

package com.example.fitnote13022021;

public class Song {

    //Song (songID INTEGER PRIMARY KEY, songName TEXT, songMP3 INTEGER)

    private int songID;
    private String songName;
    private int songMP3;

    //constructors
    public Song(int songID, String songName, int songMP3) {
        this.songID = songID;
        this.songName = songName;
        this.songMP3 = songMP3;
    }

    // toString is necessary for printing the contents of a class object
    @Override
    public String toString() {
        return "Song{" +
            "songID=" + songID +
            ", songName='" + songName + '\'' +
            ", songMP3=" + songMP3 +
            '}';
    }

    //Getters and Setters
    public int getSongID() {
        return songID;
    }

    public void setSongID(int songID) {
        this.songID = songID;
    }

    public String getSongName() {
        return songName;
    }

    public void setSongName(String songName) {
        this.songName = songName;
    }

    public int getSongMP3() {
        return songMP3;
    }

    public void setSongMP3(int songMP3) {
        this.songMP3 = songMP3;
    }
}

```

## StatisticsActivity

```
package com.example.fitnote13022021;

import androidx.appcompat.app.AppCompatActivity;
import androidx.core.content.ContextCompat;

import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.Color;
import android.os.Bundle;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.Spinner;
import android.widget.Toast;

import com.github.mikephil.charting.charts.BarChart;
import com.github.mikephil.charting.components.AxisBase;
import com.github.mikephil.charting.components.Description;
import com.github.mikephil.charting.components.XAxis;
import com.github.mikephil.charting.components.YAxis;
import com.github.mikephil.charting.data.BarData;
import com.github.mikephil.charting.data.BarDataSet;
import com.github.mikephil.charting.data.BarEntry;
import com.github.mikephil.charting.data.BubbleEntry;
import com.github.mikephil.charting.data.CandleEntry;
import com.github.mikephil.charting.data.Entry;
import com.github.mikephil.charting.data.PieEntry;
import com.github.mikephil.charting.data.RadarEntry;
import com.github.mikephil.charting.formatter.DefaultAxisValueFormatter;
import com.github.mikephil.charting.formatter.IAxisValueFormatter;
import com.github.mikephil.charting.formatter.IndexAxisValueFormatter;
import com.github.mikephil.charting.formatter.ValueFormatter;
import com.github.mikephil.charting.utils.ColorTemplate;

import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.Date;
import java.util.Locale;
import java.util.TimeZone;

public class StatisticsActivity extends AppCompatActivity {

    DataBaseHelper DataBaseHelper;
    ImageView imgUserPicture;
    //Objects to choose exercise in graph
    Spinner exercise_spinner;
    ArrayList<Exercise> exercises;
```

```
ArrayList<String> exerciseNames;  
Spinner year_spinner;  
ArrayList<String> years;  
//Button to change filter of exercise in graph  
Button btnSubmit;  
//Objects to create the BarChart(graph)  
BarChart barChart;  
BarData barData;  
BarDataSet barDataSet;  
ArrayList<BarEntry> barEntries;  
ArrayList<UserExercise> userExercisesDone;  
ArrayList<Integer> colors = new ArrayList<Integer>();  
//Values to save  
String activeUserName;  
//Filter ID of exercise for graph  
int exerciseIDToCheck = 1;  
String yearToCheck = "2021";  
//Object for date format  
SimpleDateFormat dateFormat = new SimpleDateFormat("YYYY/MM/dd",  
Locale.ENGLISH);  
@Override  
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_statistics);  
  
    DataBaseHelper = new DataBaseHelper(this);  
  
    //finding xml parts  
    //finding the BarChart (the graph)  
    //finding spinner  
    imgUserPicture = findViewById(R.id.imgUserPicture);  
    barChart = findViewById(R.id.barChart);  
    exercise_spinner = findViewById(R.id.exercise_spinner);  
    year_spinner = findViewById(R.id.year_spinner);  
    btnSubmit = findViewById(R.id.btnSubmit);  
  
    //setting the exercise_spinner list  
    exerciseNames = new ArrayList<String>();  
    //getting the exercises
```

```
exercises = DataBaseHelper.getExercises();

//populating the exercise spinner
populateExerciseSpinner();

//setting the year_spinner list
int yearNow = 2021;
if (android.os.Build.VERSION.SDK_INT >= android.os.Build.VERSION_CODES.O)
{
    DateTimeFormatter dtf = DateTimeFormatter.ofPattern("yyyy/MM/dd
HH:mm:ss");
    LocalDateTime now = LocalDateTime.now();
    yearNow = now.getYear();
}

//getting the years
years = yearRange("2020",String.valueOf(yearNow));

//populating the year spinner
populateYearSpinner();

//getting userName from intent
Intent intent = getIntent();

activeUserName = intent.getStringExtra("activeUserName");

//changing the userPicture to show the user's picture
Bitmap userPic = PictureFileHelper.getPic(this, activeUserName);
if(userPic != null && imgUserPicture != null){
    imgUserPicture.setImageBitmap(userPic);
}

//getting userExercises by the active user name
userExercisesDone =
DataBaseHelper.getUserExercisesDoneByUserName(activeUserName);

updateDataInBarChart();

//setting onClickListener to btnSubmit
btnSubmit.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {

        //gets the exercise name that was chosen
        String exerciseNameChosen =
exercise_spinner.getSelectedItem().toString();

        int size = exercises.size();

        //searching for exercise which has that name
        //and setting the chosen exercise's ID to be
        //the one who is filtering the barChart
        for (int i=0; i<size; i++){
            Exercise exercise = exercises.get(i);

if(exercise.getExerciseName().contentEquals(exerciseNameChosen)){
            exerciseIDToCheck = exercise.getExerciseID();
}

```

```

        }

        //gets the chosen year
        String chosenYear = year_spinner.getSelectedItem().toString();

        //setting the chosen year to be
        //the one who is filtering the barChart
        yearToCheck = chosenYear;

        updateDataInBarChart();
    }

});}

private void updateDataInBarChart(){

    //calling a method to clear the barChart in order to update it properly
    barChart.clear();

    //retrieve data for our barChart
    getData();

    //now we will start our barChart
    String labelDataSet = "Number above column is the repetition." + "\n";

    labelDataSet += " Year: " + yearToCheck;

    barDataSet = new BarDataSet(barEntries, labelDataSet);

    barData = new BarData(barDataSet);

    //now we have to set our data in our bar chart
    barChart.setData(barData);

    barChart.getDescription().setText("Exercises data");

    //customizing barChart
    //setting the colors to be the colors representing
    //the user's ratings
    barDataSet.setColors(colors);
    barDataSet.setValueTextColor(Color.BLACK);
    barDataSet.setValueTextSize(18f);

    //formatting XAxis to show date
    XAxis xAxis = barChart.getXAxis();

    //JAN FEB MAR APR MAY JUN JUL AUG SEP OCT NOV DEC
    String[] moths = new String[] {"JAN", "FEB", "MAR", "APR", "MAY", "JUN",
    "JUL", "AUG", "SEP", "OCT", "NOV", "DEC"};

    xAxis.setValueFormatter(new IndexAxisValueFormatter(moths));

    xAxis.setCenterAxisLabels(true);
    xAxis.setPosition(XAxis.XAxisPosition.BOTTOM);
    xAxis.setGranularity(1);
    xAxis.setGranularityEnabled(true);
}

```

```
//setting YAxis
YAxis yAxisLeft = barChart.getAxisLeft();
YAxis yAxisRight = barChart.getAxisRight();

yAxisLeft.setAxisMaximum(150);
yAxisLeft.setAxisMinimum(0);

yAxisRight.setAxisMaximum(150);
yAxisRight.setAxisMinimum(0);

barChart.setDragEnabled(true);
barChart.setVisibleXRangeMaximum(3);

float barSpace = 0.08f;
float groupSpace = 0.44f;

barChart.getXAxis().setAxisMinimum(0);
barChart.getXAxis().setAxisMaximum(12);
//barChart.getAxisLeft().setAxisMinimum(0);

//barChart.groupBars(0, groupSpace, barSpace);

barChart.invalidate();

}

//adds content to exercise spinner
private void populateExerciseSpinner() {

    int size = exercises.size();

    for (int i=0; i<size; i++){
        Exercise exercise = exercises.get(i);
        exerciseNames.add(exercise.getExerciseName());
    }

    ArrayAdapter<String> exerciseAdapter = new ArrayAdapter<>(this,
    android.R.layout.simple_spinner_item, exerciseNames);

    exerciseAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

    exercise_spinner.setAdapter(exerciseAdapter);

}

//adds content to year spinner
private void populateYearSpinner(){

    ArrayAdapter<String> exerciseAdapter = new ArrayAdapter<>(this,
    android.R.layout.simple_spinner_item, years);

    exerciseAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item)
```

```

tem);

        year_spinner.setAdapter(exerciseAdapter);
    }

//a method to get data to barChart (to barEntries)
private void getData(){

    barEntries = new ArrayList<BarEntry>();

    if(userExercisesDone != null)
    {
        int length = userExercisesDone.size();

        for (int i=0; i<length; i++) {

            UserExercise userExercise = userExercisesDone.get(i);

            //checks the filter to what exercise should the graph show
            if(userExercise.getExerciseID() == exerciseIDToCheck){

                int rating = userExercise.getRating();
                int repetition = userExercise.getRepetition();
                String date = userExercise.getDate();
                //EXAMPLE DATE: "FEB 28 2021" 11 (year - 7-10, day - 5-6,
month 0-2)

                String[] splited = date.split("\\"s+");

                int year = Integer.parseInt(splited[2]);
                int day = Integer.parseInt(splited[1]);
                int month = getMonthNumber(date);

                //String dateForGraph = day + " " + month + " " + year;

                //checks the filter to what year should the graph show
                if(year == Integer.parseInt(yearToCheck)) {
                    long dateInMillis = getTimeInMillis(day, month, year);

                    Date dateOfExercise = new Date(year, month, day);

                    BarEntry entry = new BarEntry(month, repetition);

                    //adding a color that represents the difficulty of
                    //doing the exercise
                    //the color is added to a list of colors later adapted
                    //to the barChart

                    colorRating(rating);

                    barEntries.add(entry);
                }
            }
        }
    }
}

```

```

}

private void colorRating(int rating){

    switch(rating) {
        case 1:
            //Hard
            colors.add(getResources().getColor(R.color.Hard));
            break;
        case 2:
            //Hard-Medium
            colors.add(getResources().getColor(R.color.HardMedium));
            break;
        case 3:
            //
            colors.add(getResources().getColor(R.color.Medium));
            break;
        case 4:
            //Easy-Medium
            colors.add(getResources().getColor(R.color.EasyMedium));
            break;
        case 5:
            //Easy
            colors.add(getResources().getColor(R.color.Easy));
            break;
        default:
            colors.add(getResources().getColor(R.color.Easy));
            break;
    }
}

//This Method will convert given specific day,month and year into
milliseconds.
// It will be very help when using Timpicker or Datepicker.
public static long getTimeInMillis(int day, int month, int year) {
    Calendar calendar = Calendar.getInstance();
    calendar.set(year, month, day);
    return calendar.getTimeInMillis();
}

//makes the real month a number month (JAN -> 1)
private int getMonthNumber(String date) {
    if(date.contains("JAN"))
        return 1;
    if(date.contains("FEB"))
        return 2;
    if(date.contains("MAR"))
        return 3;
    if(date.contains("APR"))
        return 4;
    if(date.contains("MAY"))
        return 5;
    if(date.contains("JUN"))
        return 6;
    if(date.contains("JUL"))
        return 7;
}

```

```
        return 7;
    if(date.contains("AUG"))
        return 8;
    if(date.contains("SEP"))
        return 9;
    if(date.contains("OCT"))
        return 10;
    if(date.contains("NOV"))
        return 11;
    if(date.contains("DEC"))
        return 12;

    //default should never happen
    return 1;
}

public static ArrayList<String> yearRange(String startYear, String endYear) {
    int cur = Integer.parseInt(startYear);
    int stop = Integer.parseInt(endYear);
    ArrayList<String> list = new ArrayList<String>();
    while (cur <= stop) {
        list.add(String.valueOf(cur++));
    }
    return list;
}

//a method to go back to ProgramUserActivity when clicking back arrow in
@Override
public void onBackPressed() {

    Intent intent = new Intent(this, ProgramUserActivity.class);
    intent.putExtra("activeUserName", activeUserName);
    startActivity(intent);
    finish();
}
}
```

User

```

package com.example.fitnote13022021;

public class User {

    // User Table:
    // (userName TEXT PRIMARY KEY, userPassword TEXT, userLevel INTEGER,
    // userWeight INTEGER, userHeight INTEGER, userBirthDate TEXT, userGender
    // TEXT, profilePic INTEGER)

    private String userName;
    private String userPassword;
    private int userWeight;
    private int userHeight;
    private String userBirthDate;
    private String userGender;
    private String profilePic;

    //constructors
    //constructor for all info
    public User(String userName, String userPassword, int userWeight, int
    userHeight, String userBirthDate, String userGender, String profilePic) {
        this.userName = userName;
        this.userPassword = userPassword;
        this.userWeight = userWeight;
        this.userHeight = userHeight;
        this.userBirthDate = userBirthDate;
        this.userGender = userGender;
        this.profilePic = profilePic;
    }

    //constructor for user
    public User(User user) {

        this.userName = user.getUserName();
        this.userPassword = user.getUserPassword();
        this.userWeight = user.getUserWeight();
        this.userHeight = user.getUserHeight();
        this.userBirthDate = user.getUserBirthDate();
        this.userGender = user.getUserGender();
        this.profilePic = user.getProfilePic();

    }

    //constructors
    //constructor for no info
    public User() {
    }

    // toString is necessary for printing the contents of a class object
    @Override
    public String toString() {
        return "User{" +
            "userName='" + userName + '\'' +
            ", userPassword='" + userPassword + '\'' +
            ", userWeight=" + userWeight +
            ", userHeight=" + userHeight +

```

```
        ", userBirthDate='\" + userBirthDate + '\" +
        ", userGender='\" + userGender + '\" +
        ", profilePic='\" + profilePic +
        '\"';

    }

//Getters and Setters
public String getUserName() {
    return userName;
}

public void setUserName(String userName) {
    this.userName = userName;
}

public String getUserPassword() {
    return userPassword;
}

public void setUserPassword(String userPassword) {
    this.userPassword = userPassword;
}

public int getUserWeight() {
    return userWeight;
}

public void setUserWeight(int userWeight) {
    this.userWeight = userWeight;
}

public int getUserHeight() {
    return userHeight;
}

public void setUserHeight(int userHeight) {
    this.userHeight = userHeight;
}

public String getUserBirthDate() {
    return userBirthDate;
}

public void setUserBirthDate(String userBirthDate) {
    this.userBirthDate = userBirthDate;
}

public String getUserGender() {
    return userGender;
}

public void setUserGender(String userGender) {
    this.userGender = userGender;
}

public String getProfilePic() {
    return profilePic;
}
```

```
public void setProfilePic(String profilePic) {  
    this.profilePic = profilePic;  
}  
  
public int getUserLevel(){  
    return 0;  
}  
}
```

## UserExercise

```

package com.example.fitnote13022021;

public class UserExercise{

    //table userExercises
    //(userExerciseID INTEGER PRIMARY KEY AUTOINCREMENT,
    // userUserName TEXT, exerciseID INTEGER, date TEXT, time INTEGER,
    // rating INTEGER, repetition INTEGER)

    private int userExerciseID;
    private String userName;
    private int exerciseID;
    private String date;
    private int time;
    private int rating;
    private int repetition;

    //constructors
    //constructor for new UserExercises
    public UserExercise(String userName, int exerciseID) {
        this.userName = userName;
        this.exerciseID = exerciseID;

        this.date = null;
        this.time = 0;
        this.rating = 0;
        this.repetition = 0;
    }

    //constructor to take existing newUserExercises
    public UserExercise(int userExerciseID, String userName, int exerciseID,
String date, int time, int rating, int repetition) {
        this.userExerciseID = userExerciseID;
        this.userName = userName;
        this.exerciseID = exerciseID;
        this.date = date;
        this.time = time;
        this.rating = rating;
        this.repetition = repetition;
    }

    // toString is necessary for printing the contents of a class object
    @Override
    public String toString() {
        return "UserExercise{" +
            "userExerciseID=" + userExerciseID +
            ", userName='" + userName + '\'' +
            ", exerciseID=" + exerciseID +
            ", date='" + date + '\'' +
            ", time=" + time +
            ", rating=" + rating +
            ", repetition=" + repetition +
            '}';
    }

    //Getters and Setters
}

```

```
public int getUserExerciseID() {
    return userExerciseID;
}

public void setUserExerciseID(int userExerciseID) {
    this.userExerciseID = userExerciseID;
}

public String getUserName() {
    return userName;
}

public void setUserName(String userName) {
    this.userName = userName;
}

public int getExerciseID() {
    return exerciseID;
}

public void setExerciseID(int exerciseID) {
    this.exerciseID = exerciseID;
}

public String getDate() {
    return date;
}

public void setDate(String date) {
    this.date = date;
}

public int getTime() {
    return time;
}

public void setTime(int time) {
    this.time = time;
}

public int getRating() {
    return rating;
}

public void setRating(int rating) {
    this.rating = rating;
}

public int getRepetition() {
    return repetition;
}

public void setRepetition(int repetition) {
    this.repetition = repetition;
}

}
```

UserExerciseDoneAdapter

```

package com.example.fitnote13022021;

import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.ImageView;
import android.widget.TextView;

import java.util.ArrayList;

public class UserExerciseDoneAdapter extends BaseAdapter {

    private ArrayList<UserExercise> arrayList;
    public static Context context;
    private DataBaseHelper DataBaseHelper;

    public UserExerciseDoneAdapter(ArrayList<UserExercise> arrayList, Context context){
        this.arrayList = arrayList;
        this.context = context;
        DataBaseHelper = new DataBaseHelper(context);
    }

    @Override
    public int getCount() {
        return arrayList.size();
    }

    @Override
    public Object getItem(int position) { return arrayList.get(position); }

    @Override
    public long getItemId(int position) {
        return 0;
    }

    @Override
    public View getView(int position, View convertView, ViewGroup parent) {

        UserExercise userExercise = arrayList.get(position);
        convertView =
LayoutInflater.from(context).inflate(R.layout.user_exercise_done_layout, null);

        ImageView imageViewOfUserExerciseDone =
convertView.findViewById(R.id.imageViewOfUserExerciseDone);
        TextView txtExerciseOfUserExerciseDone =
convertView.findViewById(R.id.txtExerciseOfUserExerciseDone);
        TextView txtDateOfUserExerciseDone =
convertView.findViewById(R.id.txtDateOfUserExerciseDone);
        TextView txtTimeOfUserExerciseDone =
convertView.findViewById(R.id.txtTimeOfUserExerciseDone);
        TextView txtRatingOfUserExerciseDone =
convertView.findViewById(R.id.txtRatingOfUserExerciseDone);
        TextView txtRepetitionOfUserExerciseDone =

```

```
convertView.findViewById(R.id.txtRepetitionOfUserExerciseDone);  
  
        //setting textviews to show userExercise info  
        String textViewText = txtDateOfUserExerciseDone.getText().toString();  
  
        txtDateOfUserExerciseDone.setText(textViewText + userExercise.getDate());  
  
        textViewText = txtTimeOfUserExerciseDone.getText().toString();  
  
        txtTimeOfUserExerciseDone.setText(textViewText + userExercise.getTime());  
  
        textViewText = txtRatingOfUserExerciseDone.getText().toString();  
  
        txtRatingOfUserExerciseDone.setText(textViewText +  
userExercise.getRating());  
  
        textViewText = txtRepetitionOfUserExerciseDone.getText().toString();  
  
        txtRepetitionOfUserExerciseDone.setText(textViewText +  
userExercise.getRepetition());  
  
        //getting the exercise by exerciseID  
        Exercise exerciseFromUserExercise =  
dataBaseHelper.getExerciseByID(userExercise.getExerciseID());  
  
        //setting the image to show exercise image  
  
imageViewOfUserExerciseDone.setImageResource(exerciseFromUserExercise.getExerciseP  
ic());  
  
        //setting the lasy textView to show the exercise's name  
        textViewText = txtExerciseOfUserExerciseDone.getText().toString();  
  
        txtExerciseOfUserExerciseDone.setText(textViewText +  
exerciseFromUserExercise.getExerciseName());  
  
    }  
    return convertView;  
}
```

## UserExerciseNotDoneAdapter

```
package com.example.fitnote13022021;

import android.app.Activity;
import android.app.AlertDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.ListAdapter;
import android.widget.TextView;
import android.widget.Toast;

import java.util.ArrayList;

public class UserExerciseNotDoneAdapter extends BaseAdapter implements ListAdapter {

    private ArrayList<UserExercisesInnerJoinEx> arrayList;
    public static Context context;
    private DataBaseHelper DataBaseHelper;
    private String activeUserName;

    public UserExerciseNotDoneAdapter(ArrayList<UserExercisesInnerJoinEx> arrayList, Context context, String userName) {
        this.arrayList = arrayList;
        this.context = context;
        DataBaseHelper = new DataBaseHelper(context);
        this.activeUserName = userName;
    }
    public UserExerciseNotDoneAdapter(ArrayList<UserExercisesInnerJoinEx> arrayList, Context context) {
        this.arrayList = arrayList;
        this.context = context;
        DataBaseHelper = new DataBaseHelper(context);
    }

    @Override
    public int getCount() {
        return arrayList.size();
    }

    @Override
    public Object getItem(int position) {
        return arrayList.get(position);
    }

    @Override
    public long getItemId(int position) {
        return 0;
    }
}
```

```

@Override
public View getView(int position, View convertView, ViewGroup parent) {

    UserExercisesInnerJoinEx userExercisesInnerJoinEx =
arrayList.get(position);
    convertView =
LayoutInflater.from(context).inflate(R.layout.user_exercise_layout, null);

    TextView txtMainTitle = convertView.findViewById(R.id.txtMainTitle);
    ImageView imageView = convertView.findViewById(R.id.imageView);
    Button btnDelete = convertView.findViewById(R.id.btnDelete);

    txtMainTitle.setText(userExercisesInnerJoinEx.getExerciseName());
    imageView.setImageResource(userExercisesInnerJoinEx.getExercisePic());

    btnDelete.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Toast.makeText(context,
txtMainTitle.getText().toString(),Toast.LENGTH_SHORT).show();

            //creating the dialog to delete the userExercise
            AlertDialog.Builder builder = new AlertDialog.Builder(context);

            builder.setTitle("delete exercise");

            builder.setMessage("Do you really want to delete?");

            //delete
            builder.setPositiveButton("Yes", new
DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {

dataBaseHelper.deleteUserExercise(activeUserName,userExercisesInnerJoinEx.getExerc
iseName(), userExercisesInnerJoinEx.getUserExerciseID());
                    arrayList.remove(position);
                    notifyDataSetChanged();
                }
            });

            //cancel
            builder.setNegativeButton("No", new
DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    dialog.cancel();
                }
            });

            //Creating and showing the dialog
            AlertDialog dialog = builder.create();

            dialog.show();

        }
    });
}

```

```
imageView.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Toast.makeText(context,
userExercisesInnerJoinEx.getExerciseName(), Toast.LENGTH_SHORT).show();

        Intent intent1 = new Intent(context,
ExecuteExerciseActivity.class);

        intent1.putExtra("ExecuteUserExerciseID",
userExercisesInnerJoinEx.getUserExerciseID());
        intent1.putExtra("ExecuteExerciseID",
userExercisesInnerJoinEx.getExerciseID());
        intent1.putExtra("activeUserName",activeUserName);
        context.startActivity(intent1);
        ((Activity)context).finish();
    }
});

txtMainTitle.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Toast.makeText(context,
userExercisesInnerJoinEx.getExerciseName(), Toast.LENGTH_SHORT).show();

        Intent intent1 = new Intent(context,
ExecuteExerciseActivity.class);

        intent1.putExtra("ExecuteUserExerciseID",
userExercisesInnerJoinEx.getUserExerciseID());
        intent1.putExtra("ExecuteExerciseID",
userExercisesInnerJoinEx.getExerciseID());
        intent1.putExtra("activeUserName",activeUserName);
        context.startActivity(intent1);
        ((Activity)context).finish();
    }
});

return convertView;
}
})
```

UserExercisesInnerJoinEx

```

package com.example.fitnote13022021;

public class UserExercisesInnerJoinEx {

    //UserExercisesInnerJoinEx (int userExerciseID, int exerciseID, String
    exerciseName, int exercisePic, String exerciseDetail)

    private int userExerciseID;
    private int exerciseID;
    private String exerciseName;
    private int exercisePic;
    private String exerciseDetail;

    //constructors
    public UserExercisesInnerJoinEx(int userExerciseID, int exerciseID, String
    exerciseName, int exercisePic, String exerciseDetail) {
        this.userExerciseID = userExerciseID;
        this.exerciseID = exerciseID;
        this.exerciseName = exerciseName;
        this.exercisePic = exercisePic;
        this.exerciseDetail = exerciseDetail;
    }

    // toString is necessary for printing the contents of a class object
    @Override
    public String toString() {
        return "UserExercisesInerJoinEx{" +
            "userExerciseID=" + userExerciseID +
            ", exerciseID=" + exerciseID +
            ", exerciseName='" + exerciseName + '\'' +
            ", exercisePic=" + exercisePic +
            ", exerciseDetail=''" + exerciseDetail + '\'' +
            '}';
    }

    //Getters and Setters
    public int getUserExerciseID() {
        return userExerciseID;
    }

    public void setUserExerciseID(int userExerciseID) {
        this.userExerciseID = userExerciseID;
    }

    public int getExerciseID() {
        return exerciseID;
    }

    public void setExerciseID(int exerciseID) {
        this.exerciseID = exerciseID;
    }

    public String getExerciseName() {
        return exerciseName;
    }
}

```

```
    }

    public void setExerciseName(String exerciseName) {
        this.exerciseName = exerciseName;
    }

    public int getExercisePic() {
        return exercisePic;
    }

    public void setExercisePic(int exercisePic) {
        this.exercisePic = exercisePic;
    }

    public String getExerciseDetail() {
        return exerciseDetail;
    }

    public void setExerciseDetail(String exerciseDetail) {
        this.exerciseDetail = exerciseDetail;
    }
}
```

## UsersAdapter

```
package com.example.fitnote13022021;

import android.app.AlertDialog;
import android.app.DatePickerDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Color;
import android.os.Bundle;
import android.provider.MediaStore;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.SeekBar;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.Nullable;

import java.util.ArrayList;
import java.util.Calendar;

public class UsersAdapter extends BaseAdapter {

    private ArrayList<User> arrayList;
    private User user;

    //get activeUserName to not delete and change him
    String activeUserName;
    int activeUserLevel;

    public static Context context;

    public UsersAdapter(ArrayList<User> arrayList, Context context, String activeUserName, int activeUserLevel){
        this.arrayList = arrayList;
        this.context = context;
        this.activeUserName = activeUserName;
        this.activeUserLevel = activeUserLevel;
    }

    @Override
    public int getCount() {
        return arrayList.size();
    }

    @Override
```

```

public Object getItem(int position) {
    return arrayList.get(position);
}

@Override
public long getItemId(int position) {
    return 0;
}

public class MyHolder {
    ImageView imgUserPicUserLayout;
    TextView txtUserNameShowUserLayout, txtUserGenderShowUserLayout;

    public MyHolder(View v) {
        imgUserPicUserLayout = (ImageView)
v.findViewById(R.id.imgUserPicUserLayout);

        txtUserNameShowUserLayout = (TextView)
v.findViewById(R.id.txtUserNameShowUserLayout);
        txtUserGenderShowUserLayout = (TextView)
v.findViewById(R.id.txtUserGenderShowUserLayout);
    }
}

// This is a very important method:
// You can write your code in this function
// You can set your views methods:
@Override
public View getView(int position, View convertView, ViewGroup parent) {

    View view = convertView;

    MyHolder holder;

    // using polymorphism on users to get Admin info as well
    user = arrayList.get(position);

    //Check if an existing view is being reused, otherwise inflate the view
    if (view == null) {
        view = LayoutInflater.from(context).inflate(R.layout.user_layout,
null);
        holder = new MyHolder(view);
        view.setTag(holder);
    }else {
        holder = (MyHolder) view.getTag();
    }

    //Setting the userPic to be the user's pic
    //changing the userPicture to show the user's picture
    Bitmap userPic = PictureFileHelper.getPic(context, user.getUserName());
    if(userPic != null && holder.imgUserPicUserLayout != null){
        holder.imgUserPicUserLayout.setImageBitmap(userPic);
    }

    //setting textviews to show userExercise info
    holder.txtUserNameShowUserLayout.setText(user.getUserName());
    holder.txtUserGenderShowUserLayout.setText(user.getUserGender());
}

```

```
        return view;  
    }  
  
}
```

## ViewExercisesResultsActivity

```
package com.example.fitnote13022021;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.graphics.Bitmap;
import android.os.Bundle;
import android.widget.ImageView;
import android.widget.ListView;
import android.widget.SearchView;
import android.widget.TextView;

import java.text.DecimalFormat;
import java.util.ArrayList;
import java.util.Locale;

public class ViewExercisesResultsActivity extends AppCompatActivity {

    //Initialize variables
    public static String activeUserName = null;

    DataBaseHelper DataBaseHelper;
    ImageView imgUserPictureExercisesResultsScreen;
    TextView txtBMIDoneExercises;
    ListView listViewUserExercisesDone;
    ArrayList<UserExercise> userExercisesDone;
    ArrayList<UserExercise> filteredUserExercisesDone;
    SearchView searchViewUserExerciseDoneList;
    UserExerciseDoneAdapter userExerciseDoneAdapter;

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

        DataBaseHelper = new DataBaseHelper(this);

        //finding XML parts / Assign variables
        imgUserPictureExercisesResultsScreen =
            findViewById(R.id.imgUserPictureExercisesResultsScreen);
        txtBMIDoneExercises = findViewById(R.id.txtBMIDoneExercises);
        searchViewUserExerciseDoneList =
            findViewById(R.id.searchViewUserExerciseDoneList);
        listViewUserExercisesDone = findViewById(R.id.listViewUserExercisesDone);

        //setting filteredExercises to be exercises
        filteredUserExercisesDone = userExercisesDone;

        Intent intent = getIntent();
    }
}
```

```

//getting the active userName from getting the user who entered with Log
in with his name given from the intent
String userName = intent.getStringExtra("activeUserName");

activeUserName = userName;

//changing the userPicture to show the user's picture
Bitmap userPic = PictureFileHelper.getPic(this, activeUserName);
if(userPic != null && imgUserPictureExercisesResultsScreen != null){
    imgUserPictureExercisesResultsScreen.setImageBitmap(userPic);
}

//getting the exercises done by the user
userExercisesDone =
dataBaseHelper.getUserExercisesDoneByUserName(activeUserName);

userExerciseDoneAdapter = new UserExerciseDoneAdapter(userExercisesDone,
ViewExercisesResultsActivity.this);

listViewUserExercisesDone.setAdapter(userExerciseDoneAdapter);

//initialize the searchWidget in order to filter exercises
initSearchWidgets();

//changing textView to show user's BMI
//Getting the user's weight and height
User activeUser = dataBaseHelper.getUserByName(activeUserName);
double userWeight = activeUser.getUserWeight();
double userHeight = activeUser.getUserHeight() * 0.01;//turning cm to m
//BMI = kg/m2
double userBMI = userWeight / (userHeight*userHeight);

String BMIMessageToUser = activeUserName + " BMI is: " + new
DecimalFormat("##.##").format(userBMI);

txtBMIDoneExercises.setText(BMIMessageToUser);
//A BMI of 25.0 or more is overweight, while the healthy range is 18.5 to
24.9.

}

private void initSearchWidgets() {

SearchView searchView = (SearchView)searchViewUserExerciseDoneList;

searchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() {
@Override
public boolean onQueryTextSubmit(String query) {
    return false;
}

//this method is called every time a user
//puts in any character into the search view
//literally any chang :D
@Override
public boolean onQueryTextChange(String newText) {

```

```

        filteredUserExercisesDone = new ArrayList<UserExercise>();

        //same as a regular for loop
        for (UserExercise userExercise: userExercisesDone)
        {

            //getting the exerciseName by exerciseID
            String exerciseName =
            DataBaseHelper.getExerciseByID(userExercise.getExerciseID()).getExerciseName();

            //if a userExercise has one of the letter in the date

            if(userExercise.getDate().toLowerCase().contains(newText.toLowerCase(Locale.ROOT)))
            {
                filteredUserExercisesDone.add(userExercise);
            }

            //if a userExercise has one of the letter in the time
            else
            if(String.valueOf(userExercise.getTime()).contains(newText.toLowerCase(Locale.ROOT)))
            {
                filteredUserExercisesDone.add(userExercise);
            }

            //if a userExercise has one of the letter in the rating
            else
            if(String.valueOf(userExercise.getRating()).contains(newText.toLowerCase(Locale.ROOT)))
            {
                filteredUserExercisesDone.add(userExercise);
            }

            //if a userExercise has one of the letter in the repetition
            else
            if(String.valueOf(userExercise.getRepetition()).contains(newText.toLowerCase(Locale.ROOT)))
            {
                filteredUserExercisesDone.add(userExercise);
            }

            //if a userExercise has one of the letter in the name
            else
            if(exerciseName.toLowerCase().contains(newText.toLowerCase(Locale.ROOT))){
                filteredUserExercisesDone.add(userExercise);
            }
        }

        UserExerciseDoneAdapter exerciseAdaptor = new
        UserExerciseDoneAdapter(filteredUserExercisesDone,
        ViewExercisesResultsActivity.this);

        listViewUserExercisesDone.setAdapter(exerciseAdaptor);

        return false;
    );
});
```

```
//a method to go back to ProgramUserActivity when clicking back arrow in  
bottom  
@Override  
public void onBackPressed() {  
  
    Intent intent = new Intent(this, ProgramUserActivity.class);  
  
    intent.putExtra("activeUserName", activeUserName);  
  
    startActivity(intent);  
  
    finish();  
  
}  
}
```

## WhatsAppSendActivity

```
package com.example.fitnote13022021;

import androidx.appcompat.app.AppCompatActivity;

import android.app.AlarmManager;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.net.Uri;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import java.util.ArrayList;
import java.util.Calendar;
import java.util.Date;

public class WhatsAppSendActivity extends AppCompatActivity {

    //Initialize variables
    DataBaseHelper DataBaseHelper;

    TextView tvName, tvPhone;

    EditText etWhatsAppMessage;

    Button btSendWhatsAppMessage;

    String userExercisesSummaryText = "Here is my data on all the exercises I did
in the last week successfully:";

    String activeUserName;

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

        DataBaseHelper = new DataBaseHelper(this);

        //Assign variables
        tvName = findViewById(R.id.tvName);
        tvPhone = findViewById(R.id.tvPhone);
        etWhatsAppMessage = findViewById(R.id.etWhatsAppMessage);
        btSendWhatsAppMessage = findViewById(R.id.btSendWhatsAppMessage);

        Intent intent = getIntent();

        activeUserName = intent.getStringExtra("activeUserName");

        //getting information from database to send on WhatsApp
        ArrayList<UserExercise> userExercisesDoneByUser =
        DataBaseHelper.getUserExercisesDoneByUserName(activeUserName);
```

```

//Takes only the UserExercises done in the last week
ArrayList<UserExercise> userExercisesDoneByUserLastWeek = new
ArrayList<UserExercise>();

int arrayListUserExercisesSize= userExercisesDoneByUser.size();

Calendar dayOfExercise = Calendar.getInstance();

//go over userExercisesDoneByUser to get exercises done in last week
for (int i=0; i<arrayListUserExercisesSize; i++) {

    UserExercise userExercise = userExercisesDoneByUser.get(i);

    String date = userExercise.getDate();

    //EXAMPLE DATE: "FEB 28 2021" 11 (year - 7-10, day - 5-6, month 0-2)

    String[] splited = date.split("\\s+");

    int year = Integer.parseInt(splited[2]);
    int day = Integer.parseInt(splited[1]);
    int month = getMonthNumber(date);

    dayOfExercise.set(Calendar.YEAR, year);
    //month starts at 0 in Calendar
    dayOfExercise.set(Calendar.MONTH, month-1);
    dayOfExercise.set(Calendar.DAY_OF_MONTH, day);
    dayOfExercise.set(Calendar.HOUR_OF_DAY, 0);
    dayOfExercise.set(Calendar.MINUTE, 0);
    dayOfExercise.set(Calendar.SECOND, 0);
    dayOfExercise.set(Calendar.MILLISECOND, 0);

    Calendar currentDate = Calendar.getInstance();

    int dayDifference = 0;
    //get the day difference between today and the date of the exercise
    //if the two dates are not null
    if (dayOfExercise != null && currentDate != null){
        dayDifference = (int)( (currentDate.getTime().getTime() -
dayOfExercise.getTime().getTime()) / (1000 * 60 * 60 * 24));
    }

    if (dayDifference >=0 && dayDifference <= 7){
        userExercisesDoneByUserLastWeek.add(userExercise);
    }
}

//We need the exercises in order to send the name of the exercise with the
data
ArrayList<Exercise> exercises = DataBaseHelper.getExercises();

int arrayListExercisesSize = exercises.size();

//go over userExercisesDoneByUserLastWeek to take data
int sizeExercisesLastWeek = userExercisesDoneByUserLastWeek.size();
for (int i=0; i<sizeExercisesLastWeek; i++) {

```

```

//((userExerciseID INTEGER PRIMARY KEY AUTOINCREMENT, userName TEXT,
// exerciseID INTEGER, date TEXT, time(SEC) INTEGER, rating INTEGER,
repetition INTEGER)
UserExercise userExercise = userExercisesDoneByUserLastWeek.get(i);

int exerciseID = userExercise.getExerciseID();

String exerciseName = "";

//getting the exercise name
for (int j=0; j<arrayListExercisesSize; j++){
    Exercise exercise = exercises.get(j);
    if(exerciseID == exercise.getExerciseID()){
        exerciseName = exercise.getExerciseName();
    }
}

String date = userExercise.getDate();
int time = userExercise.getTime();
int rating = userExercise.getRating();
int repetition = userExercise.getRepetition();

userExercisesSummaryText += "\n" + "----->" + "\n";
userExercisesSummaryText += " - The exercise: "
+ exerciseName + "\n";

userExercisesSummaryText += " - The date in which the exercise was
done: "
+ date + "\n";

userExercisesSummaryText += " - The time it took to perform the
exercise: "
+ getDurationString(time) + "\n";

userExercisesSummaryText += " - The rating " + activeUserName + " gave
to the exercise: "
+ getRating(rating) + "\n";

userExercisesSummaryText += " - " + activeUserName + " did the
exercise "
+ repetition + "times" + "\n";

}

//get contactName
String contactName = intent.getStringExtra("contactName");
//get contactNumber
String contactNumber = intent.getStringExtra("contactNumber");

String tvNameTxt = tvName.getText().toString();
String tvPhoneTxt = tvPhone.getText().toString();

tvName.setText(tvNameTxt + contactName);
tvPhone.setText(tvPhoneTxt + contactNumber);

```

```

btSendWhatsAppMessage.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        //get text to send from etWhatsAppMessage
        String textMessage = etWhatsAppMessage.getText().toString() +
"\n";
        textMessage += userExercisesSummaryText;
        sendMessage(contactNumber, textMessage);
    }
});

private void sendMessage(String phoneNumber, String text){
    //boolean to check if WhatsApp exists
    boolean installed = isAppInstalled("com.whatsapp");

    if (installed) {
        Intent intent = new Intent(Intent.ACTION_VIEW);
        //WhatsApp message requires phone number & text(the message itself)

        intent.setData(Uri.parse("http://api.whatsapp.com/send?phone="+phoneNumber+"&text="+
" "+text));
        startActivity(intent);
    }else{
        Toast.makeText(this, "WhatsApp is not installed!",
        Toast.LENGTH_SHORT).show();
    }
}

// A method to check if WhatsApp exists on the users phone
private boolean isAppInstalled(String s) {

    PackageManager packageManager = getPackageManager();
    boolean is_installed;

    try {
        packageManager.getPackageInfo(s, PackageManager.GET_ACTIVITIES);
        is_installed = true;
    } catch (PackageManager.NameNotFoundException e) {
        e.printStackTrace();
        is_installed = false;
    }

    return is_installed;
}

private String getDurationString(int seconds) {

    int hours = seconds / 3600;
    int minutes = (seconds % 3600) / 60;
    seconds = seconds % 60;

    return twoDigitString(hours) + " : " + twoDigitString(minutes) + " : " +
twoDigitString(seconds);
}

```

```

private String twoDigitString(int number) {
    if (number == 0) {
        return "00";
    }
    if (number / 10 == 0) {
        return "0" + number;
    }
    return String.valueOf(number);
}

private String getRating(int rating){
    String ratingString = "Easy";
    switch (rating){
        case 1:
            //Hard
            ratingString = "Hard";
            break;
        case 2:
            //Hard-Medium
            ratingString = "Hard-Medium";
            break;
        case 3:
            //Medium
            ratingString = "Medium";
            break;
        case 4:
            //Easy-Medium
            ratingString = "Easy-Medium";
            break;
        case 5:
            //Easy
            ratingString = "Easy";
            break;
    }
    return ratingString;
}

//makes the real month a number month (JAN -> 1)
private int getMonthNumber(String date) {
    if(date.contains("JAN"))
        return 1;
    if(date.contains("FEB"))
        return 2;
    if(date.contains("MAR"))
        return 3;
}

```

```
if(date.contains("APR"))
    return 4;
if(date.contains("MAY"))
    return 5;
if(date.contains("JUN"))
    return 6;
if(date.contains("JUL"))
    return 7;
if(date.contains("AUG"))
    return 8;
if(date.contains("SEP"))
    return 9;
if(date.contains("OCT"))
    return 10;
if(date.contains("NOV"))
    return 11;
if(date.contains("DEC"))
    return 12;

//default should never happen
return 1;
}

//a method to go back to ProgramUserActivity when clicking back arrow in
bottom
@Override
public void onBackPressed() {

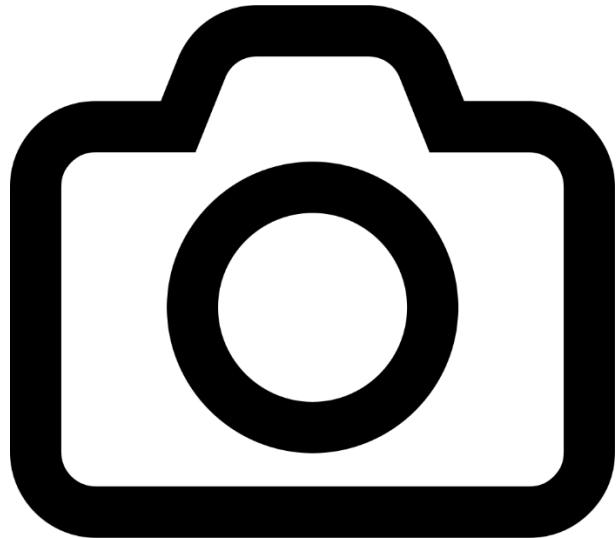
    Intent intent = new Intent(this, ShareActivity.class);
    intent.putExtra("activeUserName", activeUserName);
    startActivity(intent);
    finish();
}

}
```

## res Folder

### drawable

camra.png



default\_user.png



easy\_text.png

EASY



hammer\_curls.png

HARD

## ic\_baseline\_account\_circle.xml

```

<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="24dp"
    android:height="24dp"
    android:viewportWidth="24"
    android:viewportHeight="24"
    android:tint="?attr/colorControlNormal">
    <path
        android:fillColor="@android:color/white"
        android:pathData="M12,2C6.48,2 2,6.48 2,12s4.48,10 10,10 10,-4.48 10,-
10S17.52,2 12,2zM12,5c1.66,0 3,1.34 3,3s-1.34,3 -3,3 -3,-1.34 -3,-3 1.34,-3 3,-
3zM12,19.2c-2.5,0 -4.71,-1.28 -6,-3.22 0.03,-1.99 4,-3.08 6,-3.08 1.99,0 5.97,1.09
6,3.08 -1.29,1.94 -3.5,3.22 -6,3.22z"/>
</vector>

```

## ic\_baseline\_architecture.xml

```

<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="24dp"
    android:height="24dp"
    android:viewportWidth="24"
    android:viewportHeight="24"
    android:tint="?attr/colorControlNormal">
    <path
        android:fillColor="@android:color/white"
        android:pathData="M6.36,18.78L6.61,21l1.62,-1.54l2.77,-7.6c-0.68,-0.17 -
1.28,-0.51 -1.77,-0.98L6.36,18.78z"/>
    <path
        android:fillColor="@android:color/white"
        android:pathData="M14.77,10.88c-0.49,0.47 -1.1,0.81 -
1.77,0.98l2.77,7.6L17.39,21l0.26,-2.22L14.77,10.88z"/>
    <path
        android:fillColor="@android:color/white"
        android:pathData="M15,8c0,-1.3 -0.84,-2.4 -2,-2.82V3h-2v2.18C9.84,5.6 9,6.7
9,8c0,1.66 1.34,3 3,S15,9.66 15,8zM12,9c-0.55,0 -1,-0.45 -1,-1c0,-0.55 0.45,-1
1,-1s1,0.45 1,1C13,8.55 12.55,9 12,9z"/>
</vector>

```

## ic\_baseline\_date\_range.xml

```

<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="24dp"
    android:height="24dp"
    android:viewportWidth="24"
    android:viewportHeight="24"
    android:tint="?attr/colorControlNormal">
    <path
        android:fillColor="@android:color/white"
        android:pathData="M9,11L7,11v2h2v-2zM13,11h-2v2h2v-2zM17,11h-2v2h2v-
2zM19,4h-1L18,2h-2v2L8,4L8,2L6,2v2L5,4c-1.11,0 -1.99,0.9 -1.99,2L3,20c0,1.1 0.89,2
2,2h14c1.1,0 2,-0.9 2,-2L21,6c0,-1.1 -0.9,-2 -2,-2zM19,20L5,20L5,9h14v11z"/>
</vector>

```

ic\_baseline\_preson.xml

```

<vector android:height="24dp" android:tint="#FFB72B"
        android:viewportHeight="24" android:viewportWidth="24"
        android:width="24dp"
    xmlns:android="http://schemas.android.com/apk/res/android">
    <path android:fillColor="@android:color/white" android:pathData="M12,12c2.21,0
4,-1.79 4,-4s-1.79,-4 -4,-4 -4,1.79 -4,4 1.79,4 4,zM12,14c-2.67,0 -8,1.34 -
8,4v2h16v-2c0,-2.66 -5.33,-4 -8,-4z"/>
</vector>

```

ic\_baseline\_phonelink\_lock.xml

```

<vector xmlns:android="http://schemas.android.com/apk/res/android"
        android:width="24dp"
        android:height="24dp"
        android:viewportWidth="24"
        android:viewportHeight="24"
        android:tint="?attr/colorControlNormal">
    <path
        android:fillColor="@android:color/white"
        android:pathData="M19,1L9,1c-1.1,0 -2,0.9 -2,2v3h2L9,4h10v16L9,20v-
2L7,18v3c0,1.1 0.9,2 2,2h10c1.1,0 2,-0.9 2,-2L21,3c0,-1.1 -0.9,-2 -2,-
2zM10.8,11L10.8,9.5C10.8,8.1 9.4,7 8,7S5.2,8.1 5.2,9.5L5.2,11c-0.6,0 -1.2,0.6 -
1.2,1.2v3.5c0,0.7 0.6,1.3 1.2,1.3h5.5c0.7,0 1.3,-0.6 1.3,-1.2v-3.5c0,-0.7 -0.6,-
1.3 -1.2,-1.3zM9.5,11h-3L6.5,9.5c0,-0.8 0.7,-1.3 1.5,-1.3s1.5,0.5
1.5,1.3L9.5,11z"/>
</vector>

```

ic\_baseline\_show\_chart.xml

```

<vector xmlns:android="http://schemas.android.com/apk/res/android"
        android:width="24dp"
        android:height="24dp"
        android:viewportWidth="24"
        android:viewportHeight="24"
        android:tint="?attr/colorControlNormal">
    <path
        android:fillColor="@android:color/white"
        android:pathData="M3.5,18.49l6,-6.01 4,4L22,6.92l-1.41,-1.41 -7.09,7.97 -4,-
4L2,16.99z"/>
</vector>

```

ic\_baseline\_wc.xml

```

<vector xmlns:android="http://schemas.android.com/apk/res/android"
        android:width="24dp"
        android:height="24dp"
        android:viewportWidth="24"
        android:viewportHeight="24"
        android:tint="?attr/colorControlNormal">
    <path
        android:fillColor="@android:color/white"
        android:pathData="M5.5,22v-7.5L4,14.5L4,9c0,-1.1 0.9,-2 2,-2h3c1.1,0 2,0.9
2,2v5.5L9.5,14.5L9.5,22h-4zM18,22v-6h3l-2.54,-7.63C18.18,7.55 17.42,7 16.56,7h-
0.12c-0.86,0 -1.63,0.55 -1.9,1.37L12,16h3v6h3zM7.5,6c1.11,0 2,-0.89 2,-2s-0.89,-2
-2,-2-0.89 -2,2 0.89,2 2,2zM16.5,6c1.11,0 2,-0.89 2,-2s-0.89,-2 -2,-2 -2,0.89 -
2,2 0.89,2 2,2z"/>
</vector>

```

ic\_baseline\_wrench.xml

```
<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="24dp"
    android:height="24dp"
    android:viewportWidth="24"
    android:viewportHeight="24"
    android:tint="?attr/colorControlNormal">
<path
    android:fillColor="@android:color/white"
    android:pathData="M22.7,191c-9.1,-9.1c0.9,-2.3 0.4,-5 -1.5,-6.9 -2,-2 -5,-2.4
-7.4,-1.3L9,6 6,9 1.6,4.7C0.4,7.1 0.9,10.1 2.9,12.1c1.9,1.9 4.6,2.4
6.9,1.5L9.1,9.1c0.4,0.4 1,0.4 1.4,0L2.3,-2.3c0.5,-0.4 0.5,-1.1 0.1,-1.4z"/>
</vector>
```

ic\_launcher\_background.xml

```
<?xml version="1.0" encoding="utf-8"?>
<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="108dp"
    android:height="108dp"
    android:viewportWidth="108"
    android:viewportHeight="108">
<path
    android:fillColor="#3DDDC84"
    android:pathData="M0,0h108v108h-108z" />
<path
    android:fillColor="#00000000"
    android:pathData="M9,0L9,108"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M19,0L19,108"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M29,0L29,108"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M39,0L39,108"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M49,0L49,108"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M59,0L59,108"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M69,0L69,108"
    android:strokeWidth="0.8"
```

```
        android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M79,0L79,108"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M89,0L89,108"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M99,0L99,108"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M0,9L108,9"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M0,19L108,19"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M0,29L108,29"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M0,39L108,39"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M0,49L108,49"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M0,59L108,59"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M0,69L108,69"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M0,79L108,79"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
```

```
        android:pathData="M0,89L108,89"
        android:strokeWidth="0.8"
        android:strokeColor="#33FFFFFF" />
<path
        android:fillColor="#00000000"
        android:pathData="M0,99L108,99"
        android:strokeWidth="0.8"
        android:strokeColor="#33FFFFFF" />
<path
        android:fillColor="#00000000"
        android:pathData="M19,29L89,29"
        android:strokeWidth="0.8"
        android:strokeColor="#33FFFFFF" />
<path
        android:fillColor="#00000000"
        android:pathData="M19,39L89,39"
        android:strokeWidth="0.8"
        android:strokeColor="#33FFFFFF" />
<path
        android:fillColor="#00000000"
        android:pathData="M19,49L89,49"
        android:strokeWidth="0.8"
        android:strokeColor="#33FFFFFF" />
<path
        android:fillColor="#00000000"
        android:pathData="M19,59L89,59"
        android:strokeWidth="0.8"
        android:strokeColor="#33FFFFFF" />
<path
        android:fillColor="#00000000"
        android:pathData="M19,69L89,69"
        android:strokeWidth="0.8"
        android:strokeColor="#33FFFFFF" />
<path
        android:fillColor="#00000000"
        android:pathData="M19,79L89,79"
        android:strokeWidth="0.8"
        android:strokeColor="#33FFFFFF" />
<path
        android:fillColor="#00000000"
        android:pathData="M29,19L29,89"
        android:strokeWidth="0.8"
        android:strokeColor="#33FFFFFF" />
<path
        android:fillColor="#00000000"
        android:pathData="M39,19L39,89"
        android:strokeWidth="0.8"
        android:strokeColor="#33FFFFFF" />
<path
        android:fillColor="#00000000"
        android:pathData="M49,19L49,89"
        android:strokeWidth="0.8"
        android:strokeColor="#33FFFFFF" />
<path
        android:fillColor="#00000000"
        android:pathData="M59,19L59,89"
        android:strokeWidth="0.8"
        android:strokeColor="#33FFFFFF" />
```

```

<path
    android:fillColor="#00000000"
    android:pathData="M69,19L69,89"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
<path
    android:fillColor="#00000000"
    android:pathData="M79,19L79,89"
    android:strokeWidth="0.8"
    android:strokeColor="#33FFFFFF" />
</vector>

```

ic\_launcher\_foreground.xml

```

<vector xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:aapt="http://schemas.android.com/aapt"
    android:width="108dp"
    android:height="108dp"
    android:viewportWidth="108"
    android:viewportHeight="108">
    <path android:pathData="M31,63.928c0,0 6.4,-11 12.1,-13.1c7.2,-2.6 26,-1.4
26,-1.4l38.1,38.1L107,108.928l-32,-1L31,63.928z">
        <aapt:attr name="android:fillColor">
            <gradient
                android:endX="85.84757"
                android:endY="92.4963"
                android:startX="42.9492"
                android:startY="49.59793"
                android:type="linear">
                <item
                    android:color="#44000000"
                    android:offset="0.0" />
                <item
                    android:color="#00000000"
                    android:offset="1.0" />
            </gradient>
        </aapt:attr>
    </path>
    <path
        android:fillColor="#FFFFFF"
        android:fillType="nonZero"
        android:pathData="M65.3,45.828l3.8,-6.6c0.2,-0.4 0.1,-0.9 -0.3,-1.1c-0.4,-
0.2 -0.9,-0.1 -1.1,0.3l-3.9,6.7c-6.3,-2.8 -13.4,-2.8 -19.7,0l-3.9,-6.7c-0.2,-0.4 -
0.7,-0.5 -1.1,-0.3C38.8,38.328 38.7,38.828 38.9,39.228l3.8,6.6C36.2,49.428
31.7,56.028 31,63.928h46C76.3,56.028 71.8,49.428 65.3,45.828zM43.4,57.328c-0.8,0 -
1.5,-0.5 -1.8,-1.2c-0.3,-0.7 -0.1,-1.5 0.4,-2.1c0.5,-0.5 1.4,-0.7 2.1,-0.4c0.7,0.3
1.2,1 1.2,1.8C45.3,56.528 44.5,57.328 43.4,57.328L43.4,57.328zM64.6,57.328c-0.8,0
-1.5,-0.5 -1.8,-1.2s-0.1,-1.5 0.4,-2.1c0.5,-0.5 1.4,-0.7 2.1,-0.4c0.7,0.3 1.2,1
1.2,1.8C66.5,56.528 65.6,57.328 64.6,57.328L64.6,57.328z"
        android:strokeWidth="1"
        android:strokeColor="#00000000" />
</vector>

```

ic\_plus.xml

```
<vector android:height="24dp" android:tint="#FF9800"
    android:viewportHeight="24" android:viewportWidth="24"
    android:width="24dp"
    xmlns:android="http://schemas.android.com/apk/res/android">
    <path android:fillColor="@android:color/white" android:pathData="M13,7h-
2v4L7,11v2h4v4h2v-4h4v-2h-4L13,7zM12,2C6.48,2 2,6.48 2,12s4.48,10 10,10 10,10 -4.48
10,-10S17.52,2 12,2zM12,20c-4.41,0 -8,-3.59 -8,-8s3.59,-8 8,-8 8,3.59 8,8 -3.59,8
-8,8z"/>
</vector>
```

ic\_results.xml

```
<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="24dp"
    android:height="24dp"
    android:viewportWidth="24"
    android:viewportHeight="24"
    android:tint="?attr/colorControlNormal"
    android:autoMirrored="true">
    <path
        android:fillColor="@android:color/white"
        android:pathData="M19,3h-4.18C14.4,1.84 13.3,1 12,1c-1.3,0 -2.4,0.84 -
2.82,2L5,3c-1.1,0 -2,0.9 -2,2v14c0,1.1 0.9,2 2,2h14c1.1,0 2,-0.9 2,-2L21,5c0,-1.1
-0.9,-2 -2,-2zM12,3c0.55,0 1,0.45 1,1s-0.45,1 -1,1 -1,-0.45 -1,-1 0.45,-1 1,-
1zM14,17L7,17v-2h7v2zM17,13L7,13v-2h10v2zM17,9L7,9L7,7h10v2z"/>
</vector>
```

ic\_settings.xml

```
<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="24dp"
    android:height="24dp"
    android:viewportWidth="24"
    android:viewportHeight="24"
    android:tint="?attr/colorControlNormal">
    <path
        android:fillColor="@android:color/white"
        android:pathData="M19.14,12.94c0.04,-0.3 0.06,-0.61 0.06,-0.94c0,-0.32 -
0.02,-0.64 -0.07,-0.94l2.03,-1.58c0.18,-0.14 0.23,-0.41 0.12,-0.61l-1.92,-3.32c-
0.12,-0.22 -0.37,-0.29 -0.59,-0.22l-2.39,0.96c-0.5,-0.38 -1.03,-0.7 -1.62,-
0.94L14.4,2.81c-0.04,-0.24 -0.24,-0.41 -0.48,-0.41h-3.84c-0.24,0 -0.43,0.17 -
0.47,0.41L9.25,5.35C8.66,5.59 8.12,5.92 7.63,6.29L5.24,5.33c-0.22,-0.08 -0.47,0 -
0.59,0.22L2.74,8.87C2.62,9.08 2.66,9.34 2.86,9.48l2.03,1.58C4.84,11.36 4.8,11.69
4.8,12s0.02,0.64 0.07,0.94l-2.03,1.58c-0.18,0.14 -0.23,0.41 -
0.12,0.61l1.92,3.32c0.12,0.22 0.37,0.29 0.59,0.22l2.39,-0.96c0.5,0.38 1.03,0.7
1.62,0.94l0.36,2.54c0.05,0.24 0.24,0.41 0.48,0.41h3.84c0.24,0 0.44,-0.17 0.47,-
0.41l0.36,-2.54c0.59,-0.24 1.13,-0.56 1.62,-0.94l2.39,0.96c0.22,0.08 0.47,0 0.59,-
0.22l1.92,-3.32c0.12,-0.22 0.07,-0.47 -0.12,-0.61L19.14,12.94zM12,15.6c-1.98,0 -
3.6,-1.62 -3.6,-3.6s1.62,-3.6 3.6,-3.6s3.6,1.62 3.6,3.6s13.98,15.6 12,15.6z"/>
</vector>
```

ic\_share.xml

```
<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="24dp"
    android:height="24dp"
    android:viewportWidth="24"
    android:viewportHeight="24"
    android:tint="?attr/colorControlNormal">
<path
    android:fillColor="@android:color/white"
    android:pathData="M18,16.08c-0.76,0 -1.44,0.3 -1.96,0.77L8.91,12.7c0.05,-0.23 0.09,-0.46 0.09,-0.7s-0.04,-0.47 -0.09,-0.71L7.05,-4.11c0.54,0.5 1.25,0.81 2.04,0.81 1.66,0 3,-1.34 3,-3s-1.34,-3 -3,-3c1.34 -3,3c0,0.24 0.04,0.47 0.09,0.7L8.04,9.81C7.5,9.31 6.79,9 6,9c-1.66,0 -3,1.34 -3,3s1.34,3 3,3c0.79,0 1.5,-0.31 2.04,-0.81L7.12,4.16c-0.05,0.21 -0.08,0.43 -0.08,0.65 0,1.61 1.31,2.92 2.92,2.92 1.61,0 2.92,-1.31 2.92,-2.92s-1.31,-2.92 -2.92,-2.92z"/>
</vector>
```

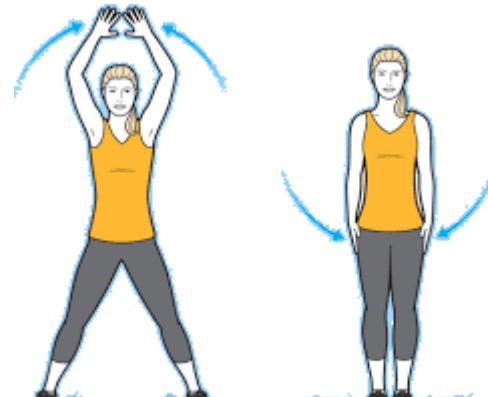
ic\_timer.xml

```
<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="24dp"
    android:height="24dp"
    android:viewportWidth="24"
    android:viewportHeight="24"
    android:tint="?attr/colorControlNormal">
<path
    android:fillColor="@android:color/white"
    android:pathData="M22,5.72l-4.6,-3.86 -1.29,1.53 4.6,3.86L22,5.72zM7.88,3.39L6.6,1.86 2.5,711.29,1.53 4.59,-3.85zM12.5,8L11,8v614.75,2.85 0.75,-1.23 -4,-2.37L12.5,8zM12,4c-4.97,0 -9,4.03 -9,9s4.02,9 9,9c4.97,0 9,-4.03 9,-9s-4.03,-9 -9,-9zM12,20c-3.87,0 -7,-3.13 -7,-7s3.13,-7 7,-7 7,3.13 7,7 -3.13,7 -7,7z"/>
</vector>
```

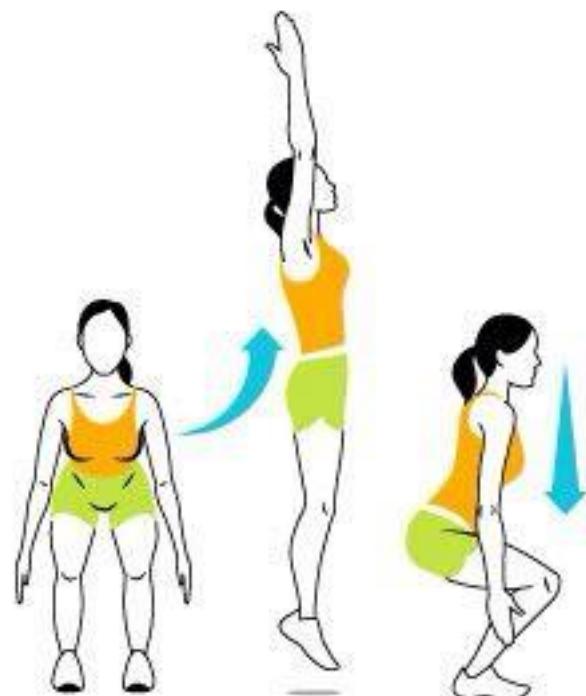
incline\_pushup.png



jumping\_jacks.png



jumping\_squat.png



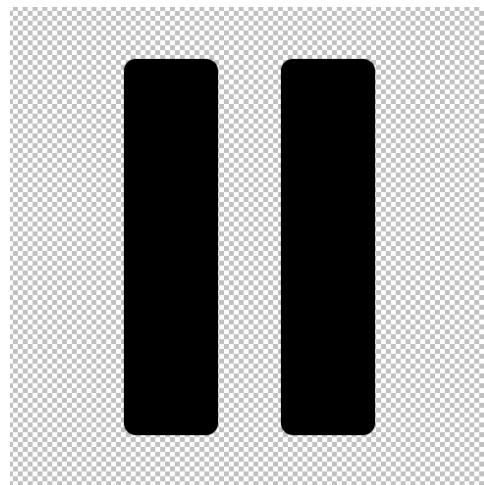
logo.png



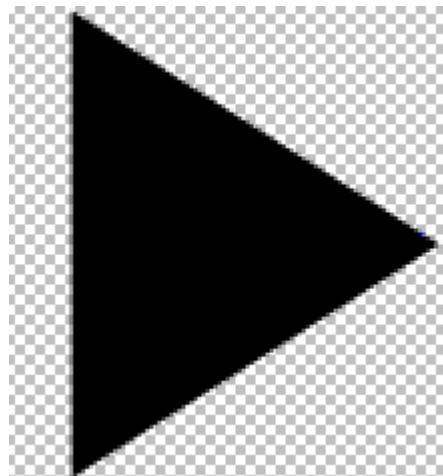
logo\_only\_pic.png



pause\_button.png



play\_button.png



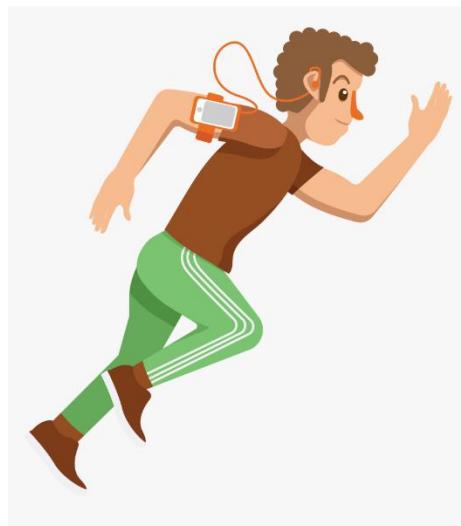
pushup.png



ropejump.png

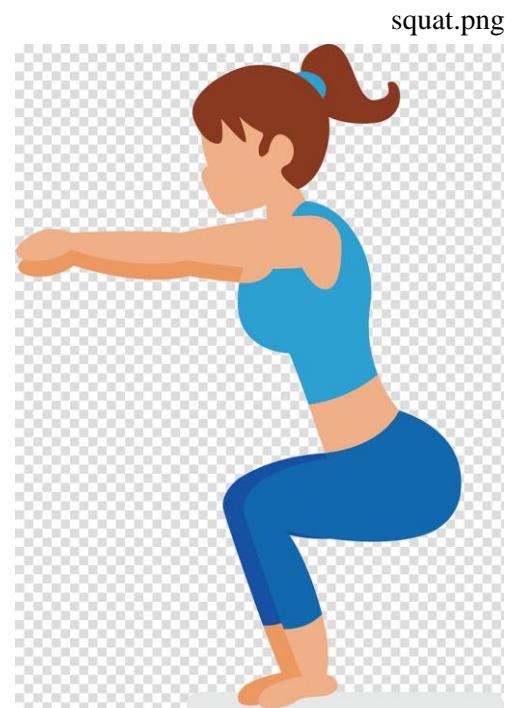


running.png

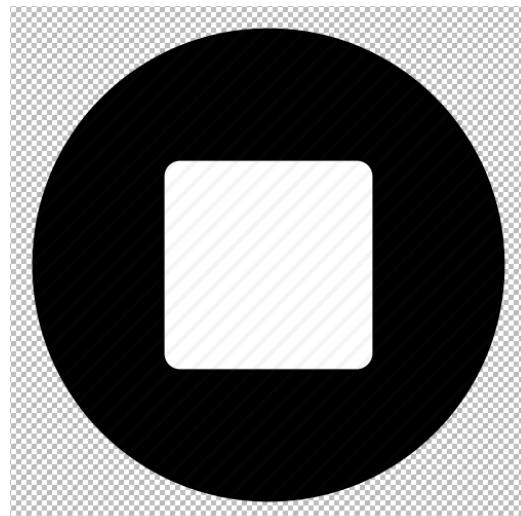


shoulder\_press.png





squat.png

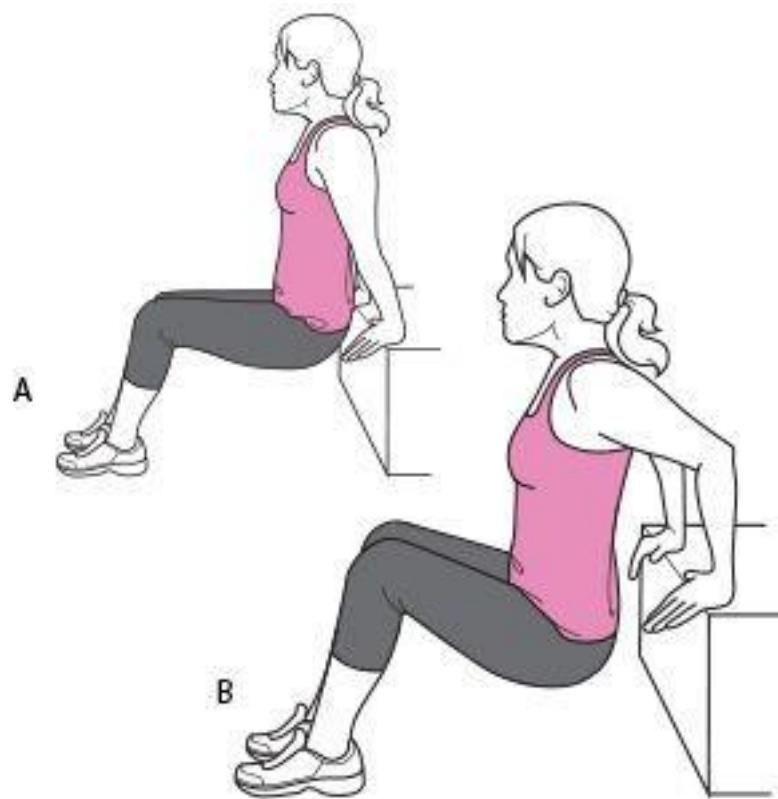


stop\_button.png

swimming.png



triceps\_dips.png



## layout

activity\_add\_exercises.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".AddExerciseActivity"
    android:background="@color/white"
    android:orientation="vertical">

    <ImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="20sp"
        android:src="@drawable/logo"></ImageView>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/txtMainTitle"
        android:text="Choose your Exercise:"
        android:textSize="30dp"
        android:textColor="@color/black"
        ></TextView>

    <SearchView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/searchViewExerciseList"
        android:iconifiedByDefault="false"
        android:queryHint="Exercise Name"
        ></SearchView>

    <ListView
        android:id="@+id/listViewExercises"
        android:layout_width="match_parent"
        android:layout_height="350dp"></ListView>

    <Button
        android:id="@+id/btCancel"
        android:layout_width="150dp"
        android:layout_height="50dp"
        android:layout_gravity="end"
        android:layout_marginTop="5dp"
        android:layout_marginEnd="20dp"
        android:text="Cancel"
        android:textAllCaps="false"
        android:backgroundTint="#FF9800"/>

</LinearLayout>
```

activity\_edit\_user.xml

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

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="Edit User:"
        android:textSize="35sp"
        android:textColor="@color/colorPrimary"
        android:id="@+id/txtTitleEditUser"
        ></TextView>

    <EditText
        android:id="@+id/etPasswordEditUser"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Password"
        android:inputType="textPassword"></EditText>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:weightSum="2">

        <TextView
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Weight (kilograms):" />

        <TextView
            android:id="@+id/txtSeekBarWeightEditUser"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="40" />

    >
</LinearLayout>

    <SeekBar
        android:id="@+id/seekBarWeightEditUser"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:max="200"
        android:min="40"
        android:layout_marginTop="15dp"/>

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

```
        android:layout_height="wrap_content"
        android:weightSum="2">

    <TextView
        android:layout_width="0dp"
        android:layout_height="wrap_content"

        android:layout_weight="1"
        android:text="Height (centimeters):" />

    <TextView
        android:id="@+id/txtSeekBarHeightEditUser"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="50" />

    >
</LinearLayout>

<SeekBar
    android:id="@+id/seekBarHeightEditUser"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:max="250"
    android:min="50"
    android:layout_marginTop="15dp"/>

<TextView
    android:id="@+id/tv_date"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Select BirthDate:"
    android:textSize="20sp"></TextView>

<Button
    android:id="@+id/btDatePickerButtonEditUser"
    style="?android:spinnerStyle"
    android:layout_width="250dp"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:onClick="openDatePicker"
    android:text="JAN 01 2020"
    android:textColor="#000000"
    android:textSize="30sp"></Button>

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Chose Gender:"
    android:textSize="14sp"></TextView>

<RadioGroup
    android:id="@+id/radioGroupGenderEditUser"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
```

```
<RadioButton
    android:id="@+id/radio_maleEditUser"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:checked="true"
    android:text="Male" />

<RadioButton
    android:id="@+id/radio_femaleEditUser"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Female" />

<RadioButton
    android:id="@+id/radio_otherEditUser"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Other" />

>
</RadioGroup>

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Chose UserLevel:"
    android:textSize="14sp"></TextView>

<RadioGroup
    android:id="@+id/radioGroupUserLevelEditUser"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="horizontal">

<RadioButton
    android:id="@+id/radio_NormalEditUser"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:checked="true"
    android:text="Normal" />

<RadioButton
    android:id="@+id/radio_AdminEditUser"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Admin" />

<RadioButton
    android:id="@+id/radio_Super_AdminEditUser"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Super Admin" />

>
</RadioGroup>
```

```
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:weightSum="2">  
  
    <Button  
        android:id="@+id/btUpdateEditUser"  
        android:layout_width="0dp"  
        android:layout_height="wrap_content"  
        android:layout_marginTop="5dp"  
        android:layout_weight="1"  
        android:text="Update"  
        android:textAllCaps="false"  
        android:textColor="@color/black"  
        android:backgroundTint="#4CAF50"/>  
  
    <Button  
        android:id="@+id/btCancelEditUser"  
        android:layout_width="0dp"  
        android:layout_height="wrap_content"  
        android:layout_marginTop="5dp"  
        android:layout_weight="1"  
        android:text="Cancel"  
        android:textAllCaps="false"  
        android:backgroundTint="#FF7700"/>  
  
    >  
/>  
</LinearLayout>  
  
<Button  
    android:id="@+id/btDeleteEditUser"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_marginTop="5dp"  
    android:layout_gravity="center"  
    android:text="Delete"  
    android:textAllCaps="false"  
    android:textColor="#FFFFFF"  
    android:backgroundTint="#F44336"/>  
  
</LinearLayout>
```

## activity\_execute\_exercise.xml

```

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

    <ImageView
        android:layout_width="300dp"
        android:layout_height="300dp"
        android:src="@drawable/logo"
        android:layout_gravity="center"
        android:layout_marginTop="50dp"
        android:id="@+id/imgExercise"
        ></ImageView>

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="00 : 00 : 00"
        android:gravity="center"
        android:textSize="30sp"
        android:textStyle="bold"
        android:id="@+id/txtTime"
        ></TextView>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="70dp"
        android:layout_marginTop="100dp"
        android:weightSum="2"
        >

        <ImageView
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginHorizontal="30dp"
            android:layout_weight="1"
            android:backgroundTint="@color/white"
            android:textColor="@color/black"
            android:id="@+id/btnFinish"
            android:src="@drawable/stop_button"
            ></ImageView>

        <ImageView
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginHorizontal="30dp"
            android:layout_weight="1"
            android:backgroundTint="@color/white"
            android:textColor="@color/black"
            android:id="@+id/btnPlay"
            android:src="@drawable/play_button"
            ></ImageView>
    
```

</LinearLayout>

</LinearLayout>

activity\_feed\_back.xml

```

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

    <TextView
        android:id="@+id/txtFirstSentence"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="How much did you do?"
        android:layout_marginTop="50dp"
        android:textSize="18sp"
        android:textStyle="bold"
    ></TextView>

    <TextView
        android:id="@+id/txtTextInput"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
    ></TextView>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:layout_marginTop="35dp"
        android:weightSum="4"
    >

        <TextView
            android:layout_width="0dp"
            android:layout_height="40sp"
            android:text="Please enter here:"
            android:textSize="15sp"
            android:layout_weight="1.5"
        ></TextView>

        <NumberPicker
            android:id="@+id/numberPicker"
            android:layout_width="0dp"
            android:layout_height="50dp"
            android:layout_weight="1"
        ></NumberPicker>

    </LinearLayout>

    <TextView
        android:id="@+id/txtSecondSentence"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Rate the difficulty:"
    ></TextView>

```

```
        android:layout_marginTop="35dp"
        android:textSize="18sp"
        android:textStyle="bold"
    ></TextView>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:layout_marginTop="35dp"
    android:weightSum="8"
    >

    <ImageView
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:src="@drawable/hard_text"></ImageView>

    <RadioGroup
        android:id="@+id/radioGroup"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="6"
        android:orientation="horizontal">

        <RadioButton
            android:id="@+id/radio_1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:checked="true"
            android:text="1" />

        <RadioButton
            android:id="@+id/radio_2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="2" />

        <RadioButton
            android:id="@+id/radio_3"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="3" />

        <RadioButton
            android:id="@+id/radio_4"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="4" />

        <RadioButton
            android:id="@+id/radio_5"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="5" />
```

```
>
</RadioGroup>

<ImageView
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:src="@drawable/easy_text"></ImageView>

</LinearLayout>

<Button
    android:id="@+id/btnFinish"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="140dp"
    android:layout_gravity="center"
    android:textSize="30sp"
    android:text="Finish"
    android:textAllCaps="false"
    android:backgroundTint="#FF9800"></Button>

</LinearLayout>
```

## activity\_information.xml

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

    <ImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginTop="35dp"
        android:src="@drawable/logo"
    ></ImageView>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginTop="10dp"
        android:gravity="center"
        android:textSize="15sp"
        android:textColor="#6F777E"
        android:text="Register, pick an exercise and start training and sharing
your success!!!"
    ></TextView>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginTop="20dp"
        android:gravity="center"
        android:textSize="15sp"
        android:textColor="#6F777E"
        android:text="Start by seeing guides on how to do some exercises:"
    ></TextView>

    <ListView
        android:layout_width="match_parent"
        android:layout_height="400dp"
        android:id="@+id/listViewExercisesInfo"
    ></ListView>

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

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal"
            android:text="from"
        ></TextView>
```

```
        android:textColor="#aabbcc"
    />
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"
    android:text="FITNOTE"
    android:textSize="16sp"
    android:textColor="@color/colorPrimary"
    />

</LinearLayout>

</LinearLayout>
```

activity\_main.xml

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainScreenActivity"
    android:background="@color/white"
    android:orientation="vertical"
    >

    <ImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginTop="20dp"
        android:src="@drawable/logo"
        ></ImageView>

    <EditText
        android:id="@+id/etUserName"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter Username"
        android:layout_marginTop="35dp"
        ></EditText>

    <EditText
        android:id="@+id/etPassword"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="textPassword"
        android:hint="Enter Password"
        android:layout_marginTop="35dp"
        >
    </EditText>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:weightSum="2"
        android:layout_marginTop="35dp"
        >

        <Button
            android:id="@+id/btLogIn"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_marginTop="5dp"
            android:layout_weight="1"
            android:text="Log In"
            android:textAllCaps="false"
            android:backgroundTint="#FF9800" />

        <Button
            android:id="@+id/btRegister"
            >
    
```

```
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="5dp"
        android:layout_weight="1"
        android:text="Register"
        android:textAllCaps="false"
        android:backgroundTint="#FF9800"/>

    </LinearLayout>
</LinearLayout>
```

activity\_manager\_users.xml

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

    <ImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginTop="35dp"
        android:src="@drawable/logo"
    ></ImageView>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginTop="10dp"
        android:gravity="center"
        android:textSize="15sp"
        android:textColor="@color/colorPrimary"
        android:text="Welcome UNDETECTED USER"
        android:id="@+id/txtWelcomeManageUsers"
    ></TextView>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginTop="20dp"
        android:gravity="center"
        android:textSize="15sp"
        android:textColor="@color/colorPrimary"
        android:text="Here you can view users information"
        android:id="@+id/txtExplainManageUser"
    ></TextView>

    <SearchView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/searchViewManageUsersList"
        android:iconifiedByDefault="false"
        android:queryHint="Enter Here"
    ></SearchView>

    <ListView
        android:layout_width="match_parent"
        android:layout_height="400dp"
        android:id="@+id/listViewManagerUsers"
    ></ListView>

    <LinearLayout
```

```
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
    >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:text="from"
        android:textColor="#aabbc"
    />
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:text="FITNOTE"
        android:textSize="16sp"
        android:textColor="@color/colorPrimary"
    />

</LinearLayout>

</LinearLayout>
```

activity\_program\_user.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/white"
    android:orientation="vertical"
    tools:context=".ProgramUserActivity">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:weightSum="3"
        android:orientation="horizontal"
        android:layout_marginTop="35dp"
        >

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/txtMainTitle"
            android:text="Welcome"
            android:textSize="30dp"
            android:textColor="@color/black"
            android:layout_weight="1"
            ></TextView>

        <de.hdodenhof.circleimageview.CircleImageView
            android:id="@+id/imgProgramUserPic"
            android:layout_width="70dp"
            android:layout_height="70dp"
            android:layout_weight="2"
            android:src="@drawable/default_user"
            app:civ_border_width="3dp"
            app:civ_border_color="@color/black"
            ></de.hdodenhof.circleimageview.CircleImageView>

    </LinearLayout>

    <SearchView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/searchViewUserExerciseList"
        android:iconifiedByDefault="false"
        android:queryHint="Exercise Name"
        ></SearchView>

    <LinearLayout
        android:id="@+id/layoutOfListView"
        android:layout_width="wrap_content"
        android:layout_height="310dp"
        android:orientation="vertical"
        android:weightSum="3"
        >
```

```
<ListView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/listViewUserExercises"
    android:layout_weight="2"
    ></ListView>

<TextView
    android:id="@+id/txtEmptyListMessage"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="It's seems you don't have any exercises to do. you can
add an exercise with the plus button :D"
    android:textSize="20sp"
    android:layout_weight="1"
    android:visibility="gone"
    ></TextView>

</LinearLayout>

<LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    >

</LinearLayout>

<de.hdodenhof.circleimageview.CircleImageView
    android:id="@+id/btnAddExercise"
    android:layout_width="70dp"
    android:layout_height="70dp"
    android:layout_gravity="end"
    android:layout_marginLeft="150dp"
    android:src="@drawable/ic_plus"
    android:text="+"
    app:civ_border_width="3dp"
    app:civ_border_color="#00FFFFFF"
    ></de.hdodenhof.circleimageview.CircleImageView>

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Select Your Notification:"
    android:textSize="16sp"
    android:textColor="@color/colorPrimary"
    android:textStyle="bold"
    android:layout_marginTop="20dp"
    android:layout_gravity="bottom"
    ></TextView>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:weightSum="6"
    android:layout_marginTop="10dp"
    >
```

```
<TextView  
    android:layout_width="80dp"  
    android:layout_height="70dp"  
    android:layout_marginLeft="10dp"  
    android:layout_marginRight="25dp"  
    android:id="@+id/tv_timer"  
    android:textSize="15sp"  
    android:text="0:00"  
    android:layout_weight="1"  
    android:textStyle="bold"  
    android:gravity="center"  
    android:drawableTop="@drawable/ic_timer"  
    android:background="@android:drawable/editbox_background"  
></TextView>  
  
<Button  
    android:id="@+id/btnSetAlarm"  
    android:textSize="10sp"  
    android:layout_width="70dp"  
    android:layout_height="50dp"  
    android:layout_gravity="end"  
    android:layout_marginTop="10dp"  
    android:layout_marginRight="10dp"  
    android:layout_weight="1"  
    android:backgroundTint="#00BCD4"  
    android:textColor="@color/black"  
    android:textStyle="bold"  
    android:text="SET"></Button>  
<Button  
    android:id="@+id/btnCancelAlarm"  
    android:textSize="10sp"  
    android:layout_width="85dp"  
    android:layout_height="50dp"  
    android:layout_gravity="end"  
    android:layout_marginTop="10dp"  
    android:layout_marginRight="35dp"  
    android:layout_weight="1"  
    android:backgroundTint="#00BCD4"  
    android:textColor="@color/black"  
    android:textStyle="bold"  
    android:text="DELETE"></Button>  
  
</LinearLayout>  
  
</LinearLayout>
```

activity\_register.xml

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/white"
    android:orientation="vertical"
    tools:context=".RegisterActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Profile pic:"
        android:layout_marginTop="10sp"/>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="10sp">

        <ImageView
            android:id="@+id/imgProfilePic"
            android:layout_width="65dp"
            android:layout_height="65dp"
            android:src="@drawable/default_user"
            ></ImageView>

        <ImageView
            android:id="@+id/imgCemraButton"
            android:layout_width="35dp"
            android:layout_height="35dp"
            android:layout_gravity="bottom"
            android:layout_marginLeft="30dp"
            android:src="@drawable/cemra"
            ></ImageView>

    >
    </LinearLayout>

    <EditText
        android:id="@+id/etUserName"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Username"
        android:inputType="textPersonName"
        ></EditText>

    <EditText
        android:id="@+id/etPassword"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Password"
        android:inputType="textPassword"
        ></EditText>

    <LinearLayout
        android:layout_width="match_parent"

```

```
        android:layout_height="wrap_content"
        android:weightSum="2">

    <TextView
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Weight (kilograms):" />

    <TextView
        android:id="@+id/txtSeekBarWeight"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="40" />

    >
</LinearLayout>

<SeekBar
    android:id="@+id/seekBarWeight"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:max="200"
    android:min="40"
    android:layout_marginTop="15dp"/>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:weightSum="2">

    <TextView
        android:layout_width="0dp"
        android:layout_height="wrap_content"

        android:layout_weight="1"
        android:text="Height (centimeters):" />

    <TextView
        android:id="@+id/txtSeekBarHeight"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="50" />

    >
</LinearLayout>

<SeekBar
    android:id="@+id/seekBarHeight"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:max="250"
    android:min="50"
    android:layout_marginTop="15dp"/>

<TextView
```

```
        android:id="@+id/tv_date"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Select BirthDate:"
        android:textSize="20sp">></TextView>

<Button
    android:id="@+id/btDatePickerButton"
    style="?android:spinnerStyle"
    android:layout_width="250dp"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:onClick="openDatePicker"
    android:text="JAN 01 2020"
    android:textColor="#000000"
    android:textSize="30sp">></Button>

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Chose Gender:"
    android:textSize="10sp">></TextView>

<RadioGroup
    android:id="@+id/radioGroup"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content">

    <RadioButton
        android:id="@+id/radio_male"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:checked="true"
        android:text="Male" />

    <RadioButton
        android:id="@+id/radio_female"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Female" />

    <RadioButton
        android:id="@+id/radio_other"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Other" />

    >
</RadioGroup>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:weightSum="2">

    <Button
        android:id="@+id/btFinishReg"
```

```
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="5dp"
        android:layout_weight="1"
        android:enabled="false"
        android:text="Finish"
        android:textAllCaps="false"
        android:textColor="@color/black"
        android:backgroundTint="#4CAF50"/>>

    <Button
        android:id="@+id/btCancel"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="5dp"
        android:layout_weight="1"
        android:text="Cancel"
        android:textAllCaps="false"
        android:backgroundTint="#FF7700"/>>

    >
</LinearLayout>

</LinearLayout>
```

activity\_settings.xml

```

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

    <androidx.appcompat.widget.Toolbar
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:title="Change your'e info here:"
        app:titleTextColor="@color/white"
        android:background="@color/colorPrimary">
    </androidx.appcompat.widget.Toolbar>

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

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

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:orientation="horizontal"
                android:layout_marginTop="10dp">

                    <de.hdodenhof.circleimageview.CircleImageView
                        android:layout_width="88dp"
                        android:layout_height="88dp"
                        android:id="@+id/imgSettingUserPic"
                        android:src="@drawable/default_user"
                        app:civ_border_width="3dp"
                        app:civ_border_color="@color/black"
                        />

                    <LinearLayout
                        android:layout_width="match_parent"
                        android:layout_height="wrap_content"
                        android:orientation="vertical"
                        android:layout_marginStart="15dp">

                        <TextView
                            android:layout_width="wrap_content"
                            android:layout_height="wrap_content"
                            android:id="@+id/txtSettingsUserNameShow"
                            android:textSize="20sp"
                            android:text="userName"
                            />
                
```

```
<TextView  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:id="@+id/txtSettingsUserGenderShow"  
    android:textSize="15sp"  
    android:text="userGender"  
/>  
  
</LinearLayout>  
  
</LinearLayout>  
  
<View  
    android:layout_width="match_parent"  
    android:layout_height="1dp"  
    android:background="#FFFFFF"/>  
  
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:orientation="horizontal"  
    android:layout_marginTop="10dp"  
    android:id="@+id/layoutSettingsPassword"  
/>  
  
<ImageView  
    android:layout_width="30dp"  
    android:layout_height="30dp"  
    android:layout_marginStart="10dp"  
    android:src="@drawable/ic_baseline_phonelink_lock" />  
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:orientation="vertical"  
    android:layout_marginStart="15dp">  
  
<TextView  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_marginStart="10dp"  
    android:textSize="16sp"  
    android:text="Password"  
/>  
  
<TextView  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:id="@+id/txtSettingsPassword"  
    android:textSize="15sp"  
    android:text=""  
/>  
  
</LinearLayout>  
  
</LinearLayout>  
  
<LinearLayout  
    android:layout_width="match_parent"
```

```
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:layout_marginTop="10dp"
        android:id="@+id/layoutSettingsWeightAndHeight"
    >

    <ImageView
        android:layout_width="30dp"
        android:layout_height="30dp"
        android:layout_marginStart="10dp"
        android:src="@drawable/ic_baseline_architecture" />
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:layout_marginStart="15dp">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="10dp"
        android:textSize="16sp"
        android:text="Weight and Height"
    />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/txtSettingsWeightHeight"
        android:textSize="15sp"
        android:text=""
    />

    </LinearLayout>
</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:layout_marginTop="10dp"
    android:id="@+id/layoutSettingBirthDate"
>

    <ImageView
        android:layout_width="30dp"
        android:layout_height="30dp"
        android:layout_marginStart="10dp"
        android:src="@drawable/ic_baseline_date_range" />
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:layout_marginStart="15dp">

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

```
        android:layout_height="wrap_content"
        android:layout_marginStart="10dp"
        android:textSize="16sp"
        android:text="Birth Date"
    />

</LinearLayout>

</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:layout_marginTop="10dp"
    android:id="@+id/layoutSettingsGender"
>

    <ImageView
        android:layout_width="30dp"
        android:layout_height="30dp"
        android:layout_marginStart="10dp"
        android:src="@drawable/ic_baseline_wc" />
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:layout_marginStart="15dp">

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginStart="10dp"
            android:textSize="16sp"
            android:text="Gender"
        />

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/txtSettingsGenderEdit"
            android:textSize="15sp"
            android:text=""
        />

    </LinearLayout>
</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:layout_marginTop="10dp"
    android:id="@+id/layoutSettingPic"
>

    <ImageView
```

```
        android:layout_width="30dp"
        android:layout_height="30dp"
        android:layout_marginStart="10dp"
        android:src="@drawable/ic_baseline_account_circle" />
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:layout_marginStart="15dp">

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginStart="10dp"
            android:textSize="16sp"
            android:text="User Picture"
        />

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/txtSettingsPictureEdit"
            android:textSize="15sp"
            android:text=""
        />

    </LinearLayout>
</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:layout_marginTop="10dp"
    android:id="@+id/layoutSettingManagerUsers"
    android:visibility="gone">
    >

    <ImageView
        android:layout_width="30dp"
        android:layout_height="30dp"
        android:layout_marginStart="10dp"
        android:src="@drawable/ic_baseline_wrench" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="25dp"
        android:textSize="16sp"
        android:text="option: manage users"
        android:id="@+id/txtAdminOPTitleSettings"
    />

</LinearLayout>

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

```
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:text="from"
        android:textColor="#aabbcc"
    />
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"
    android:text="FITNOTE"
    android:textSize="16sp"
    android:textColor="@color/colorPrimary"
/>

</LinearLayout>
</ScrollView>

</LinearLayout>
```

activity\_share.xml

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

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/txtShareWithContactScreen"
        android:layout_marginTop="35dp"
        android:text="Choose contact to send message to"
        android:gravity="center"
        android:layout_gravity="center"
        android:textSize="20sp"
        android:textColor="#FF8C00"
        android:textStyle="bold"
    ></TextView>

    <SearchView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/searchViewContactList"
        android:iconifiedByDefault="false"
        android:queryHint="Contact Name"
    ></SearchView>

    <androidx.recyclerview.widget.RecyclerView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/recyclerViewContacts"
        tools:listitem="@layout/item_contact"
        android:layout_marginBottom="35dp"
    ></androidx.recyclerview.widget.RecyclerView>

</LinearLayout>
```

## activity\_statistics.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".StatisticsActivity"
    android:orientation="vertical"
    android:layout_margin="10dp"
    android:backgroundTint="@color/colorPrimary">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:weightSum="2"
        android:layout_marginBottom="10dp"
        android:orientation="horizontal">

        <de.hdodenhof.circleimageview.CircleImageView
            android:id="@+id/imgUserPicture"
            android:layout_width="70dp"
            android:layout_height="70dp"
            android:layout_weight="1"
            android:src="@drawable/default_user"
            app:civ_border_width="3dp"
            app:civ_border_color="@color/black"
            android:layout_gravity="center"
            ></de.hdodenhof.circleimageview.CircleImageView>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="GRAPHS"
            android:layout_gravity="center"
            android:textSize="30sp"
            android:textColor="@color/colorPrimary"
            android:layout_marginTop="20dp"
            android:layout_weight="1"
            ></TextView>

    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:weightSum="2"
        android:layout_marginBottom="10dp"
        android:orientation="horizontal">

        <Spinner
            android:id="@+id/exercise_spinner"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            ></Spinner>

    </LinearLayout>
```

```
<Spinner  
    android:id="@+id/year_spinner"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_weight="1"  
    ></Spinner>  
  
</LinearLayout>  
  
<Button  
    android:id="@+id/btnSubmit"  
    android:layout_width="75dp"  
    android:layout_height="45dp"  
    android:layout_gravity="center"  
    android:backgroundTint="#FF9800"  
    android:text="Submit"  
    android:textAllCaps="false"  
    android:textColor="@color/white"  
    android:textSize="10sp"></Button>  
  
<com.github.mikephil.charting.charts.BarChart  
    android:layout_width="match_parent"  
    android:layout_height="320dp"  
    android:id="@+id/barChart"  
    ></com.github.mikephil.charting.charts.BarChart>  
  
<TextView  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="----Below is the meaning of the colors - user's rating----"  
    android:textColor="@color/colorPrimary"  
    android:textStyle="bold"  
    android:textSize="10sp"  
    android:layout_gravity="center"  
    android:gravity="center"  
    ></TextView>  
  
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:weightSum="5"  
    android:layout_marginBottom="10dp"  
    android:orientation="horizontal">  
  
<TextView  
    android:layout_width="50dp"  
    android:layout_height="60dp"  
    android:text="Hard"  
    android:textSize="8sp"  
    android:layout_weight="1"  
    android:textStyle="bold"  
    android:gravity="center"  
    android:background="@android:drawable/editbox_background"  
    android:backgroundTint="@color/Hard"  
    ></TextView>  
  
<TextView
```

```
        android:layout_width="50dp"
        android:layout_height="60dp"
        android:text="Hard-Medium"
        android:textSize="8sp"
        android:layout_weight="1"
        android:textStyle="bold"
        android:gravity="center"
        android:background="@android:drawable/editbox_background"
        android:backgroundTint="@color/HardMedium"
    ></TextView>

<TextView
    android:layout_width="50dp"
    android:layout_height="60dp"
    android:text="Medium"
    android:textSize="8sp"
    android:layout_weight="1"
    android:textStyle="bold"
    android:gravity="center"
    android:background="@android:drawable/editbox_background"
    android:backgroundTint="@color/Medium"
></TextView>

<TextView
    android:layout_width="50dp"
    android:layout_height="60dp"
    android:text="Easy-Medium"
    android:textSize="8sp"
    android:layout_weight="1"
    android:textStyle="bold"
    android:gravity="center"
    android:background="@android:drawable/editbox_background"
    android:backgroundTint="@color/EasyMedium"
></TextView>

<TextView
    android:layout_width="50dp"
    android:layout_height="60dp"
    android:text="Easy"
    android:textSize="8sp"
    android:layout_weight="1"
    android:textStyle="bold"
    android:gravity="center"
    android:background="@android:drawable/editbox_background"
    android:backgroundTint="@color/Easy"
></TextView>

</LinearLayout>
</LinearLayout>
```

## activity\_view\_done\_exercises.xml

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

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="35dp"
        android:weightSum="2"
        android:layout_marginBottom="10dp"
        android:orientation="horizontal">

        <de.hdodenhof.circleimageview.CircleImageView
            android:id="@+id/imgUserPictureExercisesResultsScreen"
            android:layout_width="70dp"
            android:layout_height="70dp"
            android:layout_weight="1"
            android:src="@drawable/default_user"
            app:civ_border_width="3dp"
            app:civ_border_color="@color/black"
            android:layout_gravity="center"
            ></de.hdodenhof.circleimageview.CircleImageView>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Results"
            android:layout_gravity="center"
            android:textSize="30sp"
            android:textColor="@color/colorPrimary"
            android:layout_marginTop="20dp"
            android:layout_weight="1"
            ></TextView>

    </LinearLayout>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/txtBMIDoneExercises"
        android:text="BMI of "
        android:textSize="20sp"
        android:textColor="@color/colorPrimary"
        ></TextView>

    <SearchView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/searchViewUserExerciseDoneList"
        android:iconifiedByDefault="false"
        android:queryHint="Enter Here"
        ></SearchView>
```

```
<LinearLayout
    android:id="@+id/layoutOfListView"
    android:layout_width="wrap_content"
    android:layout_height="300dp"
    android:orientation="vertical"
    android:weightSum="3"
    >

    <ListView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/listViewUserExercisesDone"
        android:layout_weight="2"
        ></ListView>

    <TextView
        android:id="@+id/txtEmptyListMessage"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="It's seems you don't have any exercises done. When you
finish you're exercise you will be able to view the results here."
        android:textSize="20sp"
        android:layout_weight="1"
        android:visibility="gone"
        ></TextView>

    </LinearLayout>

</LinearLayout>
```

activity\_whats\_app\_send.xml

```

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

    <TextView
        android:text="Name: "
        android:layout_height="wrap_content"
        android:layout_width="match_parent"
        android:id="@+id/tvName"
        android:textColor="@color/colorPrimary"
        android:textSize="30sp"/>

    <TextView
        android:text="Phone number: "
        android:layout_height="wrap_content"
        android:layout_width="match_parent"
        android:id="@+id/tvPhone"
        android:textColor="@color/colorPrimary"
        android:textSize="30sp"/>

    <EditText
        android:id="@+id/etWhatsAppMessage"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter Your message here"
        android:layout_marginTop="35dp"
        android:inputType="textMultiLine"
        android:scrollbars="vertical"
        android:minLines="3"
        android:maxLines="10"
        />

    <Button
        android:id="@+id/btSendWhatsAppMessage"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="5dp"
        android:layout_gravity="center"
        android:text="SEND"
        android:textAllCaps="false"
        android:textColor="@color/black"
        android:textStyle="bold"
        android:backgroundTint="#4CAF50"/>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="35dp"
        android:layout_gravity="center"
        android:text="Your message will be combined with your exercises results
from this week"
        android:textColor="@color/black"

```

```
        android:textStyle="bold"
        android:gravity="center"
    ></TextView>
</LinearLayout>
```

exercise\_layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    android:padding="16dp">

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="75dp"
        android:layout_height="75dp"
        android:src="@drawable/pushup"
        ></ImageView>

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

        <TextView
            android:id="@+id/txtMainTitle"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Main Title"
            android:textColor="@color/black"
            android:textStyle="bold"
            android:layout_margin="5dp"
            android:textSize="18sp"
            ></TextView>

        </LinearLayout>
    </LinearLayout>
```

exercise\_with\_description\_layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="16dp">

    <ImageView
        android:id="@+id/imageViewOfExerciseInDescription"
        android:layout_width="75dp"
        android:layout_height="75dp"
        android:src="@drawable/pushup"
        android:layout_gravity="center"
    ></ImageView>

    <TextView
        android:id="@+id/txtExerciseNameInDescription"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="Exercise name"
        android:textColor="@color/colorPrimary"
        android:textStyle="bold"
        android:layout_margin="5dp"
        android:textSize="25sp"
    ></TextView>

    <TextView
        android:id="@+id/txtExerciseDescription"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="Exercise description"
        android:textColor="@color/black"
        android:layout_margin="5dp"
        android:textSize="15sp"
    ></TextView>

</LinearLayout>
```

gender\_settings\_alert.xml

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

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Chose Gender:"
        android:textColor="@color/colorPrimary"
        android:textSize="20sp"></TextView>

    <RadioGroup
        android:id="@+id/radioGroupSettingsGender"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content">

        <RadioButton
            android:id="@+id/radio_maleSettingsGender"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:checked="true"
            android:text="Male" />

        <RadioButton
            android:id="@+id/radio_femaleSettingsGender"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Female" />

        <RadioButton
            android:id="@+id/radio_otherSettingsGender"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Other" />

    >
</RadioGroup>

</LinearLayout>
```

item\_contact.xml

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

    <ImageView
        android:layout_width="80dp"
        android:layout_height="80dp"
        android:src="@drawable/ic_baseline_person"
        android:id="@+id/iv_imageContact"
    ></ImageView>

    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp">

        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:id="@+id/tv_nameContact"
            android:textSize="18sp"
            android:textColor="@color/colorPrimary"
            android:paddingTop="8dp"
            android:paddingBottom="8dp"
            android:text="name"
        ></TextView>

        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:id="@+id/tv_numberContact"
            android:textSize="14sp"
            android:textColor="@color/colorPrimary"
            android:paddingTop="8dp"
            android:paddingBottom="8dp"
            android:text="number"
        ></TextView>

    </LinearLayout>
</LinearLayout>
```

profilepic\_settings\_alert.xml

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

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Profile pic:"
        android:layout_marginTop="10sp"/>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="10sp">

        <ImageView
            android:id="@+id/imgSettingsEditProfilePic"
            android:layout_width="65dp"
            android:layout_height="65dp"
            android:src="@drawable/default_user"
            ></ImageView>

        <ImageView
            android:id="@+id/imgSettingsEditCemraButton"
            android:layout_width="35dp"
            android:layout_height="35dp"
            android:layout_gravity="bottom"
            android:layout_marginLeft="30dp"
            android:src="@drawable/cemra"
            ></ImageView>

    >
</LinearLayout>

</LinearLayout>
```

## user\_exercise\_done\_layout.xml

```

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

    <ImageView
        android:id="@+id/imageViewOfUserExerciseDone"
        android:layout_width="75dp"
        android:layout_height="75dp"
        android:src="@drawable/pushup"
        android:layout_gravity="center"
    ></ImageView>

    <TextView
        android:id="@+id/txtExerciseOfUserExerciseDone"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="Exercise name: "
        android:textColor="@color/colorPrimary"
        android:textStyle="bold"
        android:layout_margin="5dp"
        android:textSize="25sp"
    ></TextView>

    <TextView
        android:id="@+id/txtDateOfUserExerciseDone"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="Date: "
        android:textColor="@color/black"
        android:layout_margin="5dp"
        android:textSize="15sp"
    ></TextView>

    <TextView
        android:id="@+id/txtTimeOfUserExerciseDone"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="Time: "
        android:textColor="@color/black"
        android:layout_margin="5dp"
        android:textSize="15sp"
    ></TextView>

    <TextView
        android:id="@+id/txtRatingOfUserExerciseDone"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="Rating: "
        android:textColor="@color/black"
        android:layout_margin="5dp"
    ></TextView>

```

```
        android:textSize="15sp"
    ></TextView>

<TextView
    android:id="@+id/txtRepetitionOfUserExerciseDone"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:gravity="center"
    android:text="Repetition: "
    android:textColor="@color/black"
    android:layout_margin="5dp"
    android:textSize="15sp"
></TextView>

</LinearLayout>
```

user\_exercise\_layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    android:padding="16dp"
    android:weightSum="3">

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="0dp"
        android:layout_height="75dp"
        android:src="@drawable/pushup"
        android:layout_weight="1"
    ></ImageView>

    <TextView
        android:id="@+id/txtMainTitle"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1.5"
        android:text="Main Title"
        android:textColor="@color/black"
        android:textStyle="bold"
        android:layout_margin="5dp"
        android:textSize="18sp"
    ></TextView>

    <Button
        android:id="@+id/btnDelete"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="0.5"
        android:backgroundTint="#FFFFFF"
        android:text="DELETE"
        android:textColor="@color/black"
        android:layout_marginHorizontal="10dp"
    ></Button>

</LinearLayout>
```

user\_layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="300dp"
    android:layout_height="match_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:orientation="horizontal"
    android:layout_marginTop="10dp"
    >

    <de.hdodenhof.circleimageview.CircleImageView
        android:layout_width="88dp"
        android:layout_height="88dp"
        android:id="@+id/imgUserPicUserLayout"
        android:src="@drawable/default_user"
        app:civ_border_width="3dp"
        app:civ_border_color="@color/black"
        />

    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:layout_marginStart="15dp">

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/txtUserNameShowUserLayout"
            android:textSize="20sp"
            android:text="userName"
            />

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/txtUserGenderShowUserLayout"
            android:textSize="15sp"
            android:text="userGender"
            />

        </LinearLayout>
    </LinearLayout>
```

weight\_height\_settings\_alert.xml

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

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:weightSum="2">

        <TextView
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Weight (kilograms):" />

        <TextView
            android:id="@+id/txtSeekBarSettingsWeight"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="40" />

    >
</LinearLayout>

<SeekBar
    android:id="@+id/seekBarSettingsWeight"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:max="200"
    android:min="40"
    android:layout_marginTop="15dp"/>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:weightSum="2">

    <TextView
        android:layout_width="0dp"
        android:layout_height="wrap_content"

        android:layout_weight="1"
        android:text="Height (centimeters):" />

    <TextView
        android:id="@+id/txtSeekBarSettingsHeight"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="50" />

    >
</LinearLayout>
```

```
<SeekBar  
    android:id="@+id/seekBarSettingsHeight"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:max="250"  
    android:min="50"  
    android:layout_marginTop="15dp"/>/  
  
</LinearLayout>
```

menu

main\_menu.xml

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

    <item
        android:title="about"
        android:id="@+id/aboutItem"
        ></item>

    <item
        android:title="exit"
        android:id="@+id/exitItem"
        ></item>

</menu>
```

program\_user\_menu.xml

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

    <item
        android:title="graphs"
        android:id="@+id/graphsItem"
        android:icon="@drawable/ic_baseline_show_chart"
        app:showAsAction="always"
        android:iconTint="@color/black"
    ></item>

    <item
        android:title="results"
        android:id="@+id/exercisesResultsItem"
        android:icon="@drawable/ic_results"
        app:showAsAction="always"
        android:iconTint="@color/black"
    ></item>

    <item
        android:title="share"
        android:id="@+id/shareItem"
        android:icon="@drawable/ic_share"
        app:showAsAction="always"
        android:iconTint="@color/black"
    ></item>

    <item
        android:title="settings"
        android:id="@+id/settingsItem"
        android:icon="@drawable/ic_settings"
        android:iconTint="@color/black"
        app:showAsAction="always"
    ></item>

</menu>
```

**mipmap****ic\_launcher file**

ic\_launcher.png



ic\_launcher.xml

```
<?xml version="1.0" encoding="utf-8"?>
<adaptive-icon xmlns:android="http://schemas.android.com/apk/res/android">
    <background android:drawable="@drawable/ic_launcher_background" />
    <foreground android:drawable="@drawable/ic_launcher_foreground" />
</adaptive-icon>
```

**ic\_launcher\_round file**

ic\_launcher\_round.png



ic\_launcher\_round.xml

```
<?xml version="1.0" encoding="utf-8"?>
<adaptive-icon xmlns:android="http://schemas.android.com/apk/res/android">
    <background android:drawable="@drawable/ic_launcher_background" />
    <foreground android:drawable="@drawable/ic_launcher_foreground" />
</adaptive-icon>
```

**raw**

phantom.mp3  
shoping.mp3  
speedrun.mp3  
speedrun2.mp3  
speedrun3.mp3

**values**

colors.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="purple_200">#FFBB86FC</color>
    <color name="purple_500">#FF6200EE</color>
    <color name="purple_700">#FF3700B3</color>
    <color name="teal_200">#FF03DAC5</color>
    <color name="teal_700">#FF018786</color>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFFF</color>
    <color name="colorPrimary">#009688</color>
    <color name="Hard">#F44336</color>
    <color name="HardMedium">#FF9800</color>
    <color name="Medium">#FFEB3B</color>
    <color name="EasyMedium">#CDDC39</color>
    <color name="Easy">#4CAF50</color>
</resources>
```

dimens.xml

```
<resources>
    <dimen name="fab_margin">16dp</dimen>
</resources>
```

strings.xml

```
<resources>
    <string name="app_name">FitNote</string>
    <string name="title_activity_settings">Settings</string>

    <!-- Preference Titles -->
    <string name="messages_header">Messages</string>
    <string name="sync_header">Sync</string>

    <!-- Messages Preferences -->
    <string name="signature_title">Your signature</string>
    <string name="reply_title">Default reply action</string>

    <!-- Sync Preferences -->
    <string name="sync_title">Sync email periodically</string>
    <string name="attachment_title">Download incoming attachments</string>
    <string name="attachment_summary_on">Automatically download attachments for incoming emails
    </string>
    <string name="attachment_summary_off">Only download attachments when manually requested</string>
    <string name="title_activity_information">InformationActivity</string>

</resources>
```

themes.xml

```

<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Theme.FitNote13022021"
parent="Theme.MaterialComponents.DayNight.DarkActionBar">
        <!-- Primary brand color. -->
        <item name="colorPrimary">@color/colorPrimary</item>
        <item name="colorPrimaryVariant">@color/colorPrimary</item>
        <item name="colorOnPrimary">@color/white</item>
        <!-- Secondary brand color. -->
        <item name="colorSecondary">@color/teal_200</item>
        <item name="colorSecondaryVariant">@color/teal_700</item>
        <item name="colorOnSecondary">@color/black</item>
        <!-- Status bar color. -->
        <item name="android:statusBarColor"
tools:targetApi="1">?attr/colorPrimaryVariant</item>
        <!-- Customize your theme here. -->
    </style>

    <style name="Theme.FitNote13022021.NoActionBar">
        <item name="windowActionBar">false</item>
        <item name="windowNoTitle">true</item>
    </style>

    <style name="Theme.FitNote13022021.AppBarOverlay"
parent="ThemeOverlay.AppCompat.Dark.ActionBar" />

    <style name="Theme.FitNote13022021.PopupOverlay"
parent="ThemeOverlay.AppCompat.Light" />
</resources>
```

night\themes.xml

```

<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Theme.FitNote13022021"
parent="Theme.MaterialComponents.DayNight.DarkActionBar">
        <!-- Primary brand color. -->
        <item name="colorPrimary">@color/purple_200</item>
        <item name="colorPrimaryVariant">@color/purple_700</item>
        <item name="colorOnPrimary">@color/black</item>
        <!-- Secondary brand color. -->
        <item name="colorSecondary">@color/teal_200</item>
        <item name="colorSecondaryVariant">@color/teal_200</item>
        <item name="colorOnSecondary">@color/black</item>
        <!-- Status bar color. -->
        <item name="android:statusBarColor"
tools:targetApi="1">?attr/colorPrimaryVariant</item>
        <!-- Customize your theme here. -->
    </style>
</resources>
```

## Gradle Scripts

project build

```
// Top-Level build file where you can add configuration options common to all sub-
// projects/modules.
buildscript {
    repositories {
        google()
        jcenter()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle:4.1.3'

        // NOTE: Do not place your application dependencies here; they belong
        // in the individual module build.gradle files
    }
}

allprojects {
    repositories {
        google()
        jcenter()
        maven { url 'https://jitpack.io' }
    }
}

task clean(type: Delete) {
    delete rootProject.buildDir
}
```

```
module build
```

```

plugins {
    id 'com.android.application'
}

android {
    compileSdkVersion 30
    buildToolsVersion "30.0.2"

    defaultConfig {
        applicationId "com.example.fitnote13022021"
        minSdkVersion 19
        targetSdkVersion 30
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'),
            'proguard-rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
}

dependencies {

    // Dependency for circular ImageView
    implementation 'de.hdodenhof:circleimageview:3.1.0'

    implementation 'androidx.appcompat:appcompat:1.2.0'
    implementation 'com.google.android.material:material:1.3.0'
    implementation 'com.google.android.material:material:1.2.1'
    implementation 'androidx.constraintlayout:constraintlayout:2.0.4'
    implementation 'com.github.PhilJay:MPAndroidChart-Realm:v3.0.3@aar'
    implementation 'com.github.PhilJay:MPAndroidChart:v3.0.3'
    implementation 'androidx.preference:preference:1.1.1'
    implementation 'androidx.navigation:navigation-fragment:2.3.4'
    implementation 'androidx.navigation:navigation-ui:2.3.4'
    testImplementation 'junit:junit:4.+'
    androidTestImplementation 'androidx.test.ext:junit:1.1.2'
    implementation 'com.github.PhilJay:MPAndroidChart:v3.1.0'

    androidTestImplementation 'androidx.test.espresso:espresso-core:3.3.0'

}

```

## grable-wrapper.properties

```
#Sat Feb 13 09:48:45 IST 2021
distributionBase=GRADLE_USER_HOME
distributionPath=wrapper/dists
zipStoreBase=GRADLE_USER_HOME
zipStorePath=wrapper/dists
distributionUrl=https\://services.gradle.org/distributions/gradle-6.5-bin.zip
proguard-rules.pro

# Add project specific ProGuard rules here.
# You can control the set of applied configuration files using the
# proguardFiles setting in build.gradle.
#
# For more details, see
#   http://developer.android.com/guide/developing/tools/proguard.html

# If your project uses WebView with JS, uncomment the following
# and specify the fully qualified class name to the JavaScript interface
# class:
#-keepclassmembers class fqcn.of.javascript.interface.for.webview {
#    public *;
#}

# Uncomment this to preserve the Line number information for
# debugging stack traces.
#-keepattributes SourceFile,LineNumberTable

# If you keep the line number information, uncomment this to
# hide the original source file name.
#-renamesourcefileattribute SourceFile
```

## gradle.properties

```
# Project-wide Gradle settings.
# IDE (e.g. Android Studio) users:
# Gradle settings configured through the IDE *will override*
# any settings specified in this file.
# For more details on how to configure your build environment visit
# http://www.gradle.org/docs/current/userguide/build_environment.html
# Specifies the JVM arguments used for the daemon process.
# The setting is particularly useful for tweaking memory settings.
org.gradle.jvmargs=-Xmx2048m -Dfile.encoding=UTF-8
# When configured, Gradle will run in incubating parallel mode.
# This option should only be used with decoupled projects. More details, visit
#
# http://www.gradle.org/docs/current/userguide/multi_project_builds.html#sec:decoupled_projects
# org.gradle.parallel=true
# AndroidX package structure to make it clearer which packages are bundled with
# the
# Android operating system, and which are packaged with your app's APK
# https://developer.android.com/topic/libraries/support-library/androidx-rn
android.useAndroidX=true
# Automatically convert third-party Libraries to use AndroidX
android.enableJetifier=true
```

Settings.gradle (FitNote13.02.2021)

```
include ':app'  
rootProject.name = "FitNote13.02.2021"                                local.properties  
  
## This file must *NOT* be checked into Version Control Systems,  
# as it contains information specific to your local configuration.  
#  
# Location of the SDK. This is only used by Gradle.  
# For customization when using a Version Control System, please read the  
# header note.  
#Thu Apr 22 15:29:58 IDT 2021  
sdk.dir=C:\\\\Users\\\\Home\\\\AppData\\\\Local\\\\Android\\\\Sdk
```