

Brain Battle

מגיש: יהודה אליסף

ת.ז. 328193602

מורה: חגי סוויד

חלופה: תכנות לטלפונים ניידים

תאריך הגשה: 12.6.2023



תוכן

5	מבוא
5	הרקע לפרויקט
5	תהליך המחקר
5	טכנולוגיות
5	אתגרים מרכזיים
6	תיאור תחום הידע
6	אובייקטים נחוצים
6	סוגי נתונים
6	פעולות על המידע
7	ארכיטקטורה
7	מסכי הפרויקט
7	מסך הרשמה והתחברות (מסך הפתיחה)
8	מסך יצירת משחק
9	מסך קוד המשחק
10	מסך הצטרפות למשחק
11	מסך המשחק
12	מסך סיום משחק
13	מסך ניקוד
14	מסך הגדרות
15	תרשים מסכים
15	
15	
16	מחלקות הפרויקט
17	מימוש הפרויקט
17	מחלקות מסכים (Fragment/Activity)
17	LoginActivity
19	MainMenuActivity
21	NewGameFragment
24	JoinGameFragment
25	SettingsFragment
26	ScoreFragment
28	GameIdFragment

29.....	GameActivity
34.....	EndGameFragment
35.....	מחלקות עזר
35.....	Game
35.....	Question
36.....	User
36.....	Player
36.....	MusicService
38.....	HttpQuestionFetcher
41.....	AnswerRecorder
44.....	MusicService
44.....	GameViewModel
46.....	ScoreViewHolder
46.....	ScoreListAdapter
47.....	קבצי תצורה
47.....	Layout קבצי
48.....	Menu קבצי
48.....	Navigation קבצי
48.....	בסיסי נתונים
48.....	SharedPreferences
49.....	Firestore
50.....	Open Trivia DB
51.....	מדריך למשתמש
51.....	תיאור כללי
51.....	הגבלות
51.....	הרשאות
51.....	הצהרות
51.....	גרסת Android מינימלית
51.....	גרסאות ומכשירים שעליהם נבדקה האפליקציה
52.....	רפלקציה
53.....	ביביליוגרפיה
54.....	נספחים
54.....	קוד הפרויקט:

מבוא

הרקע לפרויקט

הפרויקט Brain Battle (קרב מוחות) הוא משחק טריוויה ל-2 שחקנים. השחקנים מתחרים ביניהם אונליין, עונים על שאלות טריוויה שונות וצוברים נקודות. בסיום המשחק, השחקנים יכולים לראות סטטיסטיקות שונות כמו מספר התשובות הנכונות והשגויות, ואת מצב הניקוד שלהם ביחס לשחקנים אחרים.

המשחק מיועד לטווח גילאים רחב, משום שיש אפשרות לבחור את רמת הקושי הרצויה (כמובן שרמת הקושי משפיעה גם על הניקוד). להערכתי, תחום הגילאים המומלץ הוא 10 ומעלה.

בחרתי ליצור דווקא משחק כי רציתי ליהנות כמה שיותר מיצירת הפרויקט, ויצירת משחקים מוסיפה גורם של הנאה לפיתוח. בחרתי דווקא בטריוויה כי אני אוהב את המשחק, ונהנה לבחון את הידע הכללי שלי בתחומים שונים.

תהליך המחקר

כיום, ישנן אפליקציות טריוויה רבות בשוק, גם ל-2 שחקנים וגם למספרי שחקנים שונים. עם זאת, זה לא מפריע לי, כי המטרה העיקרית שלי בפיתוח במשחק היא להעמיק את הידע שלי בפיתוח אפליקציות.

טכנולוגיות

במהלך יצירת הפרויקט השתמשתי במספר טכנולוגיות שאינן חלק מתוכנית הלימודים:

1. HTTP – פרוטוקול תקשורת שמיועד לשליחת וקבלת מידע לאתרי אינטרנט ומהם. השתמשתי ב-HTTP כדי להוריד שאלות טריוויה ממסד נתונים בשם [Open Trivia Database](#), במטרה להשיג כמות גדולה של שאלות בנושאים מגוונים ורמות קושי שונות. פרמטרי כמו מספר השאלות, רמת הקושי וכו' נשלחים ב-HTTP בעזרת פרמטרי GET.
2. JSON Deserialization – השאלות במסד הנתונים מתקבלות בפורמט JSON. כדי להפוך את השאלות שהתקבלו לאובייקט Question (אובייקט שיצרתי שמייצג שאלה, עליו אפרט בהמשך) יש צורך בדה-סריאליזציה של המידע, כלומר הפיכה שלו ל-JSON לאובייקט קוד.
3. Firebase – שירות של גוגל שמאפשר מספר פיצ'רים כמו ניהול ההתחברות וההרשמה של משתמשים, ושרת אחסון בשם Firestore שמאפשר לשמור מידע בענן. השתמשתי ב-Firebase בשביל לשמור את נתונים המשחקים ולסנכרן אותם בזמן אמת, ובשביל לשמור את רשימת המשתמשים ופרטים שלהם.
4. SpeechToText – הפיכה של דיבור אנושי לסטרינג. השתמשתי בה כדי לאפשר למשתמש להקליט את התשובה (אחת, שתיים, שלוש, ארבע) במקום ללחוץ על הכפתור.

אתגרים מרכזיים

היו לי לא מעט אתגרים בזמן העבודה על הפרויקט.

בגלל שהמשחק נערך אונליין בין 2 שחקנים, ואף אחד מהשחקנים לא יכול לשחק בלי השני, הייתי צריך להשתמש ב-2 מכשירי פלאפון. לכן, יצרתי אמולטור שמדמה מכשיר פלאפון, והרצתי את האפליקציה גם בו וגם במכשיר הפיזי.

קושי נוסף שהתמודדתי איתו הוא נגרם מכך שהמשחק דורש חיבור אינטרנט רציף. אם באמצע המשחק אחד השחקנים מתנתק, האפליקציה יכולה להיתקע או לקרוס. כדי לפתור את הבעיה, הוספתי לכל תקשורת עם Firestore פונקציית OnFailure שקובעת מה לעשות במידה וחיבור האינטרנט קרס (למשל: הצגת הודעת שגיאה למשתמש וחזרה למסך הראשי). בנוסף, ההכנסתי את בקשת ה-HTTP לבלוק try catch, ובמידה וקרתה שגיאה – הודעתי למשתמש שלא ניתן להוריד את השאלות.

תיאור תחום הידע

אובייקטים נחוצים

Game: אובייקט הכולל את כל המידע של המשחק. כולל בתוכו את 2 השחקנים ורשימה של שאלות (על מבנה השחקנים והשאלות אפרט בהמשך), ואת ID של המשחק.

User: מייצג משתמש. כולל בתוכו את המידע של המשתמש (שם משתמש, ID וסטטיסטיקות של ניקוד).

Player: מייצג שחקן במשחק. יורש מ-User. מוסיף למשתמש רשימה של השאלות הנכונות והשגויות שהוא ענה במשחק.

Question: מייצג שאלה. כולל בתוכו את התשובות השגויות והתשובה הנכונה, קטגוריית השאלה ורמת הקושי שלה.

סוגי נתונים

בתוך מחלקת Game, השאלות שמורות ברשימה של אובייקטים מסוג Question.

במחלקת Question, התשובות שמורות ברשימה של סטרינגים, והאינדקס של התשובה הנכונה שמור ב-`int`. הקטגוריה ורמת הקושי שמורות כל אחת בenum משל עצמה:

```
enum DifficultyLevel{
    EASY,
    MEDIUM,
    HARD
}
enum Category{
    ALL,
    GENERAL_KNOWLEDGE,
    SCIENCE,
    COMPUTER_SCIENCE
}
```

פעולות על המידע

Game: יצירת משחק, סיום משחק, גטרים וסטרים.

User: יצירת משתמש, מחיקת משתמש, גטרים וסטרים.

Player: יצירת שחקן, חישוב של כמות התשובות הנכונות/שגויות במשחק, מתודת `equals()`, גטרים וסטרים.

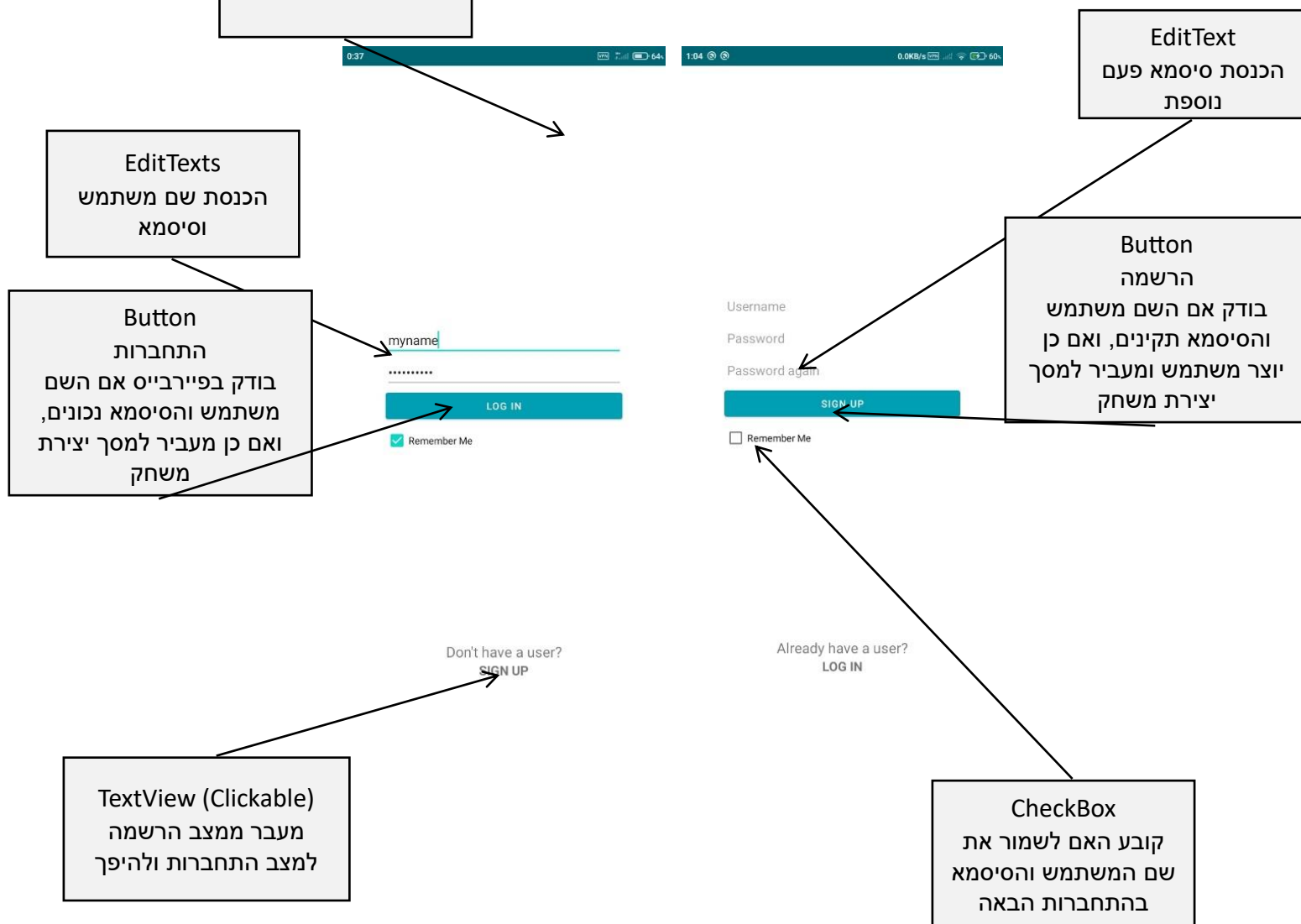
Question: יצירת שאלה, גטרים וסטרים.

ארכיטקטורה

מסכי הפרויקט

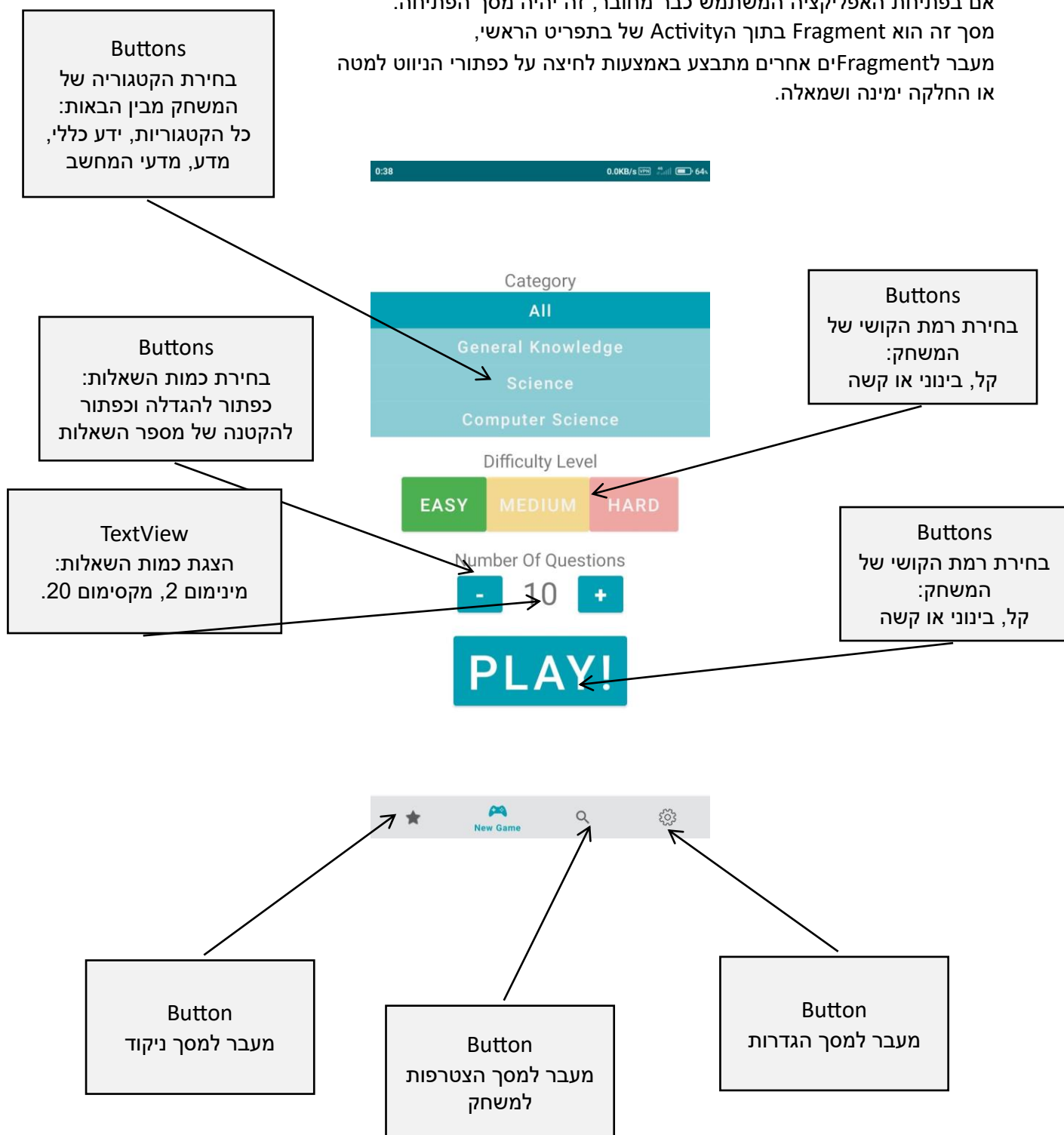
מסך הרשמה והתחברות (מסך הפתיחה)

במצב התחברות, המסך מציג 2 EditTextים בשביל שם משתמש וסיסמא, במצב הרשמה, המסך מציג EditText נוסף שמיועד להכנסת הסיסמא שוב. מעבר בין המצבים מתבצע באמצעות לחיצה על label שבתחתית המסך



מסך יצירת משחק

מאפשר למשתמש ליצור משחק חדש, ולבחור את הקטגוריה, רמת הקושי ומספר השאלות. אם בפתחת האפליקציה המשתמש כבר מחובר, זה יהיה מסך הפתיחה. מסך זה הוא Fragment בתוך Activity של בתפריט הראשי, מעבר לFragment אחרים מתבצע באמצעות לחיצה על כפתורי הניווט למטה או החלקה ימינה ושמאלה.



מסך קוד המשחק

לאחר יצירת המשחק, יופיע מסך ובו קוד אותו צריך המשתמש השני להכניס כדי להתחבר.

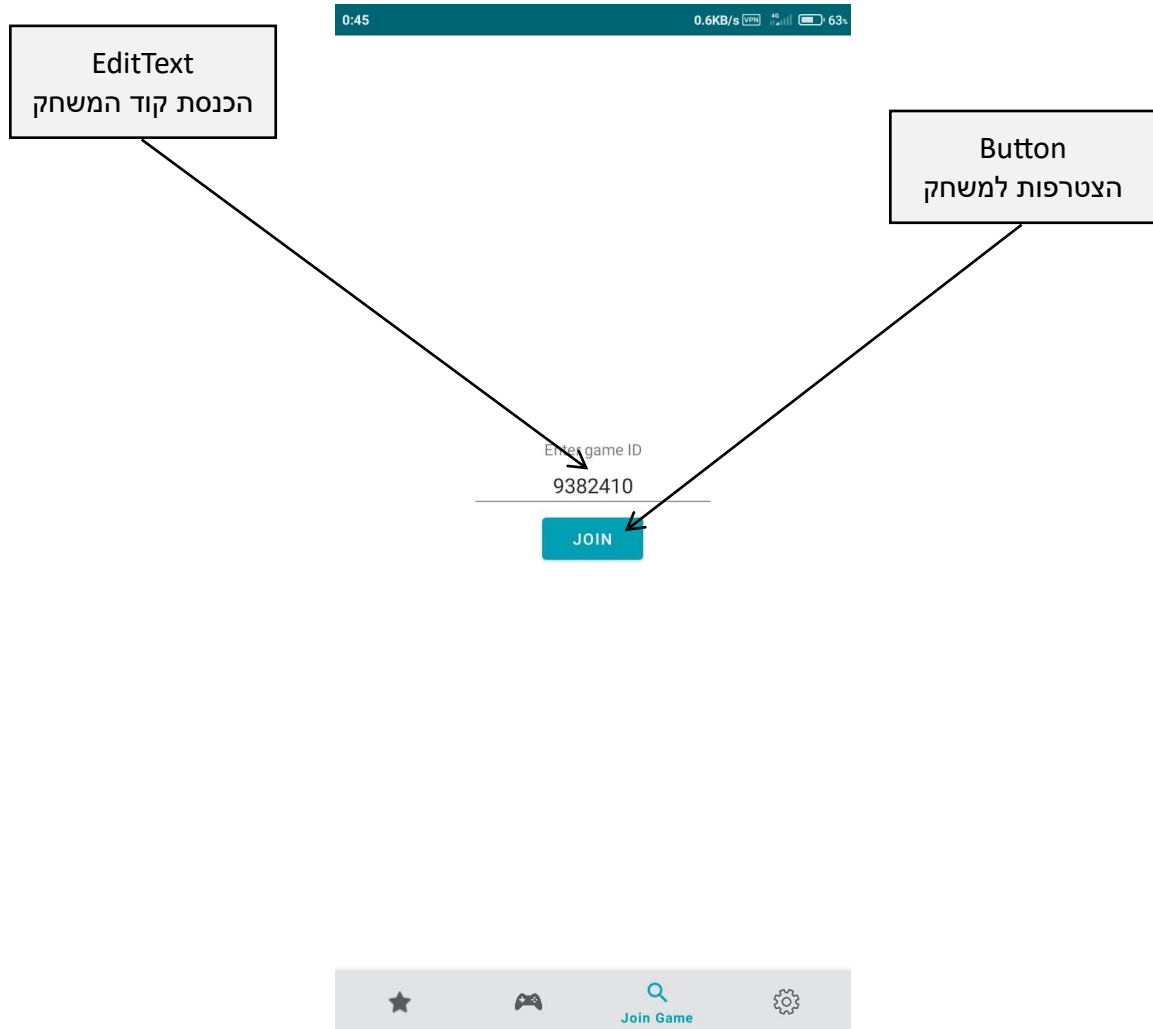
0:43 0.7KB/s 4G 63%

TextView
הצגת קוד המשחק

8065688

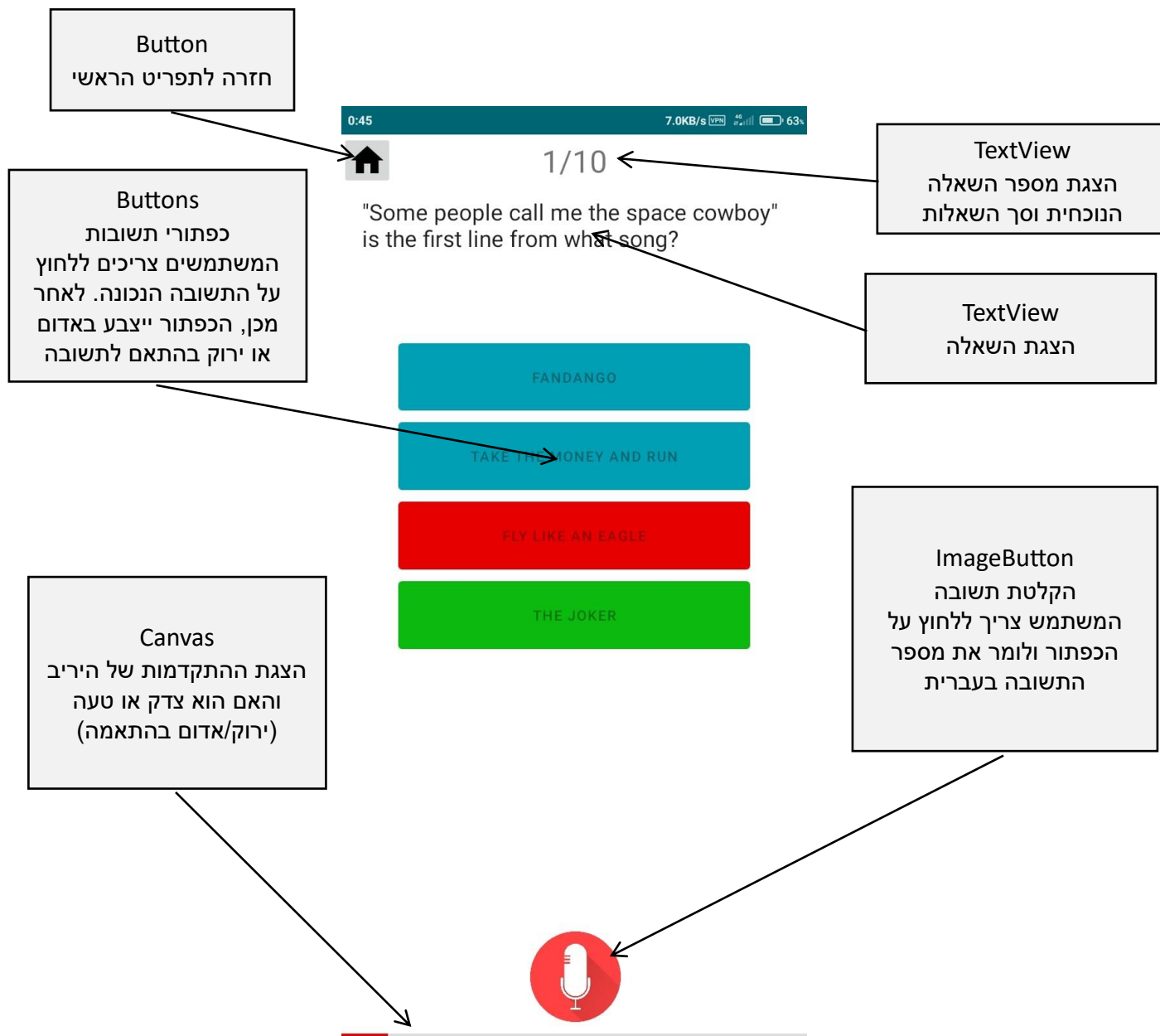
מסך הצטרפות למשחק

אחרי שהשחקן הראשון יצר את המשחק, השחקן השני יכניס במסך זה את קוד המשחק כדי להצטרך אליו.



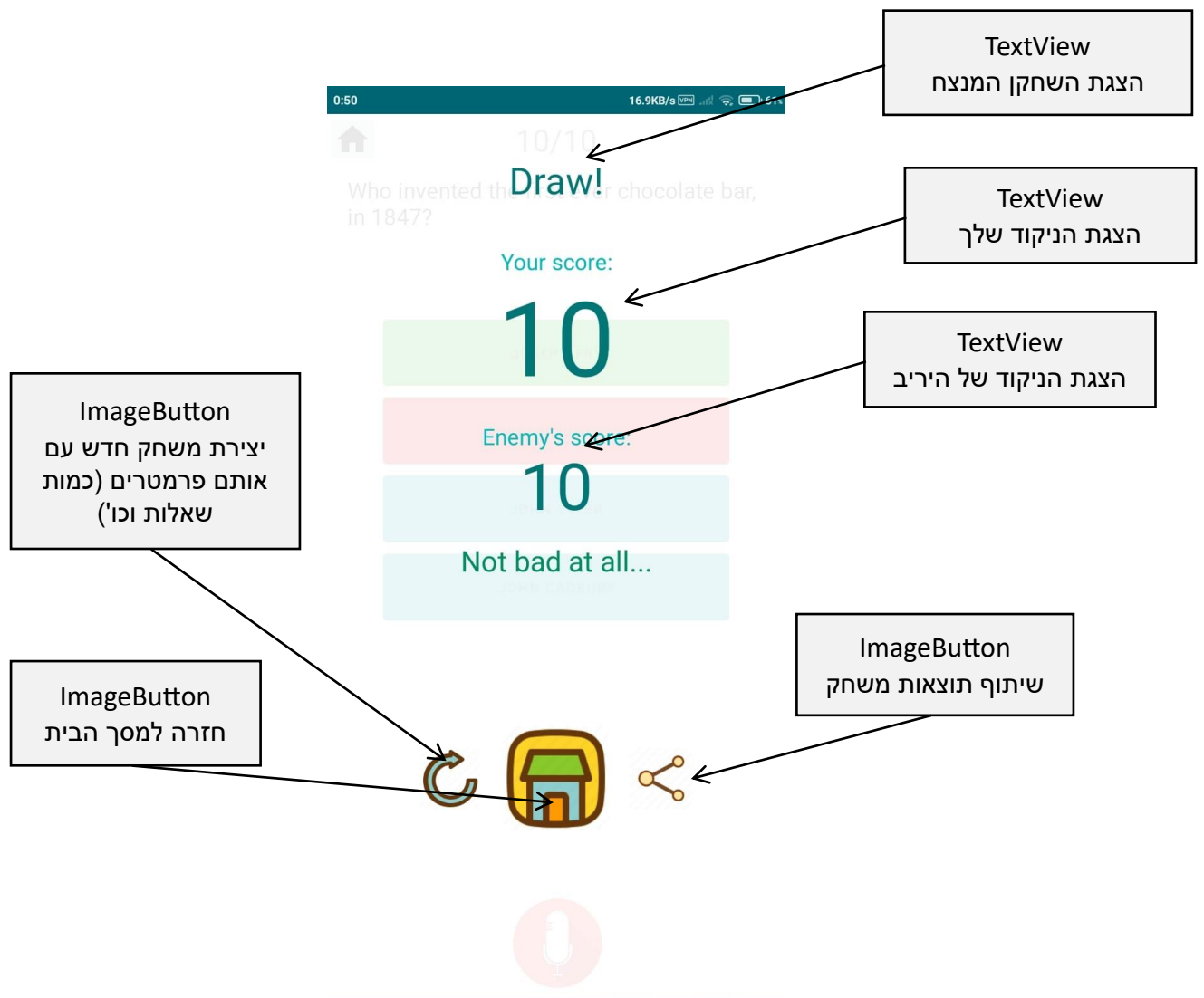
מסך המשחק

כששני השחקנים מחוברים, הם מועברים למסך זה, בו המשחק עצמו מתנהל.



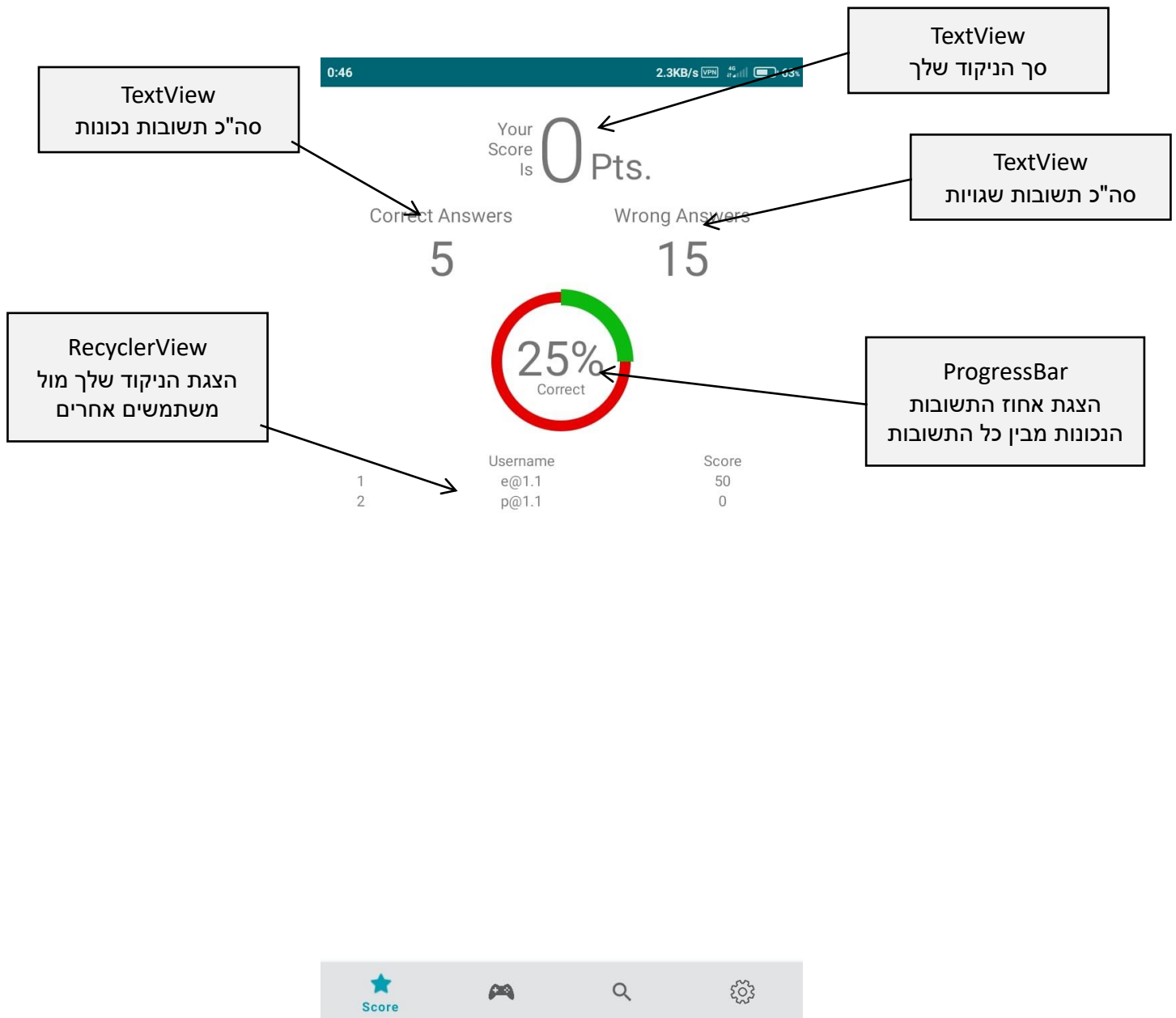
מסך סיום משחק

בסוף המשחק, יופיע מסך זה, המציג את הניקוד ואת המנצח במשחק.



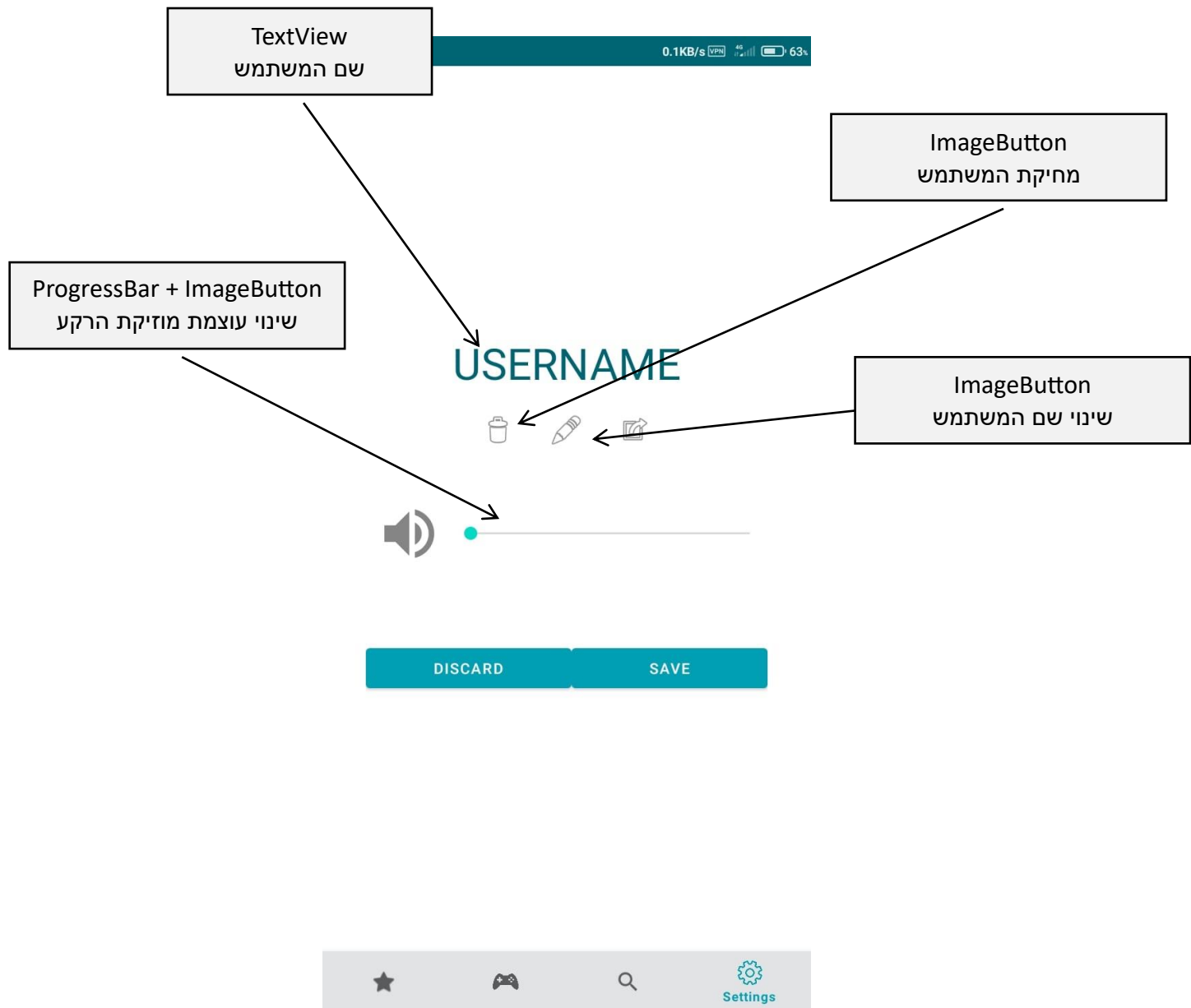
מסך ניקוד

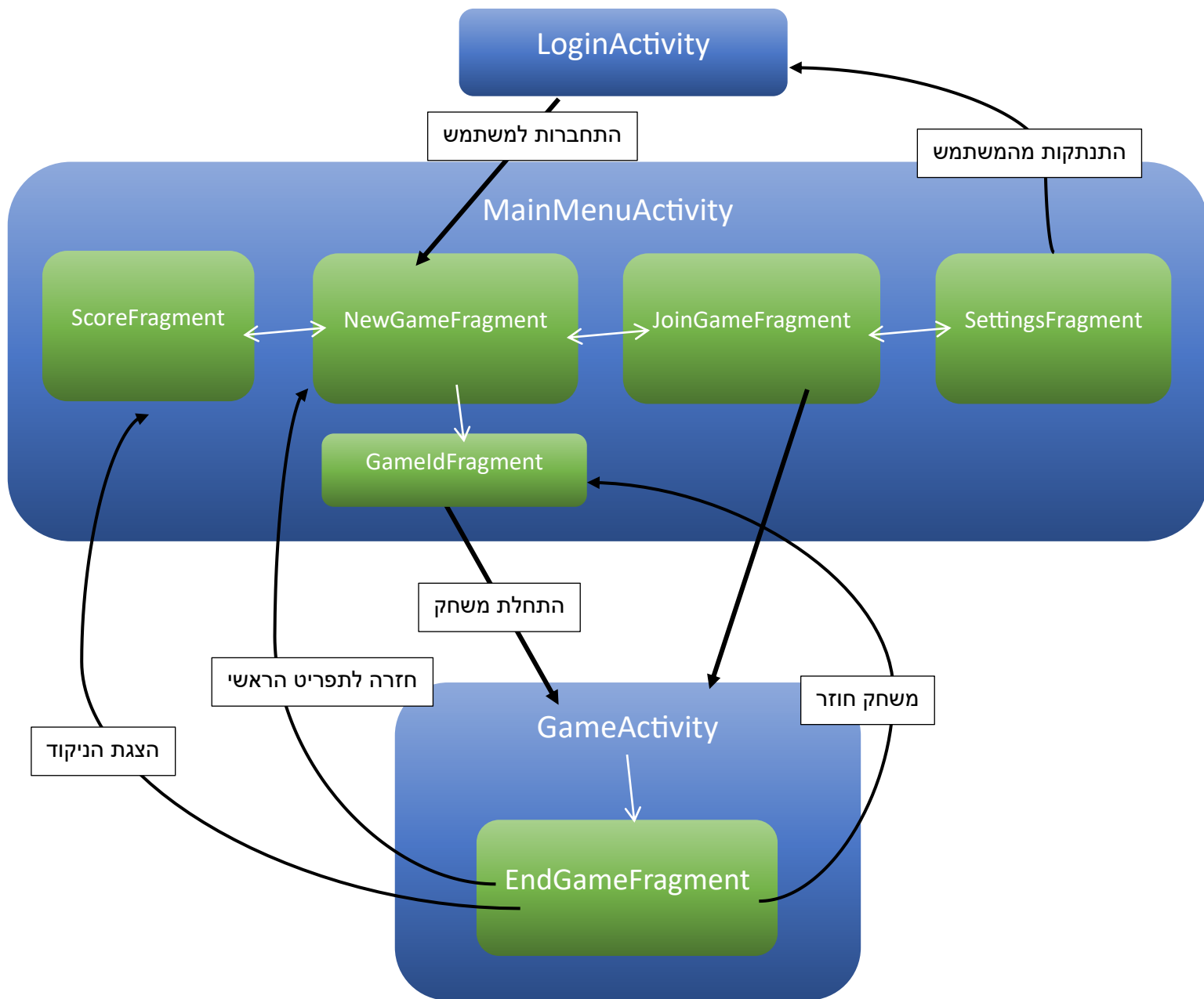
במסך זה המשתמש יכול לראות את מצב הנקודות שלו ושל אחרים, וסטטיסטיקות נוספות.

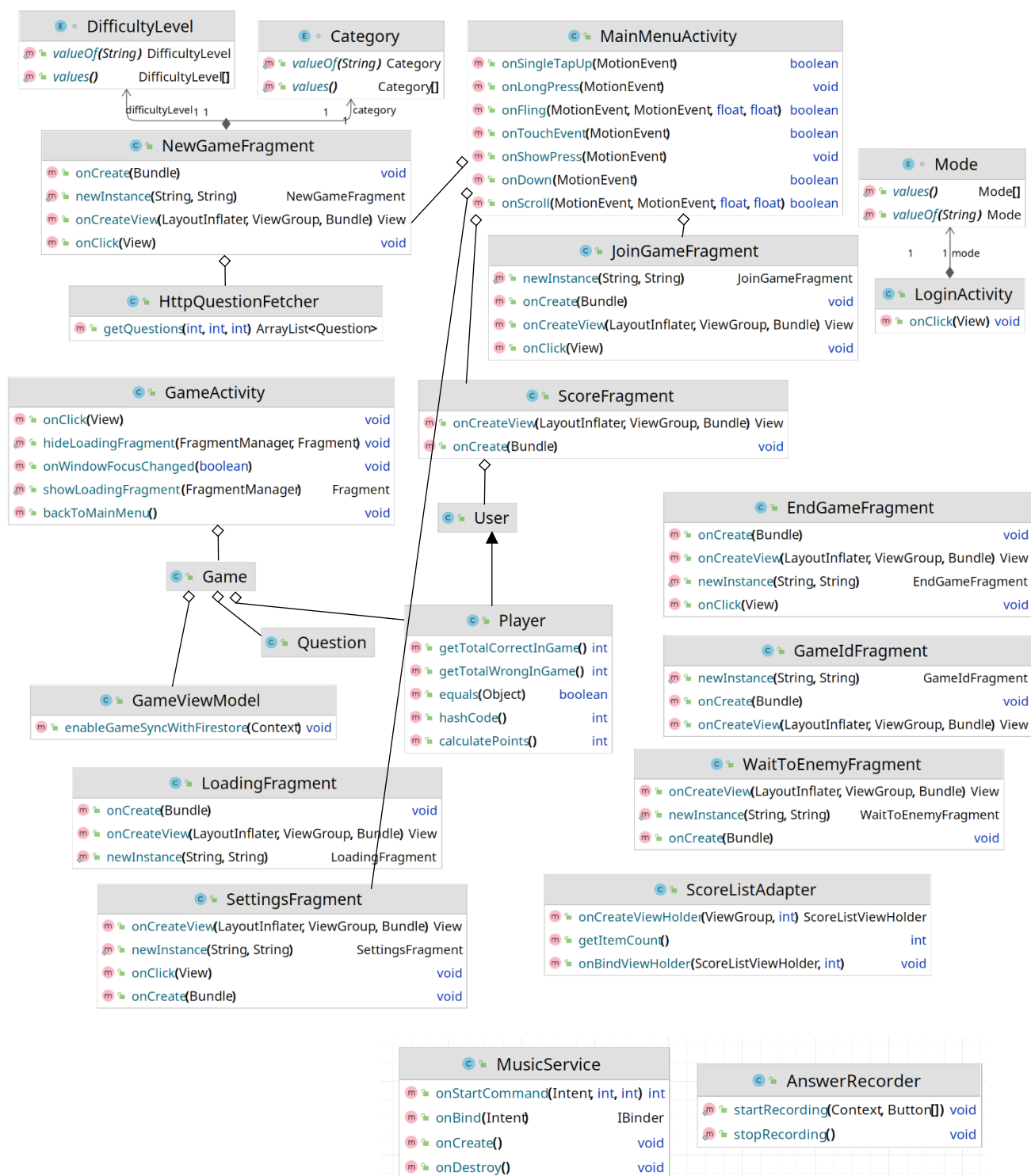


מסך הגדרות

במסך זה אפשר לערוך את הגדרות האפליקציה והגדרות המשתמש.
הערה: רוב הפיצ'רים במסך זה עדיין אינם ממומשים, אך ימומשו עד לבחינה עצמה.







מימוש הפרויקט

מחלקות מסכים (Fragment/Activity)

LoginActivity

הערה: כל התכונות מוגדרות כ-privates אלא אם נכתב אחרת.

הסבר	תכונה
קבוע: מספר הספרות המינימלי בסיסמא	static final int MIN_PASSWORD_LENGTH = 6
שם הקובץ ששומר את פרטי ההתחברות (שם המשתמש והסיסמא) ב-SharedPreferences	static final String LOGIN_PREFERENCES_FILE = "loginSp"
key של שם המשתמש ב-SharedPreferences	static final String LOGIN_PREFERENCES_USERNAME = "username"
key של הסיסמא ב-SharedPreferences	static final String LOGIN_PREFERENCES_PASSWORD = "password"
כפתור התחברות/יצירת משתמש	Button loginButton
כותרת המעבר ממצב התחברות למצב הרשמה	TextView toggleLoginModeLbl
לייבל לחיצה (Clickable) למעבר ממצב התחברות למצב הרשמה ובחזרה	TextView toggleLoginModeLink
שורה להכנסת שם המשתמש	EditText usernameTxt
שורה להכנסת הסיסמא	EditText passwordTxt
שורה להכנסת הסיסמא שוב (במצב הרשמה) כדי לוודא שאין שגיאת הקלדה	EditText passwordAgainTxt
קובע האם לזכור את שם המשתמש והסיסמא להתחברות הבאה ולהתחבר באופן אוטומטי	CheckBox rememberMeCb
מודיע אם ההתחברות/הרשמה הצליחה. אם לא כותב מה השגיאה	TextView loginStatusLbl
המצב הנוכחי – התחברות או הרשמה enum Mode { LOGIN, SIGNUP }	Mode mode

שמירת המידע בבסיס הנתונים:

המשתמשים נשמרים בפייירסטור בקולקציה (מעין <HashMap>) של אובייקטים מסוג User. המזהה של כל משתמש הוא שם המשתמש שלו.

פעולות עיקריות

(פעולות של שמירת המידע ל-sharedPreference יפורטו בחלק "בסיסי נתונים")
התחברות:

```
private void login(String username, String password) {
```

בדיקה אם שם המשתמש והסיסמא תקינים
(אורך תקין, אין תווים לא חוקיים וכו')

```
if(!validateUsernameAndPassword(username, password))  
    return;
```

אם הם תקינים - התחברות

פנייה לפייירבייס לבקשת התחברות. בסוף ההתחברות נקראת פעולת onComplete

```
FirebaseAuth.getInstance().signInWithEmailAndPassword(username,  
password).addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {  
    @Override
```

```

public void onComplete(@NonNull Task<AuthResult> task) {
    if(task.isSuccessful()){
        התחברות הצליחה
        מעבר לתפריט הראשי (מסך יצירת משחק)

        startMainMenuActivity();
        //TODO: if user isn't listed in "users" list, then create user and add it
        //can happen if user was signed in and had an connection error before he was
        added to list
    }
    Else

        התחברות נכשלה
        צריך להודיע למשתמש מה הייתה השגיאה

        loginStatusLbl.setText(task.getException().getMessage());
    }
}

{
    הרשמה:

private void signup(String username, String password, String passwordAgain) {
    בדיקה אם 2 הסיסמאות שהוכנסו זהות

    if(!password.equals(passwordAgain)){
        אם הן שונות, מודיעים למשתמש ויוצאים מהפעולה

        loginStatusLbl.setText("Passwords doesn't match");
        passwordTxt.setText("");
        passwordAgainTxt.setText("");
        setWrongColors(passwordTxt);
        setWrongColors(passwordAgainTxt);
        validateUsernameAnsPassword(username, password);
        return;
    }

    בודקים אם שם המשתמש והסיסמא תקינים
    אם לא, יוצאים מהפעולה

    if(!validateUsernameAnsPassword(username, password))
        return;

    String email = username;

    פנייה לפיירבייס לבקשת יצירת משתמש

    FirebaseAuth.getInstance().createUserWithEmailAndPassword(email,
    password).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
        @Override
        public void onComplete(@NonNull Task<AuthResult> task) {
            אחרי ביצוע הבקשה בודקים אם ההרשמה הצליחה

            if(task.isSuccessful()){
                //add user to users list
            }
        }
    });
}

```

אם היא הצליחה, מוסיפים את המשתמש לרשימת המשתמשים

```
User user = new User(username, FirebaseAuth.getInstance().getUid(), 0, 0, 0);
FirebaseFirestore.getInstance().
collection(GameActivity.USERS_COLLECTION_PATH).document(email).set(user).
addOnCompleteListener(new OnCompleteListener<Void>() {
    @Override
    public void onComplete(@NonNull Task<Void> task) {
        if(task.isSuccessful())
            אחרי שמוסיפים את המשתמש לרשימת המשתמשים, עוברים לתפריט הראשי
            startMainMenuActivity();
        else
            אם הרשמת המשתמש/ההוספה שלו לרשימת המשתמשים נכשלה, מודיעים למשתמש מה הייתה
            השגיאה
            loginStatusLabel.setText(task.getException().getMessage());
    }
});
}
else
    loginStatusLabel.setText(task.getException().getMessage());
}
});
{
```

MainMenuActivity

הסבר	תכונה
רכיב שמזהה החלקות על המסך. אם המשתמש החליק ימינה או שמאלה לאורך מרחק מסוים, עוברים מסך ימינה/שמאלה בהתאמה	GestureDetectorCompat swipeDetector
המרחק המינימלי של החלקה אופקית בשביל לעבור מסך. אם המשתמש החליק על המסך למרחק קטן יותר, לא יקרה כלום	static final int MIN_SWIPE_LENGTH = 150
תפריט בתחתית המסך	BottomNavigationView navigationView
ארבעת הפרגמנטים אליהם ניתן לעבור בתפריט ביצירת המסך, הפרגמנט המוצג הוא NewGameFragment	NewGameFragment newGameFragment
	JoinGameFragment joinGameFragment
	ScoreFragment scoreFragment
	SettingsFragment settingFragment

פעולות

מעבר בין מסכים בהחלקה:

```
public boolean onFling(@NonNull MotionEvent e1, @NonNull MotionEvent e2, float
velocityX, float velocityY) {
```

מציאת הפרגמנט שמוצג עכשיו

```
Fragment currentFragment =
getSupportFragmentManager().findFragmentById(R.id.mainFragmentContainer);
```

```
FragmentTransaction fragmentTransaction =
getSupportFragmentManager().beginTransaction();
```

בדיקה אם ההחלפה מספיק גדולה בשביל לעבור מסך

```
if(Math.abs(e2.getX() - e1.getX()) > MIN_SWIPE_LENGTH){
    //swipe
```

בדיקה אם ההחלפה הייתה ימינה או שמאלה

```
if(e2.getX() > e1.getX()){
    //right swipe
```

בהחלפה ימינה: מעבר מסך אחד שמאלה

```
if(currentFragment instanceof JoinGameFragment){
    fragmentTransaction.replace(R.id.mainFragmentContainer,
newGameFragment).commit();
    navigationView.setSelectedItemId(R.id.newGameFragment);
}
else if(currentFragment instanceof NewGameFragment){
    fragmentTransaction.replace(R.id.mainFragmentContainer,
scoreFragment).commit();
    navigationView.setSelectedItemId(R.id.scoreFragment);
}
else if(currentFragment instanceof SettingsFragment){
    fragmentTransaction.replace(R.id.mainFragmentContainer,
joinGameFragment).commit();
    navigationView.setSelectedItemId(R.id.joinGameFragment);
}
}
else{
```

בהחלפה שמאלה: מעבר מסך אחד ימינה

```
//left swipe
if(currentFragment instanceof ScoreFragment){
    fragmentTransaction.replace(R.id.mainFragmentContainer,
newGameFragment).commit();
    navigationView.setSelectedItemId(R.id.newGameFragment);
}
else if(currentFragment instanceof NewGameFragment){
    fragmentTransaction.replace(R.id.mainFragmentContainer,
joinGameFragment).commit();
    navigationView.setSelectedItemId(R.id.joinGameFragment);
}
else if(currentFragment instanceof JoinGameFragment){
    fragmentTransaction.replace(R.id.mainFragmentContainer,
settingFragment).commit();
    navigationView.setSelectedItemId(R.id.settingsFragment);
```

```
}
}
}
```

החזרה: הפעולה הסתיימה בהצלחה

```
return true;
```

```
{
```

הערה: מבחינה תכנותית, נכון יותר לשים את הFragmentים במערך. בכתיבת הקוד, היו בהתחלה רק 2 מסכים, אח"כ 3 ובסוף 4, ולכן עשיתי בדרך הזו.

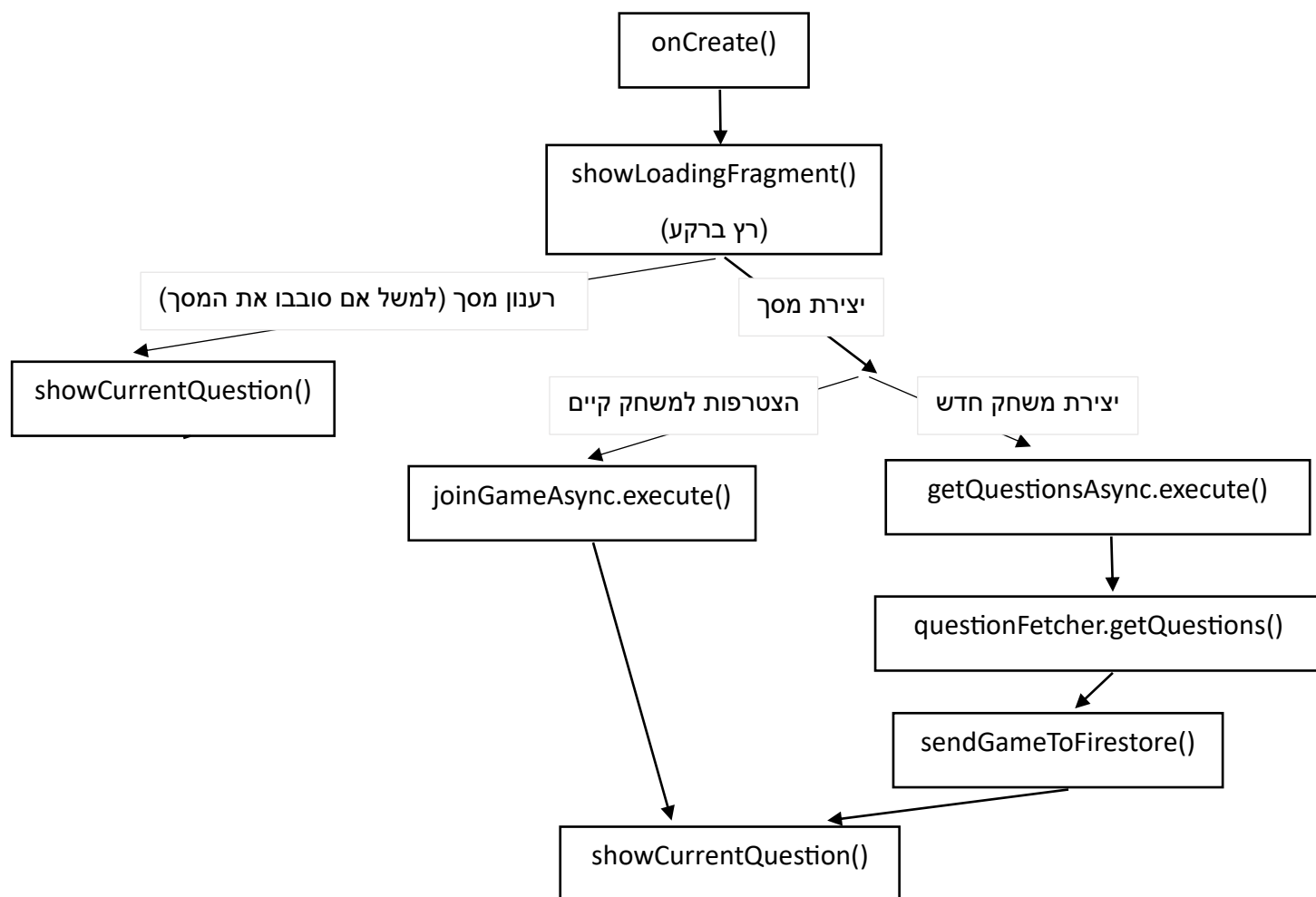
NewGameFragment

תכונה	הסבר
static final float UNSELECTED_BUTTON_ALPHA = (float) 0.4	אם בוחרים כפתור (למשל: כפתור של רמת קושי קלה), כל הכפתורים האחרים שאיתו (רמת קושי בינונית וקשה) נעשים שקופים קצת. זו רמת השקיפות שלהם.
static final int MIN_QUESTION_COUNT = 2	מספר השאלות המינימלי
static final int MAX_QUESTION_COUNT = 15	מספר השאלות המקסימלי
Button allCategoriesBtn	כפתור בחירת קטגוריה: כל הקטגוריות
Button generalKnowledgeCategoryBtn	כפתור בחירת קטגוריה: ידע כללי
Button scienceCategoryBtn	כפתור בחירת קטגוריה: מדע
Button computerScienceCategoryBtn	כפתור בחירת קטגוריה: מדעי המחשב
Category category	הקטגוריה שנבחרה
	enum Category{ ALL, GENERAL_KNOWLEDGE, SCIENCE, COMPUTER_SCIENCE }
Button easyDifficultyLevelBtn	כפתור בחירת רמת קושי: קל
Button mediumDifficultyLevelBtn	כפתור בחירת רמת קושי: בינוני
Button hardDifficultyLevelBtn	כפתור בחירת רמת קושי: קשה
DifficultyLevel difficultyLevel	רמת הקושי שנבחרה
	enum DifficultyLevel{ EASY, MEDIUM, HARD }
Button questionCountDecBtn	הקטנת כמות השאלות למשחק ב-1
Button questionCountIncBtn	הגדלת כמות השאלות למשחק ב-1
TextView questionCountValueLbl	הצגת מספר השאלות שנבחר למשתמש
int questionCount	מספר השאלות שנבחר
Button playBtn	כפתור התחלת המשחק

שמירת המידע בבסיס הנתונים:

המשחקים נשמרים בפירסטור בקולקציה (מעין <HashMap>) של אובייקטים מסוג Game. המזהה של כל משחק הוא שדה ID שלו.

תרשים זרימה



פעולות

הפעולה שנקראת כשנלחץ כפתור במסך:

```
public void onClick(View v) {
```

```
    switch (v.getId()){
```

אם זה אחד מכפתורי בחירת רמת הקושי, בוחרים את רמת הקושי הזו. קוראת לפעולה setDifficultyLevel, שבהמשך אפרט מה היא עושה.

```
    case R.id.easyDifficultyLevelBtn:
```

```
        setDifficultyLevel(EASY);
```

```
        break;
```

בדיקה איזה כפתור נלחץ

```
case R.id.mediumDifficultyLevelBtn:
    setDifficultyLevel(MEDIUM);
    break;
case R.id.hardDifficultyLevelBtn:
    setDifficultyLevel(HARD);
    break;
```

אם זה אחד מכפתורי בחירת הקטגוריה: מסמנים את הקטגוריה הזו.

```
case R.id.allCategoriesBtn:
    setCategory(ALL);
    break;
case R.id.generalKnowledgeCategoryBtn:
    setCategory(GENERAL_KNOWLEDGE);
    break;
case R.id.scienceCategoryBtn:
    setCategory(SCIENCE);
    break;
case R.id.computerScienceCategoryBtn:
    setCategory(COMPUTER_SCIENCE);
    break;
```

אם נלחץ כפתור הוספת/הורדת שאלה, מוסיפים או מורידים שאלה בהתאמה.

```
case R.id.questionCountDecBtn:
    questionCountDec();
    break;
case R.id.questionCountIncBtn:
    questionCountInc();
    break;
```

אם נלחץ כפתור התחלת משחק, עוברים למסך יצירת משחק

```
case R.id.playBtn:
    משתמשים בIntent כדי להעביר למסך את רמת הקושי, מספר השאלות והקטגוריה
    Intent intent = new Intent(getActivity(), GameActivity.class);
    int extras[] = new int[3];
    extras[QUESTIONS_COUNT_INDEX] = questionCount;
    extras[DIFFICULTY_LEVEL_INDEX] = difficultyLevel.ordinal();
    extras[CATEGORY_INDEX] = category.ordinal();
    intent.putExtra(GameActivity.NEW_GAME_EXTRAS, extras);

    intent.putExtra(IS_NEW_GAME_EXTRA, true);

    מתחילים את GameActivity

    startActivity(intent);
    getActivity().finish();
}
```

```

}

פעולה שקובעת את רמת הקושי לרמת הקושי שנבחרה:
void setDifficultyLevel(DifficultyLevel newDifficultyLevel){
    מאפסים את אחוז השקיפות של כל כפתורי קביעת רמת הקושי
    easyDifficultyLevelBtn.setAlpha(UNSELECTED_BUTTON_ALPHA);
    mediumDifficultyLevelBtn.setAlpha(UNSELECTED_BUTTON_ALPHA);
    hardDifficultyLevelBtn.setAlpha(UNSELECTED_BUTTON_ALPHA);
    בודקים מה רמת הקושי שנבחרה מסמנים את הכפתור שלה
    וקובעים את רמת הקושי להיות זו שנבחרה.

    difficultyLevel = newDifficultyLevel;

    switch (newDifficultyLevel){
        case EASY:
            easyDifficultyLevelBtn.setAlpha(1);
            break;
        case MEDIUM:
            mediumDifficultyLevelBtn.setAlpha(1);
            break;
        case HARD:
            hardDifficultyLevelBtn.setAlpha(1);
            break;
    }
}
}

```

JoinGameFragment

תכונה	הסבר
public static final int GAME_ID_LENGTH = 7	אורך הID של משחק מוגדר כpublic כי גם ביצירת משחק צריך לדעת את האורך של הID
EditText gameIdTxt	שורה בה כותבים את הID
Button joinGameBtn	הצטרפות למשחק

פעולות

הצטרפות למשחק

```

private void joinGame(){
    String id = gameIdTxt.getText().toString();
    int intId;

    try{

        הפיכת הID לint.

        intId = Integer.parseInt(id);

        אם אי אפשר להפוך לint, הID לא חוקי (ID יכול לכלול רק מספרים).
    }
}

```



```

if(id.length() != GAME_ID_LENGTH)
    אם אי אפשר להפוך לint, הID לא חוקי (ID יכול לכלול רק מספרים).
    throw new Exception();
}

ID לא חוקי. מודיעים למשתמש שהID לא חוקי.

catch (Exception e){
    Toast.makeText(getContext(), "Invalid ID entered!", Toast.LENGTH_SHORT).show();
    gameIdTxt.setText("");
    return;
}

אם הID חוקי, עוברים למסך המשחק.
intent, מודיעים שהמשתמש הצטרף למשחק (ולא יצר אותו),
ומעבירים את הID של המשחק.

Intent intent = new Intent(getActivity(), GameActivity.class);
intent.putExtra(GameActivity.IS_NEW_GAME_EXTRA, false);
intent.putExtra(GameActivity.GAME_ID_EXTRA, intId);

startActivity(intent);
}
}

```

SettingsFragment

הסבר	תכונה
עוצמת הקול של מוזיקת הרקע מוגדר כpublic כדי שיהיה אפשר לשנות ולהאזין מMusicService	public static MutableLiveData<Float> backgroundMusicVolume
כפתור מחיקת משתמש	ImageButton deleteUserBtn
כפתור עריכת שם משתמש	ImageButton editUsernameBtn
כפתור התנתקות ממשתמש	ImageButton logoutBtn
כפתור השתקת עוצמת הקול	ImageButton muteBtn
קביעת עוצמת הקול	SeekBar volumeSb

פעולות

אתחול של SeekBar שאיתו קובעים את הווליום:

```
private void initVolumeSb(View v) {
```

מציאת הרכיב בXML

```

volumeSb = v.findViewById(R.id.volumeSb);
//multiplied by 100 because volume is between 0-1 and progress is between 0-100
קביעת הערך ההתחלתי (100 – ווליום הכי גבוה).
volumeSb.setProgress(backgroundMusicVolume.getValue().intValue() * 100);
כשמשנים את הערך בSeekBar, משנים את עוצמת הקול.
שינוי של עוצמת הקול קורא לObserver בMusicService שמשנה את העוצמה.
boolean.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
    @Override

```

```

public void onProgressChanged(SeekBar seekBar, int progress, boolean fromUser) {
    backgroundMusicVolume.setValue((float)progress / 100);
}

@Override
public void onStartTrackingTouch(SeekBar seekBar) {

}

@Override
public void onStopTrackingTouch(SeekBar seekBar) {

}
});

```

ScoreFragment

הסבר	תכונה
רשימת המשתמשים	RecyclerView scoreRv
סרגל התקדמות מעגלי, שמציג כמה אחוז מהתשובות נכונות וכמה שגויות	CircularProgressBar successPercentagePb
לייבל שמציג את הנקודות של השחקן	TextView totalScoreLbl
לייבל שמציג את סה"כ התשובות הנכונות	TextView totalCorrectLbl
לייבל שמציג את סה"כ התשובות השגויות	TextView totalWrongLbl
לייבל שמציג את אחוז התשובות הנכונות	TextView successPercentageLbl
רשימת משתמשים	ArrayList<User> users
המשתמש הנוכחי	User currentUser

פעולות

יצירת הפרגמנט וקבלת נתוני המשתמש מFirestore

```

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

    קביעת השדות של המחלקה לערך שלהם בקובץ layout

    // Inflate the layout for this fragment
    View v = inflater.inflate(R.layout.fragment_score, container, false);
    scoreRv = v.findViewById(R.id.scoreRv);
    successPercentagePb = v.findViewById(R.id.successPercentagePb);
    totalScoreLbl = v.findViewById(R.id.totalScoreLbl);
    totalCorrectLbl = v.findViewById(R.id.totalCorrectLbl);
    totalWrongLbl = v.findViewById(R.id.totalWrongLbl);
    successPercentageLbl = v.findViewById(R.id.successPercentageLbl);

    הצגת פרגמנט טעינה

    Fragment loadingFragment = new LoadingFragment();

```

```
getChildFragmentManager().beginTransaction().replace(R.id.scoreLayout,
loadingFragment).commit();
```

הורדת רשימת המשתמשים מפיירסטור

```
//fetch user list
FirebaseFirestore.getInstance().
collection(GameActivity.USERS_COLLECTION_PATH).get().addOnFailureListener(new
OnFailureListener() {
    @Override
```

**במקרה שהורדת הרשימה נכשלה – הצגת הודעת שגיאה למשתמש
וחזרה למסך יצירת משחק**

```
public void onFailure(@NonNull Exception e) {
    Toast.makeText(getContext(), "Connection error!",
Toast.LENGTH_SHORT).show();
    //back to main manu
    FragmentTransaction ft = getChildFragmentManager().beginTransaction();
    ft.replace(R.id.mainFragmentContainer, new NewGameFragment()).commit();
}
}).addOnSuccessListener(new OnSuccessListener<QuerySnapshot>() {
    @Override
    public void onSuccess(QuerySnapshot queryDocumentSnapshots) {
        במקרה שההורדה הצליחה – אתחול רשימת המשתמשים לרשימה שהתקבלה.
        ArrayList<User> fetchedUserList = new ArrayList<>();
        for(DocumentSnapshot documentSnapshot :
queryDocumentSnapshots.getDocuments())
            if(documentSnapshot.exists())
                fetchedUserList.add(documentSnapshot.toObject(User.class));

        users = fetchedUserList;
```

מציאת המשתמש הנוכחי לצורך הצגת הסטטיסטיקות שלו

```
//find current user
for(User user : users)
    if(user.getId().equals(FirebaseAuth.getInstance().getUid()))
        currentUser = user;
if(currentUser == null)
    אם המשתמש הנוכחי לא קיים (לא צריך לקרות) – מציגים הודעת שגיאה.
    Toast.makeText(getContext(), "Current user not found!",
Toast.LENGTH_SHORT).show();
```

הסתרת מסך הטעינה

```
getChildFragmentManager().beginTransaction().hide(loadingFragment).commit();
קריאה לפעולה שמציגה את הסטטיסטיקות על המסך (אפרט עליה בהמשך)
showScore();
}
});
```

```
return v;
}
```

הצגת הסטטיסטיקות על המסך:

```
private void showScore() {
```

הצגת הניקוד, מספר התשובות הנכונות והשגויות של המשתמש הנוכחי

```
totalScoreLbl.setText(Integer.toString(currentUser.getScore()));
totalCorrectLbl.setText(Integer.toString(currentUser.getTotalCorrect()));
totalWrongLbl.setText(Integer.toString(currentUser.getTotalWrong()));
```

הצגת אחוז התשובות הנכונות

```
//show correct percentage progress bar
successPercentagePb.setProgressBarColor(MyColor.CORRECT_GREEN);
successPercentagePb.setProgressBarWidth(15);
successPercentagePb.setBackgroundProgressBarColor(MyColor.WRONG_RED);
successPercentagePb.setBackgroundProgressBarWidth(10);
```

```
int totalAnswers = currentUser.getTotalCorrect() + currentUser.getTotalWrong();
int progress;
if(totalAnswers > 0)
    progress = (100 * currentUser.getTotalCorrect()) / totalAnswers;
else
    //avoid division by zero
    progress = 0;
```

```
successPercentagePb.setProgressWithAnimation(progress, (long)1000);
successPercentageLbl.setText(Integer.toString(progress) + "%");
```

מיון רשימת המשתמשים לפי הניקוד שלהם

```
//show user list
users.sort((o1, o2) -> (int)(o2.getScore() - o1.getScore()));
```

הצגת רשימת המשתמשים והניקוד שלהם

```
ScoreListAdapter scoreListAdapter = new ScoreListAdapter(getContext(), users);
scoreRv.setAdapter(scoreListAdapter);
scoreRv.setLayoutManager(new LinearLayoutManager(getContext()));
```

```
}
```

GameldFragment

תכונה	הסבר
TextView startGameGameldLbl	מציג למשתמש את הID של המשחק, כדי שהשחקן השני ידע להתחבר אליו

פעולות

אתחול המסך:

```
public View onCreateView(LayoutInflater inflater, ViewGroup container,
    Bundle savedInstanceState) {
    // Inflate the layout for this fragment
```

```

View view = inflater.inflate(R.layout.fragment_game_id, container, false);
// מציאת הלייבל שמציגה את הID
startGameGameIdLbl = view.findViewById(R.id.startGameGameIdLbl);
// מציאת הID של המשחק

GameViewModel gameViewModel = new
ViewModelProvider(getActivity()).get(GameViewModel.class);
int gameId = gameViewModel.getGame().getId();
// הצגת הID למשתמש

startGameGameIdLbl.setText(Integer.toString(gameId));

return view;
}

```

GameActivity

הסבר	תכונה
השם של אוסף המשחקים בפייירסטור	public static final String GAMES_COLLECTION_PATH = "games"
השם של אוסף המשתמשים בפייירסטור	public static final String USERS_COLLECTION_PATH = "users"
שם הextra intent שכולל פרמטרים של המשחק החדש – קטגוריה, רמת קושי ומספר שאלות, כמערך של int.	public static final String NEW_GAME_EXTRAS = "extras"
שם הextra intent שאומר למחלקה האם צריך ליצור משחק חדש (כלומר – האם המשתמש הזה יוצר את המשחק או מתחבר למשחק קיים).	public static final String IS_NEW_GAME_EXTRA = "isNewGame"
שם הextra intent שכולל את הID של המשחק (בהתחברות למשחק קיים)	public static final String GAME_ID_EXTRA = "gameId"
בAsyncTask, הפרמטרים מתקבלים במערך של Integer'ים. אלו האינדקסים של פרמטרים שונים בGetQuestionsAsync. בהמשך אראה את אופן השימוש בהם.	public static final int QUESTIONS_COUNT_INDEX = 0
	public static final int DIFFICULTY_LEVEL_INDEX = 1
	public static final int CATEGORY_INDEX = 2
לייבל שמציגה את מספר השאלה הנוכחית מתוך כל השאלות	TextView currentQuestionLbl
לייבל שמציגה את גוף השאלה	TextView questionLbl
4 כפתורים שעליהם לוחצים כדי לבחור בתשובה	Button[] answerButtons
כפתור חזרה למסך הבית	ImageButton homeImgBtn
קנבס שמציג את ההתקדמות של היריב, ואת התשובות הנכונות והשגויות שלו	Canvas pbCanvas
מקום בו נמצא הקנבס הנ"ל	ImageView progressImg
ביטמאפ שנצרך בשביל הקנבס (בהמשך אראה למה צריך את שלושתם)	Bitmap progressBitmap
כפתור הקלטת תשובה	ImageButton recordImgBtn
רכיב ששומר נתונים של המשחק	GameViewModel gameVM
פרגמנט שמראה את טעינת המסך	Fragment loadingFragment

פרגמנט שמציג את הID של המשחק לפני ההתחלה, כדי שהמשתמש השני יוכל להתחבר	Fragment gameIdFragment
האובייקט המרכזי של פיירסטור שמאפשר חיבור לDB.	FirebaseFirestore firestore

פעולות

הורדת השאלות מהשרת, ביצירת משחק (משתמש בAsyncTask, מחלקה שמאפשרת להריץ פעולות ברקע תוך כדי ריצת הת'רד הראשי):

```
private class GetQuestionsAsync extends AsyncTask<Integer, Integer, ArrayList<Question>> {
    @Override
    //params: question count, difficulty level, category
    protected ArrayList<Question> doInBackground(Integer... integers) {
        יצירת אובייקט של המחלקה שמורידה את השאלות וקבלת השאלות ממנו.
        פרמטרים כמו קטגוריה וכו' מתקבלים בפרמטר integers.

        HttpQuestionFetcher questionFetcher = new HttpQuestionFetcher();
        return questionFetcher.getQuestions(integers[QUESTIONS_COUNT_INDEX],
        integers[DIFFICULTY_LEVEL_INDEX], integers[CATEGORY_INDEX]);
    }
}
```

פעולה שרצה לאחר קבלת השאלות מהשרת

```
@Override
protected void onPostExecute(ArrayList<Question> questions) {
    בדיקה אם התקבלו שאלות. אם לא, הצגת הודעת שגיאה וחזרה למסך הראשי.
    if (questions == null || questions.size() == 0) {
        Toast.makeText(getBaseContext(), "Couldn't fetch questions!",
        Toast.LENGTH_SHORT).show();
        backToMainMenu();
    } else {
        אם התקבלו שאלות בהצלחה, שומרים אותן בViewModel.
        gameVM.setQuestions(questions);
        שליחת המשחק שנוצר לפיירסטור. פעולה זו גם מציגה את gameIdFragment עד שיתחבר המשתמש.
        sendGameToFirestore();
    }
}
```

קבלת Game מהשרת, במצב התחברות למשחק:

```
private class JoinGameAsync extends AsyncTask<Integer, Integer, Void>{
    @Override
    //params: id

    הפעולה מקבלת כפרמטר את ID של המשחק.
    protected Void doInBackground(Integer... integers) {
        שמירת השחקן הנוכחי, כי הורדת המשחק מפיירסטור תמחק את המשתמש הנוכחי מGame, כי הוא עדיין לא נמצא בGame שבDB.

        Player myPlayer = gameVM.getMyPlayer(); //getting the game from firestore
        overrides player2 to null
    }
}
```

הפיכת ID לסטרינג

```
int id = integers[0];
String strId = Integer.toString(id);



### הורדה של המשחק מפיירסטור


firebaseFirestore.collection(GAMES_COLLECTION_PATH).document(strId).get().addOnSuccessListener {
    r(new OnSuccessListener<DocumentSnapshot>() {

        @Override
        public void onSuccess(DocumentSnapshot documentSnapshot) {
            //game loaded successfully
            if(documentSnapshot.exists()){

                אם ההורדה הצליחה, שמירת המשחק בviewModel:

                Game game = documentSnapshot.toObject(Game.class);

                gameVM.setGame(game);
                gameVM.enableGameSyncWithFirestore(GameActivity.this);
                gameVM.setMyPlayer(myPlayer);

                hideLoadingFragment(getSupportFragmentManager(), loadingFragment);

                הצגת השאלה הראשונה.

                showCurrentQuestion();
            }
            else{

                אם המשחק לא קיים, הצגת הודעת שגיאה וחזרה לתפריט הראשי

                Toast.makeText(GameActivity.this, "Wrong game ID!",
                    Toast.LENGTH_SHORT).show();
                backToMainMenu();
            }
        }
    });

    אם ההורדה נכשלה, הצגת הודעת שגיאה וחזרה לתפריט הראשי

    }).addOnFailureListener(new OnFailureListener() {
        @Override
        public void onFailure(@NonNull Exception e) {
            Toast.makeText(GameActivity.this, "Connection error!",
                Toast.LENGTH_SHORT).show();
            backToMainMenu();
        }
    });

    return null;
}
}
```

בכל פעם שהמשתמש בוחר תשובה, מציגים אם היא נכונה למשך שניה, ואחר כך מציגים את השאלה הבאה:

```
private void sendAnswer(int answerIndex, Button answerButton) {
    בדיקה אם התשובה שנענתה נכונה
    int currentQuestionIndex = gameVM.getPlayer().getCurrentQuestionIndex();
    boolean isCorrect = gameVM.getQuestions().get(currentQuestionIndex).correctAnswer
    == answerIndex;

    הוספה לרשימת התשובות הנכונות/שגויות של השחקן את התשובה שהוא ענה.
    ArrayList<Boolean> isCorrectList = gameVM.getPlayer().getIsCorrectList();
    isCorrectList.add(isCorrect);
    gameVM.setMyIsCorrectList(isCorrectList);

    if (isCorrect) {
        אם התשובה נכונה, מציגים אותה בירוק
        answerButton.setBackgroundColor(MyColor.CORRECT_GREEN);
    } else {
        אם לא, מציגים אותה באדום.
        answerButton.setBackgroundColor(MyColor.WRONG_RED);
    }

    answerButtons[gameVM.getQuestions().get(currentQuestionIndex).correctAnswer].setBack
    groundColor(MyColor.CORRECT_GREEN);
}

מציירים האם התשובה נכונה/שגויה על הקנבס בתחתית המסך.
הפעולה DrawIsCorrectOnProgressBar תוסבר בפירוט בהמשך.
(כרגע מציג את ההתקדמות של השחקן הנוכחי, בהמשך אשנה כך שיציג את ההתקדמות של היריב).
drawIsCorrectOnProgressBar(isCorrect, currentQuestionIndex);

הגדלת אינדקס השאלה הנוכחית ב-1.
gameVM.setCurrentQuestionIndex(currentQuestionIndex + 1);

כיבוי של כפתורי התשובה בזמן הצגת התשובה הנכונה.
for (Button answerBtn : answerButtons)
    answerBtn.setEnabled(false);
recordImgBtn.setEnabled(false);

לחכות שניה אחת לפני הצגת השאלה הבאה
new Handler().postDelayed(new Runnable() {
    @Override
    public void run() {
        אם אין עוד שאלות (נגמר המשחק), מפעילים AsyncTask שיציג את מסך סיום המשחק
        if (currentQuestionIndex == gameVM.getQuestions().size() - 1) {
            //game ended
            EndGameAsync endGameAsync = new EndGameAsync();
            endGameAsync.execute();

            אם יש עוד שאלות, מציגים את השאלה הבאה.
        } else {
            showCurrentQuestion();
        }
    }
}
```



```
    }
    }, 1000);
}
```

ציור על הקנבס שמראה אם התשובה נכונה או שגויה:

```
private void drawIsCorrectOnProgressBar(boolean isCorrect, int currentQuestion) {
    מציאת מספר השאלות במשחק
```

```
    int totalQuestions = gameVM.getQuestions().size();
```

יצירת צייר ירוק/אדום בהתאם לתשובה נכונה/שגויה

```
    Paint paint = new Paint();
```

```
    if (isCorrect)
```

```
        paint.setColor(MyColor.CORRECT_GREEN);
```

```
    else
```

```
        paint.setColor(MyColor.WRONG_RED);
```

חישוב המקום בו צריך לצייר (בציר האופקי)

```
    int start = currentQuestion * (pbCanvas.getWidth() / totalQuestions);
```

```
    int end = (currentQuestion + 1) * (pbCanvas.getWidth() / totalQuestions);
```

ציור של מלבן על הקנבס

```
    pbCanvas.drawRect(start, 0, end, pbCanvas.getHeight(), paint);
```

הכנסה של הקנבס החדש לוס.

```
    progressImg.setImageBitmap(progressBitmap);
```

```
}
```

סיום המשחק:

```
private void endGame() {
```

```
    //TODO: delete game from firestore
```

הוספה של התשובות הנכונות והשגויות לסטטיסטיקות של השחקן

```
    //update user score
```

```
    gameVM.getMyPlayer().setTotalCorrect(gameVM.getMyPlayer().getTotalCorrect() +
    gameVM.getMyPlayer().getTotalCorrectInGame());
```

```
    gameVM.getMyPlayer().setTotalWrong(gameVM.getMyPlayer().getTotalWrong() +
    gameVM.getMyPlayer().getTotalWrongInGame());
```

חישוב הנקודות של 2 המשתמשים

```
    int myPoints = gameVM.getMyPlayer().calculatePoints();
```

```
    int otherPoints = gameVM.getOtherPlayer().calculatePoints();
```

```
    if(myPoints > otherPoints)
```

הוספת הניקוד של המנצח לסך הנקודות שלו (רק המנצח מקבל נקודות)

```
    //you won, add points to total score
```

```
    gameVM.getMyPlayer().setScore(gameVM.getMyPlayer().getScore() + myPoints);
```

המרה של Player ל User כדי להכניס אותו לרשימת השחקנים בפייירסטור

```
    User myPlayerAsUser = gameVM.getMyPlayer();
```

```
    //send new user data to users list
```

הכנסה של המשתמש לרשימה

```

    firestore.collection(USERS_COLLECTION_PATH).document(gameVM.getPlayer().getEmail(
    ))
    .set(myPlayerAsUser).addOnCompleteListener(new OnCompleteListener<Void>() {
        @Override
        public void onComplete(@NonNull Task<Void> task) {
            if(!task.isSuccessful()){
                אם ההכנסה נכשלה, מציגים הודעת שגיאה וחוזרים לתפריט הראשי
                Toast.makeText(GameActivity.this, "Connection error!",
                Toast.LENGTH_SHORT).show();
                backToMainMenu();
            }
            else{
                אם ההכנסה הצליחה, עוברים לפרגמנט של סיום המשחק שמציג את התוצאה לשחקנים
                //game updated successfully
                showEndGameFragment();
            }
        }
    });
}

```

EndGameFragment

הסבר	תכונה
ה ViewModel של המשחק – מאפשר לגשת למידע של המשחק מהפרגמנט	GameViewModel gameVM
מציג את הניקוד שלך	TextView yourScoreCountLbl
מציג את הניקוד של היריב	TextView enemyScoreCountLbl
מציג את המשתמש שניצח	TextView winnerLbl
כפתור למשחק חוזר	ImageButton endGameReplayBtn
כפתור לחזרה למסך הבית	ImageButton endGameHomeBtn
כפתור לשיתוף תוצאות המשחק	ImageButton endGameShareBtn

פעולות

הצגת התוצאות:

```
private void showResults(View view) {
```

חישוב הנקודות של השחקנים

```

    int yourScore = gameVM.getPlayer().calculatePoints();
    int enemyScore = gameVM.getOtherPlayer().calculatePoints();

    yourScoreCountLbl.setText(Integer.toString(yourScore));
    enemyScoreCountLbl.setText(Integer.toString(enemyScore));

```

הצגת המשתמש המנצח

```

if(yourScore > enemyScore) {
    //you won
    winnerLbl.setText("You won!");
    הלייבל שמציג את הניקוד של המנצח יהיה גדול יותר
    yourScoreCountLbl.setTextSize(TypedValue.COMPLEX_UNIT_SP, 84);
}
else if(yourScore < enemyScore){
    //you lost
    winnerLbl.setText(gameVM.getOtherPlayer().getUsername() + " won!");
    enemyScoreCountLbl.setTextSize(TypedValue.COMPLEX_UNIT_SP, 84);
}
else{
    אם יש תיקו, אין מנצח
    //draw
    winnerLbl.setText("Draw!");

    yourScoreCountLbl.setTextSize(enemyScoreCountLbl.getTextSize());
}
}
    
```

מחלקות עזר

Game

מחלקה שכוללת את כל המידע של המשחק

הסבר	תכונה
רשימת השאלות	ArrayList<Question> questions
השחקנים	Player player1, player2
הID של המשחק	int id

פעולות

אין פעולות מיוחדות, רק get, set.

Question

הסבר	תכונה
גוף השאלה	String question
מערך של התשובות האפשריות	ArrayList<String> answers
האינדקס של התשובה הנכונה	int correctAnswer
קטגוריית השאלה	Category category
רמת הקושי	DifficultyLevel difficultyLevel

פעולות

אין פעולות מיוחדות, רק get, set.

User

במחלקה זו, כל השדות הם מסוג protected כי player יורשת ממנה

הסבר	תכונה
שם המשתמש	Protected String username
הID של המשתמש (נקרא UID ולא ID כי ככה זה מכונה בפיירבייס)	Protected String uid
הניקוד	Protected int score
סך התשובות הנכונות	Protected int totalCorrect
סך התשובות השגויות	Protected int totalWrong

פעולות

אין פעולות מיוחדות, רק get, set.

Player

יורשת מUser

הסבר	תכונה
מספר של השאלה הנוכחית במשחק	int currentQuestionIndex
רשימה שמציגה עבור כל תשובה שהמשתמש ענה אם היא נכונה או לא	ArrayList<Boolean> isCorrectList

פעולות

חישוב הנקודות:

הנוסחה לחישוב הניקוד קצת מורכבת. היא בנויה כך שלתשובה נכונה יש משקל גדול יותר מלתשובה שגויה. ככל שיש יותר תשובות נכונות, הניקוד גבוה יותר, ולהיפך. מוסיפים 1 לסכום התשובות השגויות כדי שבמקרה ואין תשובה שגויה, לא תהיה חלוקה ב-0. יש המרה לint והכפלה ב-10 כדי שהנקודות יהיו תמיד כפולה של 10.

```
public int calculatePoints() {
    return 10 * (int)(5 * (Math.pow(getTotalCorrectInGame(), 1.2)) /
(getTotalWrongInGame() + 1));
}
```

בנוסף, מומשה פעולת equals (באופן אוטומטי) שמאפשרת להשוות בין השדות של 2 שחקנים ומחזירה אם הם שווים.

MusicService

הסבר	תכונה
קבוע שהוא הID של ההתראה. ערך אקראי.	static final int ONGOING_NOTIFICATION_ID = 4242
קבוע שהוא השם של ההתראה. ערך אקראי.	static final String MUSIC_NOTIFICATION_CHANNEL_ID = "MUSIC"
נגן המדיה, שאחראי לנגן את הקובץ.	MediaPlayer mediaPlayer

פעולות

אתחול:

```
public void onCreate() {
```

```
    super.onCreate();
```

אתחול משתנים של המחלקה

```
//create player
MediaPlayer = MediaPlayer.create(this, R.raw.bg_music);
//play forever
MediaPlayer.setLooping(true);

//listen to volume changes
SettingsFragment.backgroundMusicVolume = new MutableLiveData<>();
SettingsFragment.backgroundMusicVolume.observeForever(new Observer<Float>() {
    @Override
    public void onChanged(Float aFloat) {
        mediaPlayer.setVolume(aFloat, aFloat);
    }
});

SettingsFragment.backgroundMusicVolume.setValue(1f);
{
    התחלת Servicen:

    public int onStartCommand(Intent intent, int flags, int startId) {
        //start playing

        mediaPlayer.start();

        NotificationManager notificationManager = (NotificationManager)
        getSystemService(NOTIFICATION_SERVICE);

        יצירת התראה שמודיעה על Servicen אם ה API LEVEL הוא 26 ומעלה
        כי גוגל מכריחים אותנו

        //if SDK version is over 26, a notification channel is required
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            CharSequence name = "BackgroundMusic";
            NotificationChannel channel = new
            NotificationChannel(MUSIC_NOTIFICATION_CHANNEL_ID, name,
            NotificationManager.IMPORTANCE_DEFAULT);
            channel.setDescription("Is playing music");
            notificationManager.createNotificationChannel(channel);
        }
        String channelId = "";
        if (android.os.Build.VERSION.SDK_INT >= android.os.Build.VERSION_CODES.O) {
            channelId = MUSIC_NOTIFICATION_CHANNEL_ID;
        }
        //add service notification
        Notification notification = new NotificationCompat.Builder(this, channelId)
```

```
.setOngoing(true)
.setSmallIcon(R.drawable.ic_play)
.setCategory(NotificationCompat.CATEGORY_SERVICE)
.build();
```

התחלת הService

```
//start service
startForeground(ONGOING_NOTIFICATION_ID, notification);

return START_STICKY;
{
```

HttpQuestionFetcher
הורדת השאלות מהאינטרנט

הסבר	תכונה
כתובת האתר שממנו מורידים את השאלות	static final String DATABASE_URL_ADDRESS = "https://opentdb.com/api.php"
קוד תגובה מהשרת, שמשמעותו שהבקשה בוצעה בהצלחה	static final int OK_RESPONSE_CODE = 0
כדי לקבל שאלה בקטגוריה מסוימת מהשרת, צריך לשלוח כפרמטר את הID של הקטגוריה הרצויה.	static final int GENERAL_KNOWLEDGE_CATEGORY_ID = 9
	static final int SCIENCE_CATEGORY_ID = 17
	static final int COMPUTER_SCIENCE_CATEGORY_ID = 18
התגובה מהשרת	String response

פעולות

קבלת השאלות מהשרת:

```
public ArrayList<Question> getQuestions(int questionCount, int difficultyLevel, int category) {
    response = "";
```

תהליכים שדורשים חיבור אינטרנט באנדרואיד צריכים להיות בת'רד נפרד, לכן יוצרים להם ת'רד

//run in other thread, because networking isn't allowed in main thread

```
Thread sendThread = new Thread(new Runnable() {
    @Override
    public void run() {
        try {
```

מוסיפים לכתובת הURL את הפרמטרים שמועברים לשרת.

URL לדוגמא:

https://opentdb.com/api.php?amount=10&category=9&difficulty=medium

```
URL url = new URL(DATABASE_URL_ADDRESS + getParams(questionCount, difficultyLevel, category));
```

יוצרים חיבור HTTP

```
HttpURLConnection connection = (HttpURLConnection) url.openConnection();
```

מגדירים שהפרמטרים יעברו כחלק מכתובת הURL

```
connection.setRequestMethod("GET");
```

```
if (connection.getResponseCode() != HttpURLConnection.HTTP_OK)
```

אם ההורדה נכשלה, זורקים שגיאה

```
throw new Exception("HTTP returned response code " +
```

```
connection.getResponseCode());
```

אם ההורדה הצליחה, שמים את התגובה במשתנה response

```
BufferedReader reader = new BufferedReader(new
```

```
InputStreamReader((connection.getInputStream())));
```

```
String line = reader.readLine();
```

```
while (line != null) {
```

```
    response += line;
```

```
    line = reader.readLine();
```

```
}
```

סוגרים את החיבור

```
reader.close();
```

```
} catch (Exception e) {
```

```
    response = "";
```

```
}
```

```
}
```

```
});
```

```
try {
```

```
    sendThread.start();
```

מחכים עד לסוף ריצת הת'רד כדי להמשיך

```
    sendThread.join();
```

```
} catch (Exception e) {
```

```
    return null;
```

```
}
```

אם התגובה ריקה, הייתה שגיאת רשת. לכן לא מחזירים כלום
(מי שקורא לפעולת getQuestions מוודא שלא חזר null)

```
if (response.equals(""))
```

```
    //fetching failed
```

```
    return null;
```

הופכים את השאלות שחוזרות בJSON לArrayList<Question>
בפעולה נפרדת שתפורט בהמשך

```
return deserializeQuestions(DifficultyLevel.values()[difficultyLevel],
```

```
Category.values()[category]);
```

```
}
```

המרה של השאלות מJSON לרשימת שאלות:

```
private ArrayList<Question> deserializeQuestions(DifficultyLevel difficultyLevel, Category
category) {
```

יוצרים רשימת שאלות

```
    ArrayList<Question> questions = new ArrayList<Question>();
```

```
    try {
```

```
        JSONObject jsonObject = new JSONObject(response);
```

```
        int responseCode = Integer.parseInt(jsonObject.getString("response_code"));
```

בודקים אם התגובה מהשרת היא שגיאה, אם כן מחזירים null

```
        if (responseCode != OK_RESPONSE_CODE)
```

```
            return null;
```

```
        JSONArray questionsJsonObject = jsonObject.getJSONArray("results");
```

הופכים את הטקסט של כל שאלה לאובייקט JSON של שאלה

```
        for (int i = 0; i < questionsJsonObject.length(); i++) {
```

```
            JSONObject questionJsonObject = questionsJsonObject.getJSONObject(i);
```

יוצרים שאלה

```
            Question question = new Question();
```

מאתחלים את הערכים של השאלה עם הקטגוריה ורמת הקושי שהמשתמש ביקש

```
            question.setCategory(category);
```

```
            question.setDifficultyLevel(difficultyLevel);
```

מאתחלים את השדות האחרים של השאלה בערכים שלהם מתוך הJSON

```
            String questionString = questionJsonObject.getString("question");
```

```
            question.setQuestion(questionString);
```

```
            String correctAnswer = questionJsonObject.getString("correct_answer");
```

```
            JSONArray incorrectAnswersJsonArray =
```

```
            questionJsonObject.getJSONArray("incorrect_answers");
```

```
            ArrayList<String> answers = new ArrayList<>();
```

```
            for (int j = 0; j < incorrectAnswersJsonArray.length(); j++)
```

```
                answers.add(incorrectAnswersJsonArray.getString(j));
```

```
            answers.add(correctAnswer);
```

תווים מיוחדים בJSON נשמרים בפורמט שונה. לכן, ממירים אותם בחזרה לפורמט טקסט

```
            for (int j = 0; j < answers.size(); j++)
```

```
                answers.set(j, Html.fromHtml(answers.get(j),
```

```
                Html.FROM_HTML_MODE_LEGACY).toString());
```

```
                question.setQuestion(Html.fromHtml(question.getQuestion(),
```

```
                Html.FROM_HTML_MODE_LEGACY).toString());
```

מערבבים את רשימת השאלות


```

Collections.shuffle(answers);
question.setAnswers(answers);

//check the correct answer's index
for(int j=0;j<answers.size();j++)
    if(answers.get(j).equals(correctAnswer))
        question.setCorrectAnswer(j);

questions.add(question);
}

} catch (Exception e) {
    return null;
}

return questions;
}
}

```

מוציאים את האינדקס של התשובה הנכונה

מוסיפים את השאלה הראשונה

אם הייתה שגיאה, מחזירים null

אם לא, המחזירים את השאלות

AnswerRecorder

מחלקה שאחראית על הקלטת התשובה במסך המשחק.

הסבר	תכונה
קבוע ששומר את השפה המוקלטת (עברית)	static final String HEBREW = "iw-IL"
אובייקט של המרת טקסט לדיבור	static SpeechRecognizer speechRecognizer

פעולות

```

public static void startRecording(Context context, Button[] answerButtons) {
    //create speech recognizer

    speechRecognizer = SpeechRecognizer.createSpeechRecognizer(context);

    Intent recognizerIntent = new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
    //set language to Hebrew

    recognizerIntent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,
    RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
    recognizerIntent.putExtra(RecognizerIntent.EXTRA_LANGUAGE, HEBREW);

    //ask permission to record if required
    if (ContextCompat.checkSelfPermission(context,
    android.Manifest.permission.RECORD_AUDIO) != PackageManager.PERMISSION_GRANTED)
        ActivityCompat.requestPermissions((Activity) context, new
    String[]{Manifest.permission.RECORD_AUDIO}, 1);
}

```

יצירת אובייקט של המרת דיבור לטקסט

קביעת השפה

בקשת הרשאת הקלטה אם אין

האזנה לתוצאות ההקלטה

כל הפעולות פה חייבות להיות בכל Listener, לכן אי אפשר למחוק אותן. אני משתמש רק בפעולה .OnResults

```
speechRecognizer.setRecognitionListener(new RecognitionListener() {
    //auto generated methods
    @Override
    public void onReadyForSpeech(Bundle params) {

    }

    @Override
    public void onBeginningOfSpeech() {

    }

    @Override
    public void onRmsChanged(float rmsdB) {

    }

    @Override
    public void onBufferReceived(byte[] buffer) {

    }

    @Override
    public void onEndOfSpeech() {

    }

    @Override
    public void onError(int error) {

    }

    @Override
    public void onResults(Bundle results) {
        הכל נמצא בבלוק try כי לא כל מילה שנקלטה אפשר להמיר למספר בין 1-4, ואם אי אפשר תיזרק שגיאה
        try {
            //get result from bundle
            String word =
results.getStringArrayList(SpeechRecognizer.RESULTS_RECOGNITION).get(0);

            int intResult = -1;
```

המרת התוצאה מסטרינג למספר

(כולל גם אנגלית כדי שיהיה אפשר לשנות את השפה)

```
//check if result is a number between 1-4
if (word.equals("one") || word.equals("אחת") || word.equals("1"))
    intResult = 1;
else if (word.equals("two") || word.equals("שניים") || word.equals("2"))
    intResult = 2;
else if (word.equals("three") || word.equals("שלוש") || word.equals("3"))
    intResult = 3;
else if (word.equals("four") || word.equals("ארבע") || word.equals("4"))
    intResult = 4;
```

הקטנה של התוצאה כי אינדקס במערך מתחיל מ0

```
intResult--;
```

לחיצה על הכפתור של התשובה שנבחרה

```
//get the button of this answer
Button answerButton = answerButtons[intResult];
if (answerButton.isEnabled())
    //click this button
    answerButton.callOnClick();
} catch (Exception e) {
    //value isn't a number
    //or the number is out of range
    //do nothing
}
```

```
//after recording, the speech recognizer is not needed
speechRecognizer.destroy();
}
```

```
@Override
public void onPartialResults(Bundle results) {
}
```

```
@Override
public void onEvent(int eventType, Bundle params) {
```

```
});
}
```

עצירת הקלטה (כשהמשתמש עוזב את כפתור ההקלטה):

```
public static void stopRecording() {
    //stop speech recognizer
```

עצירת listener

```
speechRecognizer.stopListening();
```

}

MusicService

הסבר	תכונה
	static final int ONGOING_NOTIFICATION_ID = 4242

GameViewModel

הסבר	תכונה
המשחק	MutableLiveData<Game> game
האם המשתמש הזה הוא יוצר המשחק (אם לא, הוא הצטרף למשחק שאחר יצר)	boolean isCreator

פעולות

הפעלת סנכרון עם המשחק בפירסטור, כדי שכל שינוי שהמשתמש האחר עשה יעודכן אצלנו, וכל שינוי שנעשה בGame יישלח לשרת:

הגדרת משתנה שישמור את הערך הקודם של myPlayer כדי לדעת אם הוא השתנה

```
private Player previousMyPlayer;
```

```
public void enableGameSyncWithFirestore(Context context){
    previousMyPlayer = getMyPlayer();
```

המרה של ID לסטרינג

```
String gameId = Integer.toString(getGame().getId());
```

```
//when other player is changed, update Game locally
```

```
FirebaseFirestore firestore = FirebaseFirestore.getInstance();
```

הוספת האזנה לשינויים במשחק בפירסטור

```
firestore.collection(GameActivity.GAMES_COLLECTION_PATH).document(gameId).addSnapshotListener(new EventListener<DocumentSnapshot>() {
```

```
@Override
```

```
public void onEvent(@Nullable DocumentSnapshot snapshot, @Nullable
```

```
FirebaseFirestoreException error) {
```

```
    if(error == null && snapshot != null && snapshot.exists()){
```

```
        //no exception
```

כאשר יש שינוי, מקבלים את המשחק אחרי השינוי

```
Game newGame = snapshot.toObject(Game.class);
```

```
if(newGame.getPlayer2() != null){
```

אם השחקן השני ריק, עדיין לא התחיל המשחק, ולכן לא עושים כלום

```
//if player2 is null, game hasn't yet started
```

```
Game gameValue = getGame();
```

אם השחקן הנוכחי הוא היוצר של המשחק, והשחקן שהצטרף (player2) הוא זה שהשתנה, מעדכנים את הערך של שחקן 2 אצלנו.

```
if(isCreator && !newGame.getPlayer2().equals(gameValue.getPlayer2())){
    //player 2 has changed
    gameValue.setPlayer2(newGame.getPlayer2());
    game.setValue(gameValue);
}
```

אם השחקן הנוכחי הוא השחקן שהצטרף, והשחקן שיצר את המשחק הוא זה שהשתנה, מעדכנים את הערך של שחקן 1

```
if(!isCreator && !newGame.getPlayer1().equals(gameValue.getPlayer1())){
    //player 1 has changed
    gameValue.setPlayer1(newGame.getPlayer1());
    game.setValue(gameValue);
}
}
}
}
});
```

הוספת האזנה לשינוי בשחקן הנוכחי, וכשהוא משתנה – עדכון של הפיירסטור

//when my player changed, update in firestore

```
game.observe((LifecycleOwner) context, new Observer<Game>() {
```

```
@Override
```

```
public void onChanged(Game newGame) {
```

צריך לבדוק אם השחקן שלנו השתנה או שהיריב השתנה. אם היריב הוא זה שהשתנה, לא צריך לעדכן בפיירסטור כי זה כבר מעודכן.

אם המשתנה ששומר את המצב הקודם של השחקן ריק, והשחקן לא ריק, השחקן השתנה הנוכחי השתנה גם השחקן הנוכחי שונה מהמצב הקודם שלו, הוא השתנה

לכן, צריך לשלוח את game לפיירסטור

```
if((previousMyPlayer == null && getMyPlayer() != null) ||
(previousMyPlayer != null && !previousMyPlayer.equals(getMyPlayer()))
    //player has changed
```

```
firestore.collection(GameActivity.GAMES_COLLECTION_PATH).document(gameId).set(newGame);
```

```
addOnFailureListener(new OnFailureListener() {
```

```
@Override
```

אם העדכון נכשל, הייתה שגיאת רשת. לכן, מעדכנים את המשתמש ויוצאים מהמשחק.

```
Public void onFailure(@NonNull Exception e) {
```

```
    Toast.makeText(context, "Connection error!",
```

```
    Toast.LENGTH_SHORT).show();
```

```
    //TODO: exit screen
```

```

    }
    });

    מעדכנים את הערך ה"קודם" של השחק הנוכחי,
    (בפעם הבאה שיהיה שינוי, השחקן הנוכחי יהיה הקודם).

    if(getMyPlayer() == null)
        //copy constructor doesn't work with null
        previousMyPlayer = null;
    else
        previousMyPlayer = new Player(getMyPlayer());
    }
    });
}

```

ScoreListViewHolder

רכיב שממנו מורכבת כל שורה ברשימת המשתמשים במסך הניקוד. כולל את ה View שמציגים את המידע על השחקן.

הסבר	תכונה
מציג את המקום של השחקן ברשימת המשתמשים	TextView place
מציג את שם המשתמש	TextView username
מציג את הניקוד	TextView score

פעולות

יצירת ה ViewHolder:

```

public ScoreListViewHolder(@NonNull View itemView) {
    super(itemView);

    מוצאים את כל אחד מה Views של XML, ושמים את הערך שלו בשדה המתאים
    place = itemView.findViewById(R.id.placeLbl);
    username = itemView.findViewById(R.id.usernameLbl);
    score = itemView.findViewById(R.id.scoreLbl);
}

```

ScoreListAdapter

מחלקה שאחראית על ניהול תוכן רשימת המשתמשים במסך הניקוד

הסבר	תכונה
הקונטקסט של ה Activity. הכרחי בשביל פעולת onCreateViewHolder	Context context
רשימת המשתמשים שהתקבלה מפייסבוק, ובה נמצא הניקוד של כל משתמש	ArrayList<User> users

פעולות

יצירה של ViewHolder:

```

public ScoreListViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {

```

```

LayoutInflater inflater = LayoutInflater.from(context);
        יצירה של שורה חדשה ברשימה
        View view = inflater.inflate(R.layout.recycler_view_score, parent, false);
        יצירת ViewHolder והחזרה שלו
        return new ScoreListViewHolder(view);
    }

```

קישור בין ViewHolder לבין המשתמש והכנסת המידע על המשתמש לViewHolder

```

public void onBindViewHolder(@NonNull ScoreListViewHolder holder, int position) {
    User user = users.get(position);
    בכל View בשורה של המשתמש, מכניסים את הערך (מקום/שם משתמש/ניקוד) של המשתמש שברשימה
    holder.place.setText(Integer.toString(position + 1));
    holder.username.setText(user.getUsername());
    holder.score.setText(Integer.toString((int) user.getScore()));
}

```

קבצי תצורה

קבצי Layout

הסבר	תכונה
קובץ העיצוב של מסך המשחק	activity_game.xml
קובץ העיצוב של מסך ההתחברות	activity_login.xml
קובץ העיצוב של מסך התפריט הראשי (בתוכו נמצאים fragments מסכי הניקוד, יצירת משחק, התחברות למשחק והגדרות)	activity_main_menu.xml
קובץ העיצוב של מסך סיום המשחק	fragment_end_game.xml
קובץ העיצוב של מסך הצגת ID של המשחק	fragment_game_id.xml
קובץ העיצוב של מסך ההצטרפות למשחק	fragment_join_game.xml
קובץ העיצוב של מסך הטעינה (לא מופיע ברשימת המסכים כי הוא כולל עיצוב בלבד ואין לו פעולות למעט onCreate)	fragment_loading.xml
קובץ העיצוב של מסך יצירת משחק	fragment_new_game.xml
קובץ העיצוב של מסך הניקוד	fragment_score.xml
קובץ העיצוב של מסך ההגדרות	fragment_settings.xml
קובץ העיצוב של מסך ההמתנה למשתמש בסוף המשחק (לא מופיע ברשימת המסכים כי הוא כולל עיצוב בלבד ואין לו פעולות למעט onCreate)	fragment_wait_to_enemy.xml
קובץ העיצוב של כל שורה ברשימת המשתמשים במסך הניקוד	recycler_view_score.xml

קבצי Menu

תכונה	הסבר
main_menu.xml	קובץ בו מוגדרות האפשרויות לבחירה שבתפריט הראשי

קבצי Navigation

תכונה	הסבר
main_nav.xml	קובץ בו מוגדרים fragments שניתן לעבור אליהן בעזרת התפריט הראשי

בסיסי נתונים

בפרויקט השתמשתי ב-3 בסיסי נתונים שונים. אפרט כאן על צורת השימוש בהם והמטרות של כל אחד.

SharedPreferences

זהו בסיס נתונים לוקאלי מובנה ב-Android. השתמשתי בו כדי לשמור את שם המשתמש והסיסמא בהתחברות, כך שבהתחברות הבאה הם יופיעו אוטומטית (אם המשתמש מעוניין בכך).

פעולות:

שמירת שם המשתמש והסיסמא:

```
private void saveLoginDataToSharedPreferences() {
    יצירת אובייקט שבו נשתמש בשביל לגשת לSharedPreferences
    SharedPreferences sharedPreferences =
    getSharedPreferences(LOGIN_PREFERENCES_FILE, MODE_PRIVATE);

    אם המשתמש סימן "זכור אותי" – שומרים את שם המשתמש והסיסמא:
    if (rememberMeCb.isChecked()) {
        //save to shared preferences

        יצירת "עורך"
        SharedPreferences.Editor sharedPreferencesEditor = sharedPreferences.edit();

        הכנסה של שם המשתמש
        sharedPreferencesEditor.putString(LOGIN_PREFERENCES_USERNAME,
        usernameTxt.getText().toString());

        הכנסה של הסיסמא
        sharedPreferencesEditor.putString(LOGIN_PREFERENCES_PASSWORD,
        passwordTxt.getText().toString());

        שמירת השינויים
        //apply changes
        sharedPreferencesEditor.apply();
    } else if (sharedPreferences.getAll().size() != 0) { //there is already shared preferences
        אם המשתמש לא סימן "זכור אותי" – מוחקים את שם המשתמש והסיסמא ששמורים (אם שמורים)
        //delete shared preferences
        sharedPreferences.edit().clear().apply();
    }
}
```



```

    }
}

שליפת שם המשתמש והסיסמא (אם נשמרו):
private void loadLoginDataFromSharedPreferences() {
    //try load the data

    יצירת אובייקט שבו נשתמש בשביל לגשת ל SharedPreferences
    SharedPreferences sharedPreferences =
    getSharedPreferences(LOGIN_PREFERENCES_FILE, MODE_PRIVATE);

    קריאת שם המשתמש
    String username = sharedPreferences.getString(LOGIN_PREFERENCES_USERNAME,
    null);

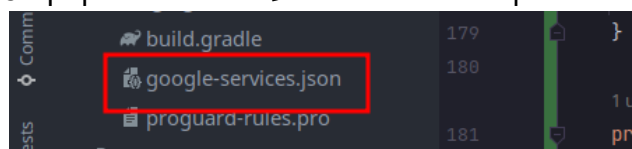
    קריאת הסיסמא
    String password = sharedPreferences.getString(LOGIN_PREFERENCES_PASSWORD, null);

    הכנסה של שם המשתמש והסיסמא לתצוגה
    //if shared preferences was found, show saved username and password in EditTexts
    if (username != null)
        usernameTxt.setText(username);
    if (password != null)
        passwordTxt.setText(password);
}

```

Firestore

בסיס נתונים זה, כמו שכבר כתבתי, הוא בסיס נתונים בענן שניתן כחלק משירות Firebase של גוגל. כדי לשמור בו מידע, חיברתי את האפליקציה לפיירבייס באמצעות הורדה של הקובץ google-services.json



והדבקה שלו באפליקציה.

בנוסף, הצטרכתי להוסיף לקובץ build.gradle של האפליקציה את הdependencies הבאים:

```

implementation platform('com.google.firebase:firebase-bom:32.0.0')
implementation 'com.google.firebase:firebase-firestore-ktx'
implementation 'com.google.firebase:firebase-auth-ktx'
implementation 'com.google.api:api-common:2.2.1'

```

כדי לשמור מידע בפיירסטור, צריך להשתמש באובייקט

Firestore.getInstance()

המידע נשמר באוספים (מבנה נתונים שמתפקד כמו HashMap) של אובייקט מסוג מסויים (במקרה שלי – User או Game).

כדי לשמור את המידע, משתמשים באובייקט של פיירסטור:

firestore.collection(שם האוסף).document(המפתח של הערך).set(הערך שמכניסים);

כדי לוודא שהשמירה הצליחה, צריך להשתמש במתודה

.addOnCompleteListener(new OnCompleteListener<Void>() {

```
@Override
public void onComplete(@NonNull Task<Void> task) {
    if(task.isSuccessful()){
        כאן נמצא הקוד שרץ אם השמירה הצליחה
    }
    else{
        כאן נמצא הקוד שרץ אם השמירה נכשלה
    }
}
});
```

שליפת מידע מהפיירסטור מתבצעת כך:

```
firestore.collection(שם האוסף).document(המפתח של הערך).get();
```

כדי לקרוא את המידע, צריך להוסיף גם כאן listener. לרוב, העדפתי להשתמש ב-addSuccessListener:

```
.addSuccessListener(new OnSuccessListener<DocumentSnapshot>() {
    @Override
    public void onSuccess(DocumentSnapshot documentSnapshot) {
        קריאת האובייקט שהתקבל
        Object object = documentSnapshot.toObject(Object.class);
    }
});
```

אלו דרכי שמירת ושליפת המידע המרכזיות בפיירסטור. אפשר לעשות איתו עוד מספר פעולות דומות (כמו: שליפה של אוסף שלם במקום ערך מסוים, הוספת listener שיודיע על שינויים בערך ועוד). על כל אחת מהפעולות השונות שהשתמשתי בהן פירטתי יותר באריכות במקום שהשתמשתי בה. (למשל: דוגמא להאזנה לשינוי של משתנה אפשר למצוא במחלקת GameViewModel).

[Open Trivia DB](#)

בבסיס נתונים זה השתמשתי רק כדי להוריד את השאלות, ולא שמרתי בו מידע בעצמי. לכן, לא אפרט עליו כאן. אופן השימוש המלא בו מפורט לעיל במחלקת HttpQuestionFetcher.

מדריך למשתמש

תיאור כללי

המשחק Brain Battle הוא משחק טריוויה אונליין ל-2 שחקנים. השחקנים מתחרים זה בזה, עונים על שאלות וצוברים נקודות. המנצח הוא השחקן בעל מספר הנקודות הגדול ביותר.

האפליקציה כוללת את הפיצ'רים הבאים:

- יצירת משתמש ומחיקתו, התנתקות והתחברות
- יצירת משחק
- התחברות למשחק קיים
- צפייה בסטטיסטיקות ובניקוד
- שליטה במוסיקת הרקע
- הקלטת תשובה באמצעות המיקרופון

הגבלות

שם המשתמש חייב לכלול לפחות תו אחד. הסיסמא צריכה לכלול לפחות 6 תווים. התשובה המוקלטת צריכה להיות בעברית. רוב הפיצ'רים באפליקציה דורשים חיבור לאינטרנט. אם אין חיבור, תוצג הודעת שגיאה והפיצ'ר לא יעבוד.

הרשאות

חיבור לאינטרנט:

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

שימוש במיקרופון:

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

הפעלת Service:

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

הצהרות

Service של מוזיקת הרקע:

```
</"service android:name=".MusicService">
```

גרסת Android מינימלית

24 :Minimum SDK

32 :Target SDK

גרסאות ומכשירים שעליהם נבדקה האפליקציה

Xiaomi Redmi 7A, Android Version 10.0, API 29 – מכשיר פיזי

FWVGA, Android Version 11.0, API 30 – אמולטור

Pixel 5, Android Version 11.0, API 30 – אמולטור

רפלקציה

נהייתי מאוד מרוב חלקי העבודה על הפרויקט. בנוסף, חברתי הרבה ידע בתכנות בכלל ובתכנות לאנדרואיד בפרט, שאני בטוח שישמש אותי בהמשך.

עם זאת, היו לא מעט קשיים. היו הרבה נושאים שאינם מתוכנית הלימודים (את המרכזיים שבהם פירטתי לעיל) והייתי צריך ללמוד בעצמי. נעזרתי באתרי אינטרנט שבהם יש מדריכים ועצות לפתרון באגים, שחלקם מופיעים בביבליוגרפיה.

יתר על כן, למדתי להשתמש בצורה טובה יותר בדיבאגר ובפיצ'רים הרבים שהוא כולל כדי להצליח למצוא בקלות באגים בתוכנית. עם הזמן למדתי גם להשתמש בlogcat שבלא מעט פעמים כותב בפירוש מה הבעיה ואיפה.

בראייה לאחור, אם הייתי עושה את הפרויקט מחדש, הייתי עושה חלקים מתיק הפרויקט לפני תחילת כתיבת הקוד עצמו, מכיוון שלא תמיד ידעתי כבר בזמן כתיבת הקוד מה תהיה המטרה המדויקת של כל כפתור, לייבל וכו' או מה התפקיד המדויק של מסך שתכננתי, וכתוצאה מכך מחקתי קטעי קוד גדולים.

אם היה לי יותר זמן, ישנם פיצ'רים רבים שהייתי רוצה להוסיף לפרויקט: timeout למענה על שאלות, התחשבות במשך הזמן שלקח לענות כחלק מהניקוד, עיצוב טוב יותר, אפשרות למשחק של שחקן יחיד או של מספר שחקנים, ועוד ועוד. אם הייתי יכול, הייתי רוצה גם להחליף את האחסון בפייירסטור בשרת עצמאי שלי שיתן לי יותר שליטה על התוכן וצורת הארגון של המידע.

לסיכום, אני מרוצה מהפרויקט, הן מהתוצר והן מהידע המשמעותי שרכשתי המשלך העבודה על הפרויקט.

ביבליוגרפיה

ה-API שלי בסיס הנתונים ממנו שולפים את השאלות - https://opentdb.com/api_config.php

סטאק אוברפלווא - [/https://stackoverflow.com](https://stackoverflow.com)

יוטיוב - [/https://www.youtube.com](https://www.youtube.com)

התיעוד הרשמי של אנדרואיד - <https://developer.android.com/docs>

התיעוד הרשמי של פיירבייס - <https://firebase.google.com/docs>

נספחים

קוד הפרויקט:

הערה: ישנם מספר פיצ'רים קטנים שעדיין לא מומשו. לכן, יהיו שינויים קלים בקוד בעתיד. בנוסף, וורד מעוות קבצי xml משיקוליו שלו.

כדי לראות את הקוד המעודכן בצורה נוחה, מומלץ להיכנס לקישור הבא:

[/https://github.com/YehudaElyasaf/brain-battle](https://github.com/YehudaElyasaf/brain-battle)

:EndGameFragment.java:

```
package com.example.trivia;
```

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

```
import android.os.AsyncTask;
```

```
import android.os.Bundle;
```

```
import androidx.fragment.app.Fragment;
```

```
import androidx.lifecycle.ViewModelProvider;
```

```
import android.os.CountDownTimer;
```

```
import android.util.TypedValue;
```

```
import android.view.LayoutInflater;
```

```
import android.view.View;
```

```
import android.view.ViewGroup;
```

```
import android.widget.ImageButton;
```

```
import android.widget.TextView;
```

```
import java.util.Timer;
```

```
import java.util.TimerTask;
```

```

/**
 * A simple {@link Fragment} subclass.
 * Use the {@link EndGameFragment#newInstance} factory method to
 * create an instance of this fragment.
 *
 */
public class EndGameFragment extends Fragment implements View.OnClickListener {
    // the fragment initialization parameters, e.g. ARG_ITEM_NUMBER
    private static final String ARG_PARAM1 = "param1";
    private static final String ARG_PARAM2 = "param2";

    private GameViewModel gameVM;

    private TextView yourScoreCountLbl;
    private TextView enemyScoreCountLbl;
    private TextView winnerLbl;

    private ImageButton endGameReplayBtn;
    private ImageButton endGameHomeBtn;
    private ImageButton endGameShareBtn;

    private String mParam1;
    private String mParam2;

    /**
     * Use this factory method to create a new instance of
     * this fragment using the provided parameters.

```

```

*
* @param param1 Parameter 1.
* @param param2 Parameter 2.
* @return A new instance of fragment EndGameFragment.
*/
public static EndGameFragment newInstance(String param1, String param2) {
    EndGameFragment fragment = new EndGameFragment();
    Bundle args = new Bundle();
    args.putString(ARG_PARAM1, param1);
    args.putString(ARG_PARAM2, param2);
    fragment.setArguments(args);
    return fragment;
}

public EndGameFragment() {
    // Required empty public constructor
}

@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    if (getArguments() != null) {
        mParam1 = getArguments().getString(ARG_PARAM1);
        mParam2 = getArguments().getString(ARG_PARAM2);
    }
}

```



```
@Override
```

```
public View onCreateView(LayoutInflater inflater, ViewGroup container,
```

```
Bundle savedInstanceState) {
```

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

```
gameVM = new ViewModelProvider(getActivity()).get(GameViewModel.class);
```

```
yourScoreCountLbl = view.findViewById(R.id.yourScoreCountLbl);
```

```
enemyScoreCountLbl = view.findViewById(R.id.enemyScoreCountLbl);
```

```
winnerLbl = view.findViewById(R.id.winnerLbl);
```

```
endGameReplayBtn = view.findViewById(R.id.endGameReplayBtn);
```

```
endGameHomeBtn = view.findViewById(R.id.endGameHomeBtn);
```

```
endGameShareBtn = view.findViewById(R.id.endGameShareBtn);
```

```
endGameReplayBtn.setOnClickListener(this);
```

```
endGameHomeBtn.setOnClickListener(this);
```

```
endGameShareBtn.setOnClickListener(this);
```

```
showResults(view);
```

```
// Inflate the layout for this fragment
```

```
return view;
```

```
}
```

```
private void showResults(View view) {
```

```
int yourScore = gameVM.getMyPlayer().calculatePoints();
```

```
int enemyScore = gameVM.getOtherPlayer().calculatePoints();
```

```
yourScoreCountLbl.setText(Integer.toString(yourScore));
```

```

enemyScoreCountLbl.setText(Integer.toString(enemyScore));

if(yourScore > enemyScore) {
    //you won
    winnerLbl.setText("You won!");
    yourScoreCountLbl.setTextSize(TypedValue.COMPLEX_UNIT_SP, 84);

    //decreaseScore(enemyScore, enemyScoreCountLbl);
}
else if(yourScore < enemyScore){
    //you lost
    winnerLbl.setText(gameVM.getOtherPlayer().getUsername() + " won!");
    enemyScoreCountLbl.setTextSize(TypedValue.COMPLEX_UNIT_SP, 84);
    //decreaseScore(yourScore, yourScoreCountLbl);
}
else{
    //draw
    winnerLbl.setText("Draw!");

    yourScoreCountLbl.setTextSize(enemyScoreCountLbl.getTextSize());
    //decreaseScore(yourScore, yourScoreCountLbl);
    //decreaseScore(enemyScore, enemyScoreCountLbl);
}
}

@Override
public void onClick(View v) {
    switch (v.getId()){

```

```

        case R.id.endGameReplayBtn:
            break;
        case R.id.endGameHomeBtn:
            Intent intent = new Intent(getContext(), MainMenuActivity.class);
            startActivity(intent);
            getActivity().finish();
            break;
        case R.id.endGameShareBtn:
            break;
    }
}
}

```

```

:HttpQuestionFetcher.java:
package com.example.trivia;

```

```

import android.text.Html;
import android.text.PrecomputedText;

```

```

import org.json.JSONArray;
import org.json.JSONObject;

```

```

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.net.HttpURLConnection;

```

```

import java.net.URI;
import java.net.URL;
import java.util.ArrayList;
import java.util.Collection;
import java.util.Collections;
import java.util.List;
import java.util.Locale;

public class HttpQuestionFetcher {
    private static final String DATABASE_URL_ADDRESS = "https://opentdb.com/api.php";

    private static final int OK_RESPONSE_CODE = 0;
    private static final int GENERAL_KNOWLEDGE_CATEGORY_ID = 9;
    private static final int SCIENCE_CATEGORY_ID = 17;
    private static final int COMPUTER_SCIENCE_CATEGORY_ID = 18;

    private String response;

    public ArrayList<Question> getQuestions(int questionCount, int difficultyLevel, int
category) {
        response = "";
        //run in other thread, because networking isn't allowed in main thread
        Thread sendThread = new Thread(new Runnable() {
            @Override
            public void run() {
                try {
                    URL url = new URL(DATABASE_URL_ADDRESS + getParams(questionCount,
difficultyLevel, category));

```

```

        HttpURLConnection connection = (HttpURLConnection) url.openConnection();
        connection.setRequestMethod("GET");

        if (connection.getResponseCode() != HttpURLConnection.HTTP_OK)
            throw new Exception("HTTP returned response code " +
connection.getResponseCode());

        BufferedReader reader = new BufferedReader(new
InputStreamReader((connection.getInputStream())));

        String line = reader.readLine();
        while (line != null) {
            response += line;
            line = reader.readLine();
        }
        reader.close();
    } catch (Exception e) {
        response = "";
    }

}

});

try {
    sendThread.start();
    sendThread.join();
} catch (Exception e) {
    return null;
}

if (response.equals(""))

```

```
//fetching failed

return null;

return deserializeQuestions(DifficultyLevel.values()[difficultyLevel],
Category.values()[category]);
}

private String getParams(int questionCount, int difficultyLevel, int category) {
    String params = "";

    params += "?amount=";
    params += Integer.toString(questionCount);

    params += "&type=multiple";

    params += "&difficulty=";
    if (difficultyLevel == DifficultyLevel.EASY.ordinal())
        params += "easy";
    else if (difficultyLevel == DifficultyLevel.MEDIUM.ordinal())
        params += "medium";
    else if (difficultyLevel == DifficultyLevel.HARD.ordinal())
        params += "hard";

    if (category != Category.ALL.ordinal()) {
        params += "&category=";
        if (category == Category.GENERAL_KNOWLEDGE.ordinal())
            params += Integer.toString(GENERAL_KNOWLEDGE_CATEGORY_ID);
        else if (category == Category.SCIENCE.ordinal())
            params += Integer.toString(SCIENCE_CATEGORY_ID);
    }
}
```

```

else if (category == Category.COMPUTER_SCIENCE.ordinal())
    params += Integer.toString(COMPUTER_SCIENCE_CATEGORY_ID);
}

return params;
}

private ArrayList<Question> deserializeQuestions(DifficultyLevel difficultyLevel, Category
category) {
    ArrayList<Question> questions = new ArrayList<Question>();
    try {
        JSONObject jsonObject = new JSONObject(response);
        int responseCode = Integer.parseInt(jsonObject.getString("response_code"));
        if (responseCode != OK_RESPONSE_CODE)
            return null;

        JSONArray questionsJsonObject = jsonObject.getJSONArray("results");

        for (int i = 0; i < questionsJsonObject.length(); i++) {
            JSONObject questionJsonObject = questionsJsonObject.getJSONObject(i);
            Question question = new Question();

            question.setCategory(category);
            question.setDifficultyLevel(difficultyLevel);

            String questionString = questionJsonObject.getString("question");
            question.setQuestion(questionString);

            String correctAnswer = questionJsonObject.getString("correct_answer");

```

```

JSONArray incorrectAnswersJSONArray =
questionJsonObject.getJSONArray("incorrect_answers");

ArrayList<String> answers = new ArrayList<>();

for (int j = 0; j < incorrectAnswersJSONArray.length(); j++)
    answers.add(incorrectAnswersJSONArray.getString(j));
answers.add(correctAnswer);

for (int j = 0; j < answers.size(); j++)
    answers.set(j, Html.fromHtml(answers.get(j),
Html.FROM_HTML_MODE_LEGACY).toString());

question.setQuestion(Html.fromHtml(question.getQuestion(),
Html.FROM_HTML_MODE_LEGACY).toString());

Collections.shuffle(answers);
question.setAnswers(answers);

//check the correct answer's index
for(int j=0;j<answers.size();j++)
    if(answers.get(j).equals(correctAnswer))
        question.setCorrectAnswer(j);

questions.add(question);
}

} catch (Exception e) {
    return null;
}

```



```
        return questions;
    }
}
```

:SettingsFragment.java:

```
package com.example.trivia;
```

```
import android.os.Bundle;
```

```
import androidx.annotation.NonNull;
```

```
import androidx.fragment.app.Fragment;
```

```
import androidx.lifecycle.LifecycleOwner;
```

```
import androidx.lifecycle.LiveData;
```

```
import androidx.lifecycle.MutableLiveData;
```

```
import androidx.lifecycle.Observer;
```

```
import android.view.LayoutInflater;
```

```
import android.view.View;
```

```
import android.view.ViewGroup;
```

```
import android.widget.ImageButton;
```

```
import android.widget.SeekBar;
```

```
/**
```

```
 * A simple {@link Fragment} subclass.
```

```
 * Use the {@link SettingsFragment#newInstance} factory method to
```

```

* create an instance of this fragment.
*/

public class SettingsFragment extends Fragment implements View.OnClickListener {

    public static MutableLiveData<Float> backgroundMusicVolume;

    private ImageButton deleteUserBtn;
    private ImageButton editUsernameBtn;
    private ImageButton logoutBtn;
    private ImageButton muteBtn;
    private SeekBar volumeSb;

    // the fragment initialization parameters, e.g. ARG_ITEM_NUMBER
    private static final String ARG_PARAM1 = "param1";
    private static final String ARG_PARAM2 = "param2";

    private String mParam1;
    private String mParam2;

    public SettingsFragment() {
        // Required empty public constructor
    }

    /**
     * Use this factory method to create a new instance of
     * this fragment using the provided parameters.
     *
     * @param param1 Parameter 1.
     * @param param2 Parameter 2.

```

```

* @return A new instance of fragment SettingsFragment.
*/
public static SettingsFragment newInstance(String param1, String param2) {
    SettingsFragment fragment = new SettingsFragment();
    Bundle args = new Bundle();
    args.putString(ARG_PARAM1, param1);
    args.putString(ARG_PARAM2, param2);
    fragment.setArguments(args);
    return fragment;
}

@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    if (getArguments() != null) {
        mParam1 = getArguments().getString(ARG_PARAM1);
        mParam2 = getArguments().getString(ARG_PARAM2);
    }
}

private void initView(View v) {
    deleteUserBtn = v.findViewById(R.id.deleteUserImgBtn);
    editUsernameBtn = v.findViewById(R.id.editUsernameImgBtn);
    logoutBtn = v.findViewById(R.id.logoutImgBtn);
    muteBtn = v.findViewById(R.id.muteImgBtn);
    initVolumeSb(v);

    deleteUserBtn.setOnClickListener(this);
}

```

```

editUsernameBtn.setOnClickListener(this);

muteBtn.setOnClickListener(this);
}

private void initVolumeSb(View v) {
    volumeSb = v.findViewById(R.id.volumeSb);
    //multiplied by 100 because volume is between 0-1 and progress is between 0-100
    volumeSb.setProgress(backgroundMusicVolume.getValue().intValue() * 100);
    volumeSb.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
        @Override
        public void onProgressChanged(SeekBar seekBar, int progress, boolean fromUser) {
            backgroundMusicVolume.setValue((float)progress / 100);
        }

        @Override
        public void onStartTrackingTouch(SeekBar seekBar) {

        }

        @Override
        public void onStopTrackingTouch(SeekBar seekBar) {

        }
    });
}

@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,

```

```

        Bundle savedInstanceState) {

    // Inflate the layout for this fragment
    View view = inflater.inflate(R.layout.fragment_settings, container, false);
    initView(view);

    return view;
}

private void toggleMute() {
    if (volumeSb.isEnabled()) {
        //mute
        backgroundMusicVolume.setValue(0f);
        muteBtn.setImageDrawable(getActivity().getDrawable(R.drawable.ic_volume_off));
        volumeSb.setEnabled(false);
    } else {
        //unmute
        backgroundMusicVolume.setValue(1f);
        muteBtn.setImageDrawable(getActivity().getDrawable(R.drawable.ic_volume));
        volumeSb.setEnabled(true);
    }
}

@Override
public void onClick(View v) {
    switch (v.getId()) {
        case R.id.deleteUserImgBtn:
            break;
    }
}

```

```

        case R.id.editUsernameImgBtn:
            break;
        case R.id.logoutImgBtn:
            break;
        case R.id.muteImgBtn:
            toggleMute();
            break;
    }
}
}

```

:WaitToEnemyFragment.java:

```
package com.example.trivia;
```

```
import android.os.Bundle;
```

```
import androidx.fragment.app.Fragment;
```

```
import android.view.LayoutInflater;
```

```
import android.view.View;
```

```
import android.view.ViewGroup;
```

```
/**
```

```
 * A simple {@link Fragment} subclass.
```

```
 * Use the {@link WaitToEnemyFragment#newInstance} factory method to
```

```
 * create an instance of this fragment.
```

```
 */
```

```

public class WaitToEnemyFragment extends Fragment {

    // TODO: Rename parameter arguments, choose names that match
    // the fragment initialization parameters, e.g. ARG_ITEM_NUMBER
    private static final String ARG_PARAM1 = "param1";
    private static final String ARG_PARAM2 = "param2";

    // TODO: Rename and change types of parameters
    private String mParam1;
    private String mParam2;

    public WaitToEnemyFragment() {
        // Required empty public constructor
    }

    /**
     * Use this factory method to create a new instance of
     * this fragment using the provided parameters.
     *
     * @param param1 Parameter 1.
     * @param param2 Parameter 2.
     * @return A new instance of fragment WaitToEnemyFragment.
     */
    // TODO: Rename and change types and number of parameters
    public static WaitToEnemyFragment newInstance(String param1, String param2) {
        WaitToEnemyFragment fragment = new WaitToEnemyFragment();
        Bundle args = new Bundle();
        args.putString(ARG_PARAM1, param1);

```

```
args.putString(ARG_PARAM2, param2);
fragment.setArguments(args);
return fragment;
}
```

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    if (getArguments() != null) {
        mParam1 = getArguments().getString(ARG_PARAM1);
        mParam2 = getArguments().getString(ARG_PARAM2);
    }
}
```

```
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
    Bundle savedInstanceState) {
    // Inflate the layout for this fragment
    return inflater.inflate(R.layout.fragment_wait_to_enemy, container, false);
}
}
```

:ScoreFragment.java:

```
package com.example.trivia;
```

```
import android.os.Bundle;
```



```
import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;
import androidx.fragment.app.FragmentTransaction;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;
import android.widget.Toast;

import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.firestore.DocumentSnapshot;
import com.google.firebase.firestore.FirebaseFirestore;
import com.google.firebase.firestore.QuerySnapshot;
import com.mikhaellopez.circularprogressbar.CircularProgressBar;

import java.util.ArrayList;

/**
 * A simple {@link Fragment} subclass.
 * Use the {@link ScoreFragment#newInstance} factory method to
 * create an instance of this fragment.
 */
public class ScoreFragment extends Fragment {
```

```

RecyclerView scoreRv;

CircularProgressBar successPercentagePb;

TextView totalScoreLbl;

TextView totalCorrectLbl;

TextView totalWrongLbl;

TextView successPercentageLbl;


ArrayList<User> users;

User currentUser;


public ScoreFragment() {
    // Required empty public constructor
}


@Override

public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
}


@Override

public View onCreateView(LayoutInflater inflater, ViewGroup container,
                           Bundle savedInstanceState) {
    // Inflate the layout for this fragment
    View v = inflater.inflate(R.layout.fragment_score, container, false);
    scoreRv = v.findViewById(R.id.scoreRv);
    successPercentagePb = v.findViewById(R.id.successPercentagePb);
    totalScoreLbl = v.findViewById(R.id.totalScoreLbl);
    totalCorrectLbl = v.findViewById(R.id.totalCorrectLbl);
}

```

```

totalWrongLbl = v.findViewById(R.id.totalWrongLbl);
successPercentageLbl = v.findViewById(R.id.successPercentageLbl);

Fragment loadingFragment = new LoadingFragment();
getChildFragmentManager().beginTransaction().replace(R.id.scoreLayout,
loadingFragment).commit();

//fetch user list
FirebaseFirestore.getInstance().
    collection(GameActivity.USERS_COLLECTION_PATH).get().addOnFailureListener(new
OnFailureListener() {
    @Override
    public void onFailure(@NonNull Exception e) {
        Toast.makeText(getContext(), "Connection error!",
Toast.LENGTH_SHORT).show();
        //back to main manu
        FragmentTransaction ft = getChildFragmentManager().beginTransaction();
        ft.replace(R.id.mainFragmentContainer, new NewGameFragment()).commit();
    }
}).addOnSuccessListener(new OnSuccessListener<QuerySnapshot>() {
    @Override
    public void onSuccess(QuerySnapshot queryDocumentSnapshots) {
        ArrayList<User> fetchedUserList = new ArrayList<>();
        for(DocumentSnapshot documentSnapshot :
queryDocumentSnapshots.getDocuments())
            if(documentSnapshot.exists())
                fetchedUserList.add(documentSnapshot.toObject(User.class));

        users = fetchedUserList;
        //find current user

```

```

        for(User user : users)
            if(user.getId().equals(FirebaseAuth.getInstance().getId()))
                currentUser = user;
            if(currentUser == null)
                Toast.makeText(getContext(), "Current user not found!",
                    Toast.LENGTH_SHORT).show();

getChildFragmentManager().beginTransaction().hide/loadingFragment).commit();

        showScore();
    }
});

return v;
}

private void showScore() {
    totalScoreLbl.setText(Integer.toString(currentUser.getScore()));
    totalCorrectLbl.setText(Integer.toString(currentUser.getTotalCorrect()));
    totalWrongLbl.setText(Integer.toString(currentUser.getTotalWrong()));

    //show correct percentage progress bar
    successPercentagePb.setProgress(ProgressBarColor(MyColor.CORRECT_GREEN));
    successPercentagePb.setProgressWidth(15);
    successPercentagePb.setBackgroundProgressBarColor(MyColor.WRONG_RED);
    successPercentagePb.setBackgroundProgressBarWidth(10);

    int totalAnswers = currentUser.getTotalCorrect() + currentUser.getTotalWrong();
    int progress;

```

```

if(totalAnswers > 0)
    progress = (100 * currentUser.getTotalCorrect()) / totalAnswers;
else
    //avoid division by zero
    progress = 0;

successPercentagePb.setProgressWithAnimation(progress, (long)1000);
successPercentageLbl.setText(Integer.toString(progress) + "%");

//show user list
users.sort((o1, o2) -> (int)(o2.getScore() - o1.getScore()));
ScoreListAdapter scoreListAdapter = new ScoreListAdapter(getContext(), users);
scoreRv.setAdapter(scoreListAdapter);
scoreRv.setLayoutManager(new LinearLayoutManager(getContext()));
}
}

```

:JoinGameFragment.java:

```
package com.example.trivia;
```

```

import static com.example.trivia.GameActivity.CATEGORY_INDEX;
import static com.example.trivia.GameActivity.DIFFICULTY_LEVEL_INDEX;
import static com.example.trivia.GameActivity.IS_NEW_GAME_EXTRA;
import static com.example.trivia.GameActivity.QUESTIONS_COUNT_INDEX;

```

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

```
import android.os.Bundle;
```

```
import androidx.fragment.app.Fragment;

import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import com.google.android.gms.tasks.OnSuccessListener;
import com.google.firebase.firestore.DocumentChange;
import com.google.firebase.firestore.DocumentSnapshot;
import com.google.firebase.firestore.FirebaseFirestore;
import com.google.firebase.firestore.QuerySnapshot;

import java.util.ArrayList;

/**
 * A simple {@link Fragment} subclass.
 * Use the {@link JoinGameFragment#newInstance} factory method to
 * create an instance of this fragment.
 */
public class JoinGameFragment extends Fragment implements View.OnClickListener {
    // the fragment initialization parameters, e.g. ARG_ITEM_NUMBER
    private static final String ARG_PARAM1 = "param1";
    private static final String ARG_PARAM2 = "param2";
```

```

private String mParam1;
private String mParam2;
public static final int GAME_ID_LENGTH = 7;

private EditText gameIdTxt;
private Button joinGameBtn;

public JoinGameFragment() {
    // Required empty public constructor
}

/**
 * Use this factory method to create a new instance of
 * this fragment using the provided parameters.
 *
 * @param param1 Parameter 1.
 * @param param2 Parameter 2.
 * @return A new instance of fragment JoinGameFragment.
 */
public static JoinGameFragment newInstance(String param1, String param2) {
    JoinGameFragment fragment = new JoinGameFragment();
    Bundle args = new Bundle();
    args.putString(ARG_PARAM1, param1);
    args.putString(ARG_PARAM2, param2);
    fragment.setArguments(args);
    return fragment;
}

```

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    if (getArguments() != null) {
        mParam1 = getArguments().getString(ARG_PARAM1);
        mParam2 = getArguments().getString(ARG_PARAM2);
    }
}
```

```
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
    Bundle savedInstanceState) {
    // Inflate the layout for this fragment
    View view = inflater.inflate(R.layout.fragment_join_game, container, false);

    gameIdTxt = view.findViewById(R.id.gameIdTxt);
    joinGameBtn = view.findViewById(R.id.joinGameBtn);
    joinGameBtn.setOnClickListener(this);

    return view;
}
```

```
@Override
public void onClick(View v) {
    switch (v.getId()){
        case R.id.joinGameBtn:
            joinGame();
    }
}
```



```

    }

    private void joinGame(){
        String id = gameIdTxt.getText().toString();
        int intId;

        try{
            intId = Integer.parseInt(id);
            if(id.length() != GAME_ID_LENGTH)
                throw new Exception();
        }
        catch (Exception e){
            Toast.makeText(getContext(), "Invalid ID entered!", Toast.LENGTH_SHORT).show();
            gameIdTxt.setText("");
            return;
        }

        Intent intent = new Intent(getActivity(), GameActivity.class);
        intent.putExtra(GameActivity.IS_NEW_GAME_EXTRA, false);
        intent.putExtra(GameActivity.GAME_ID_EXTRA, intId);

        startActivity(intent);
    }
}

```

:Player.java:

```
package com.example.trivia;
```

```

import java.util.ArrayList;
import java.util.Objects;

public class Player extends User{
    //player is a user while a game
    private int currentQuestionIndex;

    private ArrayList<Boolean> isCorrectList; //each cell represents the question in the same
    index

    public Player(String username, String uid, int score, int totalCorrect, int totalWrong) {
        super(username, uid, score, totalCorrect, totalWrong);
        currentQuestionIndex = 0;
        isCorrectList = new ArrayList<>();
    }

    public Player(User user) {
        super(user);
        currentQuestionIndex = 0;
        isCorrectList = new ArrayList<>();
    }

    public Player(Player player) {
        super(player);
        currentQuestionIndex = player.currentQuestionIndex;
        isCorrectList = player.isCorrectList;
    }

    public Player() {
        super();
        currentQuestionIndex = 0;
    }

```

```

        isCorrectList = new ArrayList<>();
    }

    public int getCurrentQuestionIndex() {
        return currentQuestionIndex;
    }

    public void setCurrentQuestionIndex(int currentQuestionIndex) {
        this.currentQuestionIndex = currentQuestionIndex;
    }

    public ArrayList<Boolean> getIsCorrectList() {
        return isCorrectList;
    }

    public void setIsCorrectList(ArrayList<Boolean> isCorrectList) {
        this.isCorrectList = isCorrectList;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Player player = (Player) o;
        return currentQuestionIndex == player.currentQuestionIndex &&
            Objects.equals(isCorrectList, player.isCorrectList);
    }

    @Override

```

```

public int hashCode() {
    return Objects.hash(currentQuestionIndex, isCorrectList);
}

public int calculatePoints() {
    //(totalCorrect * 20 / (totalWrong + 1)) * 3, with round to 10
    return 10 * (int)(5 * (Math.pow(getTotalCorrectInGame(), 1.2)) /
(getTotalWrongInGame() + 1));
}

public int getTotalCorrectInGame(){
    int count = 0;
    for(boolean isCorrect : isCorrectList)
        if(isCorrect)
            count++;

    return count;
}

public int getTotalWrongInGame(){
    int count = 0;
    for(boolean isCorrect : isCorrectList)
        if(!isCorrect)
            count++;

    return count;
}
}

```

```
:LoginActivity.java:
```

```
package com.example.trivia;
```

```
import androidx.annotation.NonNull;
```

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

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

```
import android.content.SharedPreferences;
```

```
import android.os.Bundle;
```

```
import android.view.View;
```

```
import android.widget.Button;
```

```
import android.widget.CheckBox;
```

```
import android.widget.EditText;
```

```
import android.widget.TextView;
```

```
import com.google.android.gms.tasks.OnCompleteListener;
```

```
import com.google.android.gms.tasks.Task;
```

```
import com.google.firebase.auth.AuthResult;
```

```
import com.google.firebase.auth.FirebaseAuth;
```

```
import com.google.firebase.firestore.FirebaseFirestore;
```

```
enum Mode {
```

```
    LOGIN,
```

```
    SIGNUP
```

```
}
```

```
public class LoginActivity extends AppCompatActivity implements View.OnClickListener {
```

```

private static final int MIN_PASSWORD_LENGTH = 6;
private static final String LOGIN_PREFERENCES_FILE = "loginSp";
private static final String LOGIN_PREFERENCES_USERNAME = "username";
private static final String LOGIN_PREFERENCES_PASSWORD = "password";

private Button loginButton;
private TextView toggleLoginModeLbl;
private TextView toggleLoginModeLink;
private EditText usernameTxt;

private EditText passwordTxt;
private EditText passwordAgainTxt;
private CheckBox rememberMeCb;
private TextView loginStatusLbl;

private Mode mode;

private void switchToLoginMode() {
    loginButton.setText("Log In");
    toggleLoginModeLbl.setText("Don't have a user?");
    toggleLoginModeLink.setText("SIGN UP");
    passwordAgainTxt.setVisibility(View.GONE);
    mode = Mode.LOGIN;
}

private void switchToSignupMode() {
    loginButton.setText("Sign Up");
    toggleLoginModeLbl.setText("Already have a user?");

```

```
toggleLoginModeLink.setText("LOG IN");
passwordAgainTxt.setVisibility(View.VISIBLE);
mode = Mode.SIGNUP;
}
```

```
private void login(String username, String password) {
    if (!validateUsernameAnsPassword(username, password))
        return;

    FirebaseAuth.getInstance().signInWithEmailAndPassword(username,
password).addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
        @Override
        public void onComplete(@NonNull Task<AuthResult> task) {
            if (task.isSuccessful()) {
                startMainMenuActivity();

                //TODO: if user isn't listed in "users" list, then create user and add it
                //can happen if user was signed in and had an connection error before he was
added to list
            } else
                loginStatusLbl.setText(task.getException().getMessage());
        }
    });
}
```

```
private void signup(String username, String password, String passwordAgain) {
    //TODO: sign in with user instead of mail
    if (!password.equals(passwordAgain)) {
        loginStatusLbl.setText("Passwords doesn't match");
        passwordTxt.setText("");
        passwordAgainTxt.setText("");
    }
}
```

```

setWrongColors(passwordTxt);
setWrongColors(passwordAgainTxt);

validateUsernameAnsPassword(username, password);
return;
}

```

```

if (!validateUsernameAnsPassword(username, password))
    return;

```

```
String email = username;
```

```

FirebaseAuth.getInstance().createUserWithEmailAndPassword(email,
password).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
    @Override
    public void onComplete(@NonNull Task<AuthResult> task) {
        if (task.isSuccessful()) {
            //add user to users list
            User user = new User(username, FirebaseAuth.getInstance().getUid(), 0, 0, 0);
            FirebaseFirestore.getInstance().

```

```

collection(GameActivity.USERS_COLLECTION_PATH).document(email).set(user).
    addOnCompleteListener(new OnCompleteListener<Void>() {
        @Override
        public void onComplete(@NonNull Task<Void> task) {
            if (task.isSuccessful())
                startMainMenuActivity();
            else
                loginStatusLbl.setText(task.getException().getMessage());

```



```

        }
    });
} else
    loginStatusLbl.setText(task.getException().getMessage());
}
});
}

```

```

private boolean validateUsernameAnsPassword(String username, String password) {
    boolean retValue = true;

    if (username.length() == 0) {
        loginStatusLbl.setText(loginStatusLbl.getText() + "\nPlease enter username");
        setWrongColors(usernameTxt);
        retValue = false;
    }

    if (password.length() < MIN_PASSWORD_LENGTH) {
        loginStatusLbl.setText(loginStatusLbl.getText() + "\nPassword must be at least 6
characters");
        setWrongColors(passwordTxt);
        setWrongColors(passwordAgainTxt);
        retValue = false;
    }

    return retValue;
}

@Override
public void onClick(View v) {

```

```
//reset "wrong" colors
resetColors(usernameTxt);
resetColors(passwordTxt);
resetColors(passwordAgainTxt);
loginStatusLbl.setText("");

switch (v.getId()) {
    case R.id.loginBtn:
        //if required, save username and password to shared preferences
        saveLoginDataToSharedPreferences();

        if (mode == Mode.LOGIN)
            login(usernameTxt.getText().toString(), passwordTxt.getText().toString());
        else if (mode == Mode.SIGNUP)
            signup(usernameTxt.getText().toString(), passwordTxt.getText().toString(),
passwordAgainTxt.getText().toString());
        break;

    case R.id.toggleLoginModeLink:
        if (mode == Mode.LOGIN)
            switchToSignupMode();
        else if (mode == Mode.SIGNUP)
            switchToLoginMode();
        break;
}
}

private void saveLoginDataToSharedPreferences() {
```

```

    SharedPreferences sharedPreferences =
getSharedPreferences(LOGIN_PREFERENCES_FILE, MODE_PRIVATE);

    if (rememberMeCb.isChecked()) {
        //save to shared preferences
        SharedPreferences.Editor sharedPreferencesEditor = sharedPreferences.edit();

        sharedPreferencesEditor.putString(LOGIN_PREFERENCES_USERNAME,
usernameTxt.getText().toString());

        sharedPreferencesEditor.putString(LOGIN_PREFERENCES_PASSWORD,
passwordTxt.getText().toString());

        //apply changes
        sharedPreferencesEditor.apply();
    } else if (sharedPreferences.getAll().size() != 0) { //there is already shared preferences
file
        //delete shared preferences
        sharedPreferences.edit().clear().apply();
    }
}

private void loadLoginDataFromSharedPreferences() {
    //try load the data
    SharedPreferences sharedPreferences =
getSharedPreferences(LOGIN_PREFERENCES_FILE, MODE_PRIVATE);

    String username = sharedPreferences.getString(LOGIN_PREFERENCES_USERNAME,
null);

    String password = sharedPreferences.getString(LOGIN_PREFERENCES_PASSWORD, null);

    //if shared preferences was found, show saved username and password in EditTexts

```

```

    if (username != null)
        usernameTxt.setText(username);
    if (password != null)
        passwordTxt.setText(password);
}

private void setWrongColors(EditText editText) {
    editText.setHintTextColor(MyColor.WRONG_HINT_COLOR);
    editText.setBackgroundColor(MyColor.RED_100);
}

private void resetColors(EditText editText) {
    editText.setHintTextColor(MyColor.DEFAULT_HINT_COLOR);
    editText.setBackgroundColor(0); //transparent
}

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

    loginButton = findViewById(R.id.loginBtn);
    toggleLoginModeLbl = findViewById(R.id.toggleLoginModeLbl);
    toggleLoginModeLink = findViewById(R.id.toggleLoginModeLink);
    usernameTxt = findViewById(R.id.loginUsernameTxt);
    passwordTxt = findViewById(R.id.loginPasswordTxt);
    passwordAgainTxt = findViewById(R.id.loginPasswordAgainTxt);
    rememberMeCb = findViewById(R.id.loginRememberMeCb);

```

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

```
toggleLoginModeLink.setOnClickListener(this);
```

```
loginButton.setOnClickListener(this);
```

```
if (FirebaseAuth.getInstance().getCurrentUser() != null)
```

```
    //FirebaseAuth.getInstance().signOut();
```

```
    startMainMenuActivity();
```

```
else
```

```
    loadLoginDataFromSharedPreferences();
```

```
mode = Mode.LOGIN;
```

```
switchToLoginMode();
```

```
}
```

```
private void startMainMenuActivity() {
```

```
    Intent intent = new Intent(this, MainMenuActivity.class);
```

```
    startActivity(intent);
```

```
    finish();
```

```
}
```

```
}
```

```
:LoadingFragment.java:
```

```
package com.example.trivia;
```

```

import android.os.Bundle;

import androidx.fragment.app.Fragment;

import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

/**
 * A simple {@link Fragment} subclass.
 * Use the {@link LoadingFragment#newInstance} factory method to
 * create an instance of this fragment.
 */
public class LoadingFragment extends Fragment {

    // the fragment initialization parameters, e.g. ARG_ITEM_NUMBER
    private static final String ARG_PARAM1 = "param1";
    private static final String ARG_PARAM2 = "param2";

    private String mParam1;
    private String mParam2;

    public LoadingFragment() {
        // Required empty public constructor
    }

    /**
     * Use this factory method to create a new instance of

```

```

* this fragment using the provided parameters.
*
* @param param1 Parameter 1.
* @param param2 Parameter 2.
* @return A new instance of fragment LoadGameFragment.
*/
public static LoadingFragment newInstance(String param1, String param2) {
    LoadingFragment fragment = new LoadingFragment();
    Bundle args = new Bundle();
    args.putString(ARG_PARAM1, param1);
    args.putString(ARG_PARAM2, param2);
    fragment.setArguments(args);
    return fragment;
}

@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    if (getArguments() != null) {
        mParam1 = getArguments().getString(ARG_PARAM1);
        mParam2 = getArguments().getString(ARG_PARAM2);
    }
}

@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
    Bundle savedInstanceState) {
    // Inflate the layout for this fragment

```

```
        return inflater.inflate(R.layout.fragment_loading, container, false);
    }
}
```

:NewGameFragment.java:

```
package com.example.trivia;
```

```
import static com.example.trivia.Category.ALL;
import static com.example.trivia.Category.COMPUTER_SCIENCE;
import static com.example.trivia.Category.GENERAL_KNOWLEDGE;
import static com.example.trivia.Category.SCIENCE;
import static com.example.trivia.DifficultyLevel.EASY;
import static com.example.trivia.DifficultyLevel.HARD;
import static com.example.trivia.DifficultyLevel.MEDIUM;
import static com.example.trivia.GameActivity.*;
```

```
import android.content.Intent;
import android.os.Bundle;
```

```
import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;
```

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



```
import android.widget.Toast;
```

```
import com.google.android.gms.tasks.OnCompleteListener;
```

```
import com.google.android.gms.tasks.OnFailureListener;
```

```
import com.google.android.gms.tasks.OnSuccessListener;
```

```
import com.google.android.gms.tasks.Task;
```

```
import com.google.android.gms.tasks.Tasks;
```

```
import com.google.api.core.ApiFuture;
```

```
import com.google.firebase.firestore.DocumentReference;
```

```
import com.google.firebase.firestore.DocumentSnapshot;
```

```
import com.google.firebase.firestore.FirebaseFirestore;
```

```
import com.google.firebase.firestore.QuerySnapshot;
```

```
import com.google.firebase.firestore.core.FirestoreClient;
```

```
import java.util.ArrayList;
```

```
import java.util.List;
```

```
enum DifficultyLevel{
```

```
    EASY,
```

```
    MEDIUM,
```

```
    HARD
```

```
}
```

```
enum Category{
```

```
    ALL,
```

```
    GENERAL_KNOWLEDGE,
```

```
    SCIENCE,
```

```
    COMPUTER_SCIENCE
```

```
}
```

```
public class NewGameFragment extends Fragment
    implements View.OnClickListener {
    private static final float UNSELECTED_BUTTON_ALPHA = (float) 0.4;
    private static final int MIN_QUESTION_COUNT = 2;
    private static final int MAX_QUESTION_COUNT = 15;

    private Button allCategoriesBtn, generalKnowledgeCategoryBtn,
        scienceCategoryBtn, computerScienceCategoryBtn;
    private Category category;
    private Button easyDifficultyLevelBtn, mediumDifficultyLevelBtn, hardDifficultyLevelBtn;
    private DifficultyLevel difficultyLevel;
    private Button questionCountDecBtn, questionCountIncBtn;
    private TextView questionCountValueLbl;
    private int questionCount;
    private Button playBtn;

    public NewGameFragment() {
        // Required empty public constructor
    }

    /**
     * Use this factory method to create a new instance of
     * this fragment using the provided parameters.
     *
     * @param param1 Parameter 1.
```

```

private static final String

* @param param2 Parameter 2.
* @return A new instance of fragment StartGameFragment.
*/

public static NewGameFragment newInstance(String param1, String param2) {
    NewGameFragment fragment = new NewGameFragment();
    Bundle args = new Bundle();
    fragment.setArguments(args);
    return fragment;
}

@Override

public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
}

void setCategory(Category newCategory){
    int selectedColor = R.color.purple_500;
    int unselectedColor = R.color.purple_200;

    allCategoriesBtn.setAlpha(UNSELECTED_BUTTON_ALPHA);
    generalKnowledgeCategoryBtn.setAlpha(UNSELECTED_BUTTON_ALPHA);
    scienceCategoryBtn.setAlpha(UNSELECTED_BUTTON_ALPHA);
    computerScienceCategoryBtn.setAlpha(UNSELECTED_BUTTON_ALPHA);

    switch (newCategory){
        case ALL:
            allCategoriesBtn.setAlpha(1);

```

```

        category = ALL;
        break;
    case GENERAL_KNOWLEDGE:
        generalKnowledgeCategoryBtn.setAlpha(1);
        category = GENERAL_KNOWLEDGE;
        break;
    case SCIENCE:
        scienceCategoryBtn.setAlpha(1);
        category = SCIENCE;
        break;
    case COMPUTER_SCIENCE:
        computerScienceCategoryBtn.setAlpha(1);
        category = COMPUTER_SCIENCE;
        break;
    }
}

void setDifficultyLevel(DifficultyLevel newDifficultyLevel){
    easyDifficultyLevelBtn.setAlpha(UNSELECTED_BUTTON_ALPHA);
    mediumDifficultyLevelBtn.setAlpha(UNSELECTED_BUTTON_ALPHA);
    hardDifficultyLevelBtn.setAlpha(UNSELECTED_BUTTON_ALPHA);

    difficultyLevel = newDifficultyLevel;

    switch (newDifficultyLevel){
        case EASY:
            easyDifficultyLevelBtn.setAlpha(1);
            break;
        case MEDIUM:

```

```

        mediumDifficultyLevelBtn.setAlpha(1);

        break;

    case HARD:

        hardDifficultyLevelBtn.setAlpha(1);

        break;

    }

}

void questionCountDec(){
    questionCount--;

    questionCountValueLbl.setText(Integer.toString(questionCount));

    if(questionCount == MIN_QUESTION_COUNT)
        questionCountDecBtn.setEnabled(false);

    questionCountIncBtn.setEnabled(true);
}

void questionCountInc(){
    questionCount++;

    questionCountValueLbl.setText(Integer.toString(questionCount));

    if(questionCount == MAX_QUESTION_COUNT)
        questionCountIncBtn.setEnabled(false);

    questionCountDecBtn.setEnabled(true);
}

private void initView(View v){
    allCategoriesBtn = v.findViewById(R.id.allCategoriesBtn);

```

```
generalKnowledgeCategoryBtn = v.findViewById(R.id.generalKnowledgeCategoryBtn);
scienceCategoryBtn = v.findViewById(R.id.scienceCategoryBtn);
computerScienceCategoryBtn = v.findViewById(R.id.computerScienceCategoryBtn);;
```

```
easyDifficultyLevelBtn = v.findViewById(R.id.easyDifficultyLevelBtn);
mediumDifficultyLevelBtn = v.findViewById(R.id.mediumDifficultyLevelBtn);
hardDifficultyLevelBtn = v.findViewById(R.id.hardDifficultyLevelBtn);
```

```
questionCountDecBtn = v.findViewById(R.id.questionCountDecBtn);
questionCountIncBtn = v.findViewById(R.id.questionCountIncBtn);
questionCountValueLbl = v.findViewById(R.id.questionCountValueLbl);
```

```
playBtn = v.findViewById(R.id.playBtn);
```

```
allCategoriesBtn.setOnClickListener(this);
generalKnowledgeCategoryBtn.setOnClickListener(this);
scienceCategoryBtn.setOnClickListener(this);
computerScienceCategoryBtn.setOnClickListener(this);
setCategory(ALL);
```

```
easyDifficultyLevelBtn.setOnClickListener(this);
mediumDifficultyLevelBtn.setOnClickListener(this);
hardDifficultyLevelBtn.setOnClickListener(this);
setDifficultyLevel(EASY);
```

```
questionCountDecBtn.setOnClickListener(this);
questionCountIncBtn.setOnClickListener(this);
```

```

questionCount = 10;
questionCountValueLbl.setText(Integer.toString(questionCount));

playBtn.setOnClickListener(this);
}

@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
        Bundle savedInstanceState) {
    // Inflate the layout for this fragment
    View view = inflater.inflate(R.layout.fragment_new_game, container, false);
    initView(view);

    return view;
}

@Override
public void onClick(View v) {
    switch (v.getId()){
        case R.id.easyDifficultyLevelBtn:
            setDifficultyLevel(EASY);
            break;
        case R.id.mediumDifficultyLevelBtn:
            setDifficultyLevel(MEDIUM);
            break;
        case R.id.hardDifficultyLevelBtn:
            setDifficultyLevel(HARD);
            break;
    }
}

```

```

case R.id.allCategoriesBtn:
    setCategory(ALL);
    break;

case R.id.generalKnowledgeCategoryBtn:
    setCategory(GENERAL_KNOWLEDGE);
    break;

case R.id.scienceCategoryBtn:
    setCategory(SCIENCE);
    break;

case R.id.computerScienceCategoryBtn:
    setCategory(COMPUTER_SCIENCE);
    break;


case R.id.questionCountDecBtn:
    questionCountDec();
    break;

case R.id.questionCountIncBtn:
    questionCountInc();
    break;


case R.id.playBtn:
    Intent intent = new Intent(getActivity(), GameActivity.class);
    int extras[] = new int[3];
    extras[QUESTIONS_COUNT_INDEX] = questionCount;
    extras[DIFFICULTY_LEVEL_INDEX] = difficultyLevel.ordinal();
    extras[CATEGORY_INDEX] = category.ordinal();
    intent.putExtra(GameActivity.NEW_GAME_EXTRAS, extras);

```



```

        intent.putExtra(IS_NEW_GAME_EXTRA, true);

        startActivity(intent);
        getActivity().finish();
    }
}
}

```

:Question.java:

```
package com.example.trivia;
```

```
import java.util.ArrayList;
```

```
public class Question {
```

```
    String question;
```

```
    ArrayList<String> answers;
```

```
    int correctAnswer;
```

```
    Category category;
```

```
    DifficultyLevel difficultyLevel;
```

```
    public String getQuestion() {
```

```
        return question;
```

```
    }
```

```
    public void setQuestion(String question) {
```

```
        this.question = question;
```

```
    }
```

```

public Category getCategory() {
    return category;
}

public ArrayList<String> getAnswers() {
    return answers;
}

public void setAnswers(ArrayList<String> answers) {
    this.answers = answers;
}

public void setCategory(Category category) {
    this.category = category;
}

public DifficultyLevel getDifficultyLevel() {
    return difficultyLevel;
}

public void setDifficultyLevel(DifficultyLevel difficultyLevel) {
    this.difficultyLevel = difficultyLevel;
}

public int getCorrectAnswer() {
    return correctAnswer;
}

```

```
public void setCorrectAnswer(int correctAnswer) {
    this.correctAnswer = correctAnswer;
}
}
```

:MusicService.java:

```
package com.example.trivia;
```

```
import android.app.Notification;
import android.app.NotificationChannel;
import android.app.NotificationManager;
import android.app.Service;
import android.content.Context;
import android.content.Intent;
import android.database.Observable;
import android.media.MediaPlayer;
import android.os.Build;
import android.os.ConditionVariable;
import android.os.IBinder;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.core.app.NotificationCompat;
import androidx.lifecycle.LifecycleOwner;
import androidx.lifecycle.LiveData;
```

```

import androidx.lifecycle.MutableLiveData;
import androidx.lifecycle.Observer;

public class MusicService extends Service {

    private static final int ONGOING_NOTIFICATION_ID = 4242;
    private static final String MUSIC_NOTIFICATION_CHANNEL_ID = "MUSIC";
    private MediaPlayer mediaPlayer;

    @Nullable
    @Override
    public IBinder onBind(Intent intent) {
        return null;
    }

    @Override
    public void onCreate() {
        super.onCreate();

        //create player
        mediaPlayer = MediaPlayer.create(this, R.raw.bg_music);

        //play forever
        mediaPlayer.setLooping(true);

        //listen to volume changes
        SettingsFragment.backgroundMusicVolume = new MutableLiveData<>();
        SettingsFragment.backgroundMusicVolume.observeForever(new Observer<Float>() {
            @Override

```

```
public void onChanged(Float aFloat) {
    mediaPlayer.setVolume(aFloat, aFloat);
}
});
```

```
SettingsFragment.backgroundMusicVolume.setValue(1f);
}
```

```
@Override
```

```
public int onStartCommand(Intent intent, int flags, int startId) {
    //start playing
    mediaPlayer.start();
}
```

```
NotificationManager notificationManager = (NotificationManager)
getSystemService(NOTIFICATION_SERVICE);
```

```
//if SDK version is over 26, a notification channel is required
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
    CharSequence name = "BackgroundMusic";

    NotificationChannel channel = new
NotificationChannel(MUSIC_NOTIFICATION_CHANNEL_ID, name,
NotificationManager.IMPORTANCE_DEFAULT);

    channel.setDescription("Is playing music");
    notificationManager.createNotificationChannel(channel);
}

String channelId = "";

if (android.os.Build.VERSION.SDK_INT >= android.os.Build.VERSION_CODES.O) {
    channelId = MUSIC_NOTIFICATION_CHANNEL_ID;
}
```

```
//add service notification
Notification notification = new NotificationCompat.Builder(this, channelId)
    .setOngoing(true)
    .setSmallIcon(R.drawable.ic_play)
    .setCategory(NotificationCompat.CATEGORY_SERVICE)
    .build();

//start service
startForeground(ONGOING_NOTIFICATION_ID, notification);

return START_STICKY;
}

@Override
public void onDestroy() {
    super.onDestroy();
    //stop service
    mediaPlayer.stop();
}
}
```

```
:AnswerRecorder.java:
package com.example.trivia;

import android.Manifest;
import android.app.Activity;
```

```

import android.content.Context;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.os.Bundle;
import android.speech.RecognitionListener;
import android.speech.RecognizerIntent;
import android.speech.SpeechRecognizer;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

public class AnswerRecorder {
    private static final String HEBREW = "iw-IL";
    private static SpeechRecognizer speechRecognizer = null;

    public static void startRecording(Context context, Button[] answerButtons) {
        //create speech recognizer
        speechRecognizer = SpeechRecognizer.createSpeechRecognizer(context);

        Intent recognizerIntent = new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
        //set language to Hebrew
        recognizerIntent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,
            RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
        recognizerIntent.putExtra(RecognizerIntent.EXTRA_LANGUAGE, HEBREW);
    }
}

```

```

//ask permission to record if required

if (ContextCompat.checkSelfPermission(context,
android.Manifest.permission.RECORD_AUDIO) != PackageManager.PERMISSION_GRANTED)

    ActivityCompat.requestPermissions((Activity) context, new
String[]{Manifest.permission.RECORD_AUDIO}, 1);


speechRecognizer.setRecognitionListener(new RecognitionListener() {

    //auto generated methods

    @Override

    public void onReadyForSpeech(Bundle params) {

    }

    @Override

    public void onBeginningOfSpeech() {

    }

    @Override

    public void onRmsChanged(float rmsdB) {

    }

    @Override

    public void onBufferReceived(byte[] buffer) {

    }

    @Override

```



```

public void onEndOfSpeech() {

}

@Override
public void onError(int error) {

}

@Override
public void onResults(Bundle results) {
    try {
        //get result from bundle
        String word =
results.getStringArrayList(SpeechRecognizer.RESULTS_RECOGNITION).get(0);

        int intResult = -1;

        //check if result is a number between 1-4
        if (word.equals("one") || word.equals("אחת") || word.equals("1"))
            intResult = 1;
        else if (word.equals("two") || word.equals("שתיים") || word.equals("2"))
            intResult = 2;
        else if (word.equals("three") || word.equals("שלוש") || word.equals("3"))
            intResult = 3;
        else if (word.equals("four") || word.equals("ארבע") || word.equals("4"))
            intResult = 4;

        intResult--;
    }
}

```

```

        //get the button of this answer
        Button answerButton = answerButtons[intResult];
        if (answerButton.isEnabled())
            //click this button
            answerButton.callOnClick();
    } catch (Exception e) {
        //value isn't a number
        //or the number is out of range
        //do nothing
    }

    //after recording, the speech recognizer is not needed
    speechRecognizer.destroy();
}

@Override
public void onPartialResults(Bundle results) {
}

@Override
public void onEvent(int eventType, Bundle params) {
}

});

//try recording
if (SpeechRecognizer.isRecognitionAvailable(context))

```

```

        speechRecognizer.startListening(recognizerIntent);
    else
        Toast.makeText(context, "Recording not available!", Toast.LENGTH_SHORT).show();
    }

    public static void stopRecording() {
        //stop speech recognizer
        speechRecognizer.stopListening();
    }
}

```

:GameIdFragment.java:

```
package com.example.trivia;
```

```
import android.os.Bundle;
```

```
import androidx.fragment.app.Fragment;
```

```
import androidx.lifecycle.ViewModelProvider;
```

```
import android.view.LayoutInflater;
```

```
import android.view.View;
```

```
import android.view.ViewGroup;
```

```
import android.widget.TextView;
```

```
/**
```

```
 * A simple {@link Fragment} subclass.
```

* Use the {@link GameldFragment#newInstance} factory method to

* create an instance of this fragment.

*/

```
public class GameldFragment extends Fragment {
```

```
    // TODO: Rename parameter arguments, choose names that match
```

```
    // the fragment initialization parameters, e.g. ARG_ITEM_NUMBER
```

```
    private static final String ARG_PARAM1 = "param1";
```

```
    private static final String ARG_PARAM2 = "param2";
```

```
    private TextView startGameGameldLbl;
```

```
    // TODO: Rename and change types of parameters
```

```
    private String mParam1;
```

```
    private String mParam2;
```

```
    public GameldFragment() {
```

```
        // Required empty public constructor
```

```
    }
```

```
    /**
```

```
     * Use this factory method to create a new instance of
```

```
     * this fragment using the provided parameters.
```

```
     *
```

```
     * @param param1 Parameter 1.
```

```
     * @param param2 Parameter 2.
```

```
     * @return A new instance of fragment GameldFragment.
```

```
     */
```

```
    // TODO: Rename and change types and number of parameters
```

```
public static GameldFragment newInstance(String param1, String param2) {
    GameldFragment fragment = new GameldFragment();
    Bundle args = new Bundle();
    args.putString(ARG_PARAM1, param1);
    args.putString(ARG_PARAM2, param2);
    fragment.setArguments(args);
    return fragment;
}
```

@Override

```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    if (getArguments() != null) {
        mParam1 = getArguments().getString(ARG_PARAM1);
        mParam2 = getArguments().getString(ARG_PARAM2);
    }
}
```

@Override

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

    // Inflate the layout for this fragment
    View view = inflater.inflate(R.layout.fragment_game_id, container, false);

    startGameGameldLbl = view.findViewById(R.id.startGameGameldLbl);

    GameViewModel gameViewModel = new
    ViewModelProvider(getActivity()).get(GameViewModel.class);

    int gameId = gameViewModel.getGame().getId();
```

```

startGameGameIdLbl.setText(Integer.toString(gameId));

return view;
}
}

```

:GameActivity.java:

```

package com.example.trivia;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;
import androidx.core.content.PackageManagerCompat;
import androidx.fragment.app.Fragment;
import androidx.fragment.app.FragmentManager;
import androidx.fragment.app.FragmentTransaction;
import androidx.lifecycle.Observer;
import androidx.lifecycle.ViewModelProvider;

import android.Manifest;
import android.app.Activity;
import android.content.Context;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.graphics.Bitmap;

```

```
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.os.AsyncTask;
import android.os.Bundle;
import android.os.Handler;
import android.provider.CalendarContract;
import android.speech.RecognitionListener;
import android.speech.RecognizerIntent;
import android.speech.SpeechRecognizer;
import android.view.MotionEvent;
import android.view.View;
import android.widget.Button;
import android.widget.ImageButton;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.firestore.DocumentSnapshot;
import com.google.firebase.firestore.FirebaseFirestore;

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

```

import java.util.Objects;

import java.util.Random;

public class GameActivity extends AppCompatActivity implements View.OnClickListener,
View.OnTouchListener {

    public static final String GAMES_COLLECTION_PATH = "games";
    public static final String USERS_COLLECTION_PATH = "users";

    public static final String NEW_GAME_EXTRAS = "extras";
    public static final String IS_NEW_GAME_EXTRA = "isNewGame";
    public static final String GAME_ID_EXTRA = "gameId";

    public static final int QUESTIONS_COUNT_INDEX = 0;
    public static final int DIFFICULTY_LEVEL_INDEX = 1;
    public static final int CATEGORY_INDEX = 2;

    private TextView currentQuestionLbl;
    private TextView questionLbl;
    private Button[] answerButtons;

    private ImageButton homeImgBtn;
    private ImageView progressImg;
    private Canvas pbCanvas;
    private Bitmap progressBitmap;
    private ImageButton recordImgBtn;

    private GameViewModel gameVM;
    private Fragment loadingFragment;
    private Fragment waitForEnemyFragment;

```



```

private Fragment gameldFragment;

private FirebaseFirestore firestore;

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

    //show loading fragment
    new ShowLoadingFragmentAsync().execute();

    currentQuestionLbl = findViewById(R.id.currentQuestionLbl);
    questionLbl = findViewById(R.id.questionLbl);
    answerButtons = new Button[4];
    answerButtons[0] = findViewById(R.id.answer0Btn);
    answerButtons[1] = findViewById(R.id.answer1Btn);
    answerButtons[2] = findViewById(R.id.answer2Btn);
    answerButtons[3] = findViewById(R.id.answer3Btn);

    homeImgBtn = findViewById(R.id.homeBtn);
    recordImgBtn = findViewById(R.id.recordImgBtn);
    progressImg = findViewById(R.id.progressImg);

    gameVM = new ViewModelProvider(this).get(GameViewModel.class);
    firestore = FirebaseFirestore.getInstance();

    //is creator = is this player the game creator
    gameVM.setCreator(getIntent().getBooleanExtra(IS_NEW_GAME_EXTRA, false));

```

```

homeImgBtn.setOnClickListener(this);

for (Button answer : answerButtons)
    answer.setOnClickListener(this);

recordImgBtn.setOnTouchListener(this);

if (gameVM.getMyPlayer() == null) {
    //init screen
    //get user data
    String email = FirebaseAuth.getInstance().getCurrentUser().getEmail();

    firestore.collection(USERS_COLLECTION_PATH).document(email).get().addOnFailureListener(
        new OnFailureListener() {
            @Override
            public void onFailure(@NonNull Exception e) {
                Toast.makeText(GameActivity.this, "Connection error!",
                    Toast.LENGTH_SHORT).show();//not shown
                backToMainMenu();
            }
        }).addOnSuccessListener(new OnSuccessListener<DocumentSnapshot>() {
            @Override
            public void onSuccess(DocumentSnapshot documentSnapshot) {
                //add user to viewModel
                User user = documentSnapshot.toObject(User.class);
                gameVM.setMyPlayer(new Player(user));

                //create/join game
                //screen not initialized
                if (gameVM.isCreator()) {
                    //create new game

```

```

        int[] extras = getIntent().getIntArrayExtra(NEW_GAME_EXTRAS);

        int questionsCount = extras[QUESTIONS_COUNT_INDEX];

        DifficultyLevel difficultyLevel =
DifficultyLevel.values()[extras[DIFFICULTY_LEVEL_INDEX]];

        Category category = Category.values()[extras[CATEGORY_INDEX]];

        GetQuestionsAsync getQuestionsAsync = new GetQuestionsAsync();
        getQuestionsAsync.execute(questionsCount, difficultyLevel.ordinal(),
category.ordinal());
    } else {
        //not creator
        //join existing game

        int id = getIntent().getIntExtra(GAME_ID_EXTRA, -1);

        JoinGameAsync joinGameAsync = new JoinGameAsync();
        joinGameAsync.execute(id);
    }
}

});

} else {
    //screen already initialized
    showCurrentQuestion();
}
}

@Override
public void onWindowFocusChanged(boolean hasFocus) {
    super.onWindowFocusChanged(hasFocus);

```

```

        initProgressBarCanvas();
    }

    //TODO: make non-static
    public static Fragment showLoadingFragment(FragmentManager fm) {
        FragmentTransaction fragmentTransaction = fm.beginTransaction();
        Fragment loadingFragment = new LoadingFragment();
        fragmentTransaction.replace(R.id.gameLayout, loadingFragment);
        fragmentTransaction.commit();

        return loadingFragment;
    }

    public static void hideLoadingFragment(FragmentManager fm, Fragment
loadGameFragment) {
        FragmentTransaction fragmentTransaction = fm.beginTransaction();
        fragmentTransaction.hide(loadGameFragment);
        fragmentTransaction.commit();
    }

    private void showFragment(Fragment fragment) {
        FragmentTransaction fragmentTransaction =
getSupportFragmentManager().beginTransaction();
        fragmentTransaction.replace(R.id.gameLayout, fragment);
        fragmentTransaction.commit();
    }

    private void hideFragment(Fragment fragment) {

```

```

        FragmentTransaction fragmentTransaction =
getSupportFragmentManager().beginTransaction();

        fragmentTransaction.hide(fragment);

        fragmentTransaction.commit();
    }

```

```

private void showEndGameFragment() {
    homeImgBtn.setEnabled(false);
}

```

```

        FragmentTransaction fragmentTransaction =
getSupportFragmentManager().beginTransaction();

        fragmentTransaction.replace(R.id.gameLayout, new EndGameFragment());

        fragmentTransaction.commit();
    }

```

```

private void waitForEnemy() {
    loadingFragment = showLoadingFragment(getSupportFragmentManager());

    int questionCount = gameVM.getQuestions().size();
    if (gameVM.getOtherPlayer().getCurrentQuestionIndex() == questionCount)
        //enemy finished too
        endGame();
    else {
        waitForEnemyFragment = new WaitToEnemyFragment();
        showFragment(waitForEnemyFragment);

        //wait until enemy ends the game too
        gameVM.getGameLiveData().observe(GameActivity.this, new Observer<Game>() {
            @Override
            public void onChanged(Game game) {

```

```

        if (gameVM.getOtherPlayer().getCurrentQuestionIndex() == questionCount) {
            //enemy finished
            hideFragment(waitForEnemyFragment);
            endGame();
        }

    }

});
}
}

```

```

private void endGame() {
    //TODO: delete game from firestore

    //update user score
    gameVM.getMyPlayer().setTotalCorrect(gameVM.getMyPlayer().getTotalCorrect() +
    gameVM.getMyPlayer().getTotalCorrectInGame());

    gameVM.getMyPlayer().setTotalWrong(gameVM.getMyPlayer().getTotalWrong() +
    gameVM.getMyPlayer().getTotalWrongInGame());

    int myPoints = gameVM.getMyPlayer().calculatePoints();
    int otherPoints = gameVM.getOtherPlayer().calculatePoints();
    if (myPoints > otherPoints)
        //you won, add points to total score
        gameVM.getMyPlayer().setScore(gameVM.getMyPlayer().getScore() + myPoints);

    User myPlayerAsUser = gameVM.getMyPlayer();

    //send new user data to users list

```

```

    firestore.collection(USERS_COLLECTION_PATH).document(gameVM.getMyPlayer().getEmail(
    ))

        .set(myPlayerAsUser).addOnCompleteListener(new OnCompleteListener<Void>() {

            @Override

            public void onComplete(@NonNull Task<Void> task) {

                if (!task.isSuccessful()) {

                    Toast.makeText(GameActivity.this, "Connection error!",
                    Toast.LENGTH_SHORT).show();

                    backToMainMenu();

                } else {

                    //game updated successfully

                    showEndGameFragment();

                }

            }

        });
    }

```

```

private class GetQuestionsAsync extends AsyncTask<Integer, Integer, ArrayList<Question>>
{

    @Override

    //params: question count, difficulty level, category

    protected ArrayList<Question> doInBackground(Integer... integers) {

        HttpQuestionFetcher questionFetcher = new HttpQuestionFetcher();

        return questionFetcher.getQuestions(integers[QUESTIONS_COUNT_INDEX],
        integers[DIFFICULTY_LEVEL_INDEX], integers[CATEGORY_INDEX]);

    }

    @Override

```

```
protected void onPostExecute(ArrayList<Question> questions) {
    if (questions == null || questions.size() == 0) {
        Toast.makeText(getBaseContext(), "Couldn't fetch questions!",
            Toast.LENGTH_SHORT).show();
        backToMainMenu();
    } else {
        gameVM.setQuestions(questions);

        sendGameToFirestore();
    }
}
}
```

```
private class JoinGameAsync extends AsyncTask<Integer, Integer, Void> {
    @Override
    //params: id
    protected Void doInBackground(Integer... integers) {
        Player myPlayer = gameVM.getMyPlayer(); //getting the game from firestore
        overrides player2 to null
        int id = integers[0];
        String strId = Integer.toString(id);

        firestore.collection(GAMES_COLLECTION_PATH).document(strId).get().addOnSuccessListener(
            new OnSuccessListener<DocumentSnapshot>() {
                @Override
                public void onSuccess(DocumentSnapshot documentSnapshot) {
                    //game loaded successfully
                    if (documentSnapshot.exists()) {
                        Game game = documentSnapshot.toObject(Game.class);
```



```

        gameVM.setGame(game);
        gameVM.enableGameSyncWithFirestore(GameActivity.this);
        gameVM.setMyPlayer(myPlayer);

        hideLoadingFragment(getSupportFragmentManager(), loadingFragment);
        showCurrentQuestion();
    } else {
        Toast.makeText(GameActivity.this, "Wrong game ID!",
            Toast.LENGTH_SHORT).show();
        backToMainMenu();
    }
}

}).addOnFailureListener(new OnFailureListener() {
    @Override
    public void onFailure(@NonNull Exception e) {
        Toast.makeText(GameActivity.this, "Connection error!",
            Toast.LENGTH_SHORT).show();
        backToMainMenu();
    }
});

return null;
}
}

private class ShowLoadingFragmentAsync extends AsyncTask<Void, Void, Void> {
    @Override
    protected Void doInBackground(Void... voids) {

```

```

        loadingFragment = showLoadingFragment(getSupportFragmentManager());
        return null;
    }
}

private void sendGameToFirestore() {
    //random integer between 100,000-999,999
    int minId = (int) Math.pow(10, JoinGameFragment.GAME_ID_LENGTH - 1);
    int maxId = (int) Math.pow(10, JoinGameFragment.GAME_ID_LENGTH);
    int gameId = minId + (new Random()).nextInt(maxId - minId);

    gameVM.setGameId(gameId);

    //add game to DB, and wait for an enemy
    //TODO: validate ID not duplicated
    //TODO: change loading label

    firestore.collection(GAMES_COLLECTION_PATH).document(Integer.toString(gameId))
        .set(gameVM.getGame()).addOnFailureListener(new OnFailureListener() {
            @Override
            public void onFailure(@NonNull Exception e) {
                Toast.makeText(getBaseContext(), "Failed to create game!",
                    Toast.LENGTH_SHORT).show();
                backToMainMenu();
            }
        }).addOnSuccessListener(new OnSuccessListener<Void>() {
            @Override
            public void onSuccess(Void unused) {
                hideLoadingFragment(getSupportFragmentManager(), loadingFragment);
            }
        });
}

```

```

        gameIdFragment = new GameIdFragment();
        showFragment(gameIdFragment);
        gameVM.enableGameSyncWithFirestore(GameActivity.this);

        //wait until a an enemy joins
        gameVM.getGameLiveData().observe(GameActivity.this, new
Observer<Game>() {
            @Override
            public void onChanged(Game game) {
                if (game.getPlayer2() != null) {
                    //enemy joined
                    hideFragment(gameIdFragment);
                    showCurrentQuestion();
                }
            }
        });
    }
});
}

public void backToMainMenu() {
    Intent intent = new Intent(this, MainMenuActivity.class);
    startActivity(intent);
    finish();
}

private void showCurrentQuestion() {
    int currentQuestionIndex = 0;
    Player myPlayer = gameVM.getMyPlayer();

```

```

if (myPlayer != null)
    currentQuestionIndex = myPlayer.getCurrentQuestionIndex();
int totalQuestions = gameVM.getQuestions().size();

if (currentQuestionIndex >= totalQuestions)
    //no more questions
    return;

Question question = gameVM.getQuestions().get(currentQuestionIndex);

currentQuestionLbl.setText((currentQuestionIndex + 1) +
    "/" +
    gameVM.getQuestions().size());

questionLbl.setText(question.getQuestion());
for (int i = 0; i < answerButtons.length; i++)
    answerButtons[i].setText(question.getAnswers().get(i));

int color = getResources().getColor(R.color.main_color_500);
for (Button answer : answerButtons)
    answer.setBackgroundColor(getResources().getColor(R.color.main_color_500));

for (Button answerBtn : answerButtons)
    answerBtn.setEnabled(true);
recordImgBtn.setEnabled(true);
}

private void sendAnswer(int answerIndex, Button answerButton) {

```

```

int currentQuestionIndex = gameVM.getPlayer().getCurrentQuestionIndex();

boolean isCorrect = gameVM.getQuestions().get(currentQuestionIndex).correctAnswer
== answerIndex;

ArrayList<Boolean> isCorrectList = gameVM.getPlayer().getIsCorrectList();
isCorrectList.add(isCorrect);
gameVM.setMyIsCorrectList(isCorrectList);

if (isCorrect) {
    answerButton.setBackgroundColor(MyColor.CORRECT_GREEN);
} else {
    answerButton.setBackgroundColor(MyColor.WRONG_RED);

answerButtons[gameVM.getQuestions().get(currentQuestionIndex).correctAnswer].setBack
groundColor(MyColor.CORRECT_GREEN);
}

drawIsCorrectOnProgressBar(isCorrect, currentQuestionIndex);

gameVM.setCurrentQuestionIndex(currentQuestionIndex + 1);

for (Button answerBtn : answerButtons)
    answerBtn.setEnabled(false);
recordImgBtn.setEnabled(false);

new Handler().postDelayed(new Runnable() {
    @Override
    public void run() {
        if (currentQuestionIndex == gameVM.getQuestions().size() - 1) {

```

```

        //game ended
        waitForEnemy(); //function waitForEnemy also ends the game

    } else {
        showCurrentQuestion();
    }
}
}, 1000);
}

```

```

private void initProgressBarCanvas() {
    //draw question progress bar
    Paint paint = new Paint();
    paint.setColor(0xAACCCCCC); //light gray

    progressBitmap = Bitmap.createBitmap(progressImg.getWidth(),
    progressImg.getHeight(), Bitmap.Config.ARGB_8888);
    pbCanvas = new Canvas(progressBitmap);

    pbCanvas.drawRect(0, 0, pbCanvas.getWidth(), pbCanvas.getHeight(), paint);
    progressImg.setImageBitmap(progressBitmap);

    Player myPlayer = gameVM.getMyPlayer();
    if (myPlayer != null) {
        ArrayList<Boolean> isCorrectList = gameVM.getMyPlayer().getIsCorrectList();
        for (int i = 0; i < isCorrectList.size(); i++)
            drawIsCorrectOnProgressBar(isCorrectList.get(i), i);
    }
}

```

```
}
```

```
private void drawIsCorrectOnProgressBar(boolean isCorrect, int currentQuestion) {
    int totalQuestions = gameVM.getQuestions().size();

    Paint paint = new Paint();
    if (isCorrect)
        paint.setColor(MyColor.CORRECT_GREEN);
    else
        paint.setColor(MyColor.WRONG_RED);

    int start = currentQuestion * (pbCanvas.getWidth() / totalQuestions);
    int end = (currentQuestion + 1) * (pbCanvas.getWidth() / totalQuestions);
    pbCanvas.drawRect(start, 0, end, pbCanvas.getHeight(), paint);

    progressImg.setImageBitmap(progressBitmap);
}
```

```
@Override
public void onClick(View v) {
    for (int i = 0; i < answerButtons.length; i++)
        if (v.getId() == answerButtons[i].getId())
            sendAnswer(i, (Button) v);
}
```

```
@Override
public boolean onTouch(View v, MotionEvent event) {
    if (v.getId() == R.id.recordImgBtn) {
```

```

        if (event.getAction() == MotionEvent.ACTION_DOWN) {
            AnswerRecorder.startRecording(this, answerButtons);
        } else if (event.getAction() == MotionEvent.ACTION_UP) {
            AnswerRecorder.stopRecording();
        }
    }

    return true;
}
}

```

:GameViewModel.java:

```
package com.example.trivia;
```

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

```
import android.widget.Toast;
```

```
import androidx.annotation.NonNull;
```

```
import androidx.annotation.Nullable;
```

```
import androidx.lifecycle.LifecycleOwner;
```

```
import androidx.lifecycle.MutableLiveData;
```

```
import androidx.lifecycle.Observer;
```

```
import androidx.lifecycle.ViewModel;
```

```
import com.google.android.gms.tasks.OnFailureListener;
```

```
import com.google.firebase.firestore.DocumentSnapshot;
```



```
import com.google.firebase.firestore.EventListener;
import com.google.firebase.firestore.FirebaseFirestore;
import com.google.firebase.firestore.FirebaseFirestoreException;

import java.util.ArrayList;

public class GameViewModel extends ViewModel {
    private MutableLiveData<Game> game;
    private boolean isCreator; //is this player the game creator?

    public GameViewModel() {
        game = new MutableLiveData<>();
        game.setValue(new Game());
        isCreator = true;
    }

    public Game getGame() {
        return game.getValue();
    }

    public void setGame(Game game) {
        this.game.setValue(game);
    }

    public boolean isCreator() {
        return isCreator;
    }

    public void setCreator(boolean creator) {
```

```

        isCreator = creator;
    }

    public void setQuestions(ArrayList<Question> questions) {
        getGame().setQuestions(questions);
    }

    public Player getPlayer1(){
        return getGame().getPlayer1();
    }

    public Player getPlayer2(){
        return getGame().getPlayer2();
    }

    public void setPlayer1(Player player1){
        getGame().setPlayer1(player1);
    }

    public void setPlayer2(Player player2){
        getGame().setPlayer2(player2);
    }

    public ArrayList<Question> getQuestions() {
        return getGame().getQuestions();
    }

    public Player getMyPlayer() {
        if(isCreator)
            //creator is player1

```

```

        return getGame().getPlayer1();
    else
        return getGame().getPlayer2();
}

```

```

public void setMyPlayer(Player player) {
    //creator is player1
    Game newGame = getGame();
    if(isCreator)
        newGame.setPlayer1(player);
    else
        newGame.setPlayer2(player);

    game.setValue(newGame);
}

```

```

//save the state of current player
//used to know whether it is changed in game's observer
private Player previousMyPlayer;
public void enableGameSyncWithFirestore(Context context){
    previousMyPlayer = getMyPlayer();
    String gameId = Integer.toString(getGame().getId());

    //when other player is changed, update Game locally
    FirebaseFirestore firestore = FirebaseFirestore.getInstance();

```

```

    firestore.collection(GameActivity.GAMES_COLLECTION_PATH).document(gameId).addSnapshotListener(new EventListener<DocumentSnapshot>() {
        @Override

```

```

public void onEvent(@Nullable DocumentSnapshot snapshot, @Nullable
FirebaseFirestoreException error) {

    if(error == null && snapshot != null && snapshot.exists()){

        //no exception

        Game newGame = snapshot.toObject(Game.class);

        if(newGame.getPlayer2() != null){

            //if player2 is null, game hasn't yet started

            Game gameValue = getGame();

            if(isCreator && !newGame.getPlayer2().equals(gameValue.getPlayer2())){

                //player 2 has changed

                gameValue.setPlayer2(newGame.getPlayer2());

                game.setValue(gameValue);

            }

            if(!isCreator && !newGame.getPlayer1().equals(gameValue.getPlayer1())){

                //player 1 has changed

                gameValue.setPlayer1(newGame.getPlayer1());

                game.setValue(gameValue);

            }

        }

    }

}

});

```

//when my player changed, update in firestore

```

game.observe((LifecycleOwner) context, new Observer<Game>() {

    @Override

    public void onChanged(Game newGame) {

        if((previousMyPlayer == null && getMyPlayer() != null) ||

```

```
(previousMyPlayer != null && !previousMyPlayer.equals(getMyPlayer()))
```

```
//player has changed
```

```
firestore.collection(GameActivity.GAMES_COLLECTION_PATH).document(gameId).set(newGame);
```

```
addOnFailureListener(new OnFailureListener() {
```

```
@Override
```

```
public void onFailure(@NonNull Exception e) {
```

```
    Toast.makeText(context, "Connection error!",  
    Toast.LENGTH_SHORT).show();
```

```
    //TODO: exit screen
```

```
}
```

```
});
```

```
if(getMyPlayer() == null)
```

```
    //copy constructor doesn't work with null
```

```
    previousMyPlayer = null;
```

```
else
```

```
    previousMyPlayer = new Player(getMyPlayer());
```

```
}
```

```
});
```

```
}
```

```
public MutableLiveData<Game> getGameLiveData() {
```

```
    return game;
```

```
}
```

```
public void setMyCurrentQuestionIndex(int i) {
```

```
//creator is player1
Game newGame = getGame();
if(isCreator)
    newGame.getPlayer1().setCurrentQuestionIndex(i);
else
    newGame.getPlayer2().setCurrentQuestionIndex(i);

game.setValue(newGame);
}

public void setMyIsCorrectList(ArrayList<Boolean> isCorrectList) {
    //creator is player1
    Game newGame = getGame();
    if(isCreator)
        newGame.getPlayer1().setIsCorrectList(isCorrectList);
    else
        newGame.getPlayer2().setIsCorrectList(isCorrectList);

    game.setValue(newGame);
}

public void setGameId(int gameId) {
    Game newGame = getGame();
    newGame.setId(gameId);
    game.setValue(newGame);
}

public Player getOtherPlayer() {
```

```

        if(isCreator)
            return getGame().getPlayer2();
        else
            return getGame().getPlayer1();
    }
}

```

:MyColor.java:

```
package com.example.trivia;
```

```

public class MyColor {
    static final int CORRECT_GREEN = 0xFF0BB90F;
    static final int WRONG_RED = 0xFFE70000;
    static final int RED_100 = 0x10FF0000;
    static final int WRONG_HINT_COLOR = 0x66d40f0f;
    static final int DEFAULT_HINT_COLOR = 0x61000000;
}

```

:Game.java:

```
package com.example.trivia;
```

```

import java.util.ArrayList;

public class Game {
    private ArrayList<Question> questions;
}

```

//player1 is the player who created the game, player2 is the other one.

```
private Player player1, player2;
```

```
private int id;
```

```
public Game() {
```

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

```
    player1 = null;
```

```
    player2 = null;
```

```
    id = -1;
```

```
}
```

```
public ArrayList<Question> getQuestions() {
```

```
    return questions;
```

```
}
```

```
public int getId() {
```

```
    return id;
```

```
}
```

```
public void setId(int id) {
```

```
    this.id = id;
```

```
}
```

```
public void setQuestions(ArrayList<Question> questions) {
```

```
    this.questions = questions;
```

```
}
```

```
public Player getPlayer1() {
```



```

        return player1;
    }

    public void setPlayer1(Player player1) {
        this.player1 = player1;
    }

    public Player getPlayer2() {
        return player2;
    }

    public void setPlayer2(Player player2) {
        this.player2 = player2;
    }
}

```

:User.java:

```

package com.example.trivia;

public class User {
    protected String username;
    protected String uid;
    protected int score;
    protected int totalCorrect;
    protected int totalWrong;
}

```

```
public User(String username, String uid, int score, int totalCorrect, int totalWrong) {  
    this.username = username;  
    this.uid = uid;  
    this.score = score;  
    this.totalCorrect = totalCorrect;  
    this.totalWrong = totalWrong;  
}
```

```
public User() {  
    this.username = "";  
    this.uid = "";  
    this.score = -1;  
    this.totalCorrect = -1;  
    this.totalWrong = -1;  
}
```

```
public User(User user) {  
    this.username = user.username;  
    this.uid = user.uid;  
    this.score = user.score;  
    this.totalCorrect = user.totalCorrect;  
    this.totalWrong = user.totalWrong;  
}
```

```
public String getUsername() {  
    return username;  
}
```

```
public void setUsername(String username) {  
    this.username = username;  
}
```

```
public String getUid() {  
    return uid;  
}
```

```
public void setUid(String uid) {  
    this.uid = uid;  
}
```

```
public int getTotalCorrect() {  
    return totalCorrect;  
}
```

```
public void setTotalCorrect(int totalCorrect) {  
    this.totalCorrect = totalCorrect;  
}
```

```
public int getTotalWrong() {  
    return totalWrong;  
}
```

```
public void setTotalWrong(int totalWrong) {  
    this.totalWrong = totalWrong;  
}
```

```

public int getScore() {
    return score;
}

public void setScore(int score) {
    this.score = score;
}

public String getEmail(){
    //TODO: return username + "@1.1"
    return username;
}
}

```

:ScoreListAdapter.java:

```

package com.example.trivia;

import android.content.Context;
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 ScoreListAdapter extends
RecyclerView.Adapter<ScoreListAdapter.ScoreListViewHolder> {

    Context context;

    ArrayList<User> users;

    public ScoreListAdapter(Context context, ArrayList<User> users) {

        this.context = context;

        this.users = users;

    }

    @NonNull

    @Override

    public ScoreListViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int
viewType) {

        LayoutInflater inflater = LayoutInflater.from(context);

        View view = inflater.inflate(R.layout.recycler_view_score, parent, false);

        return new ScoreListViewHolder(view);

    }

    @Override

    public void onBindViewHolder(@NonNull ScoreListViewHolder holder, int position) {

        User user = users.get(position);

        holder.place.setText(Integer.toString(position + 1));

        holder.username.setText(user.getUsername());

        holder.score.setText(Integer.toString((int) user.getScore()));

    }
}
```

```

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

public class ScoreViewHolder extends RecyclerView.ViewHolder {
    TextView place;
    TextView username;
    TextView score;

    public ScoreViewHolder(@NonNull View itemView) {
        super(itemView);

        place = itemView.findViewById(R.id.placeLbl);
        username = itemView.findViewById(R.id.usernameLbl);
        score = itemView.findViewById(R.id.scoreLbl);
    }
}
}

```

:MainMenuActivity.java:

```
package com.example.trivia;
```

```
import androidx.annotation.NonNull;
```

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

```
import androidx.core.content.ContextCompat;
import androidx.core.view.GestureDetectorCompat;
import androidx.fragment.app.Fragment;
import androidx.fragment.app.FragmentTransaction;

import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.view.GestureDetector;
import android.view.MenuItem;
import android.view.MotionEvent;

import com.google.android.material.bottomnavigation.BottomNavigationView;
import com.google.android.material.navigation.NavigationBarView;

public class MainMenuActivity extends AppCompatActivity
    implements GestureDetector.OnGestureListener {
    private static final int MIN_SWIPE_LENGTH = 150;

    private GestureDetectorCompat swipeDetector;
    private BottomNavigationView navigationView;
    private NewGameFragment newGameFragment;
    private JoinGameFragment joinGameFragment;
    private ScoreFragment scoreFragment;
    private SettingsFragment settingFragment;
    private Intent bgMusicIntent;

    @Override
```

```

protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main_menu);

    //start background music service
    bgMusicIntent = new Intent(this, MusicService.class);
    ContextCompat.startForegroundService(this, bgMusicIntent);

    navigationView = findViewById(R.id.mainNavigationView);
    newGameFragment = new NewGameFragment();
    joinGameFragment = new JoinGameFragment();
    scoreFragment = new ScoreFragment();
    settingFragment = new SettingsFragment();
    swipeDetector = new GestureDetectorCompat(this, this);

    getSupportFragmentManager().beginTransaction().replace(R.id.mainFragmentContainer,
    newGameFragment).commit();

    navigationView.setSelectedItemId(R.id.newGameFragment);

    navigationView.setOnItemSelectedListener(new
    NavigationBarView.OnItemSelectedListener() {
        @Override
        public boolean onNavigationItemSelected(@NonNull MenuItem item) {
            switch (item.getItemId()){
                case R.id.newGameFragment:

                    getSupportFragmentManager().beginTransaction().replace(R.id.mainFragmentContainer,
                    newGameFragment).commit();

                    return true;
            }
        }
    });
}

```



```
case R.id.joinGameFragment:
```

```
getSupportFragmentManager().beginTransaction().replace(R.id.mainFragmentContainer,
joinGameFragment).commit();
```

```
return true;
```

```
case R.id.scoreFragment:
```

```
getSupportFragmentManager().beginTransaction().replace(R.id.mainFragmentContainer,
scoreFragment).commit();
```

```
return true;
```

```
case R.id.settingsFragment:
```

```
getSupportFragmentManager().beginTransaction().replace(R.id.mainFragmentContainer,
settingFragment).commit();
```

```
return true;
```

```
}
```

```
return false;
```

```
}
```

```
});
```

```
}
```

```
@Override
```

```
public boolean onDown(@NonNull MotionEvent e) {
```

```
return false;
```

```
}
```

```
@Override
```

```
public void onShowPress(@NonNull MotionEvent e) {
```

```
}
```

```
@Override
```

```
public boolean onSingleTapUp(@NonNull MotionEvent e) {
```

```
    return false;
```

```
}
```

```
@Override
```

```
public boolean onScroll(@NonNull MotionEvent e1, @NonNull MotionEvent e2, float  
distanceX, float distanceY) {
```

```
    return false;
```

```
}
```

```
@Override
```

```
public void onLongPress(@NonNull MotionEvent e) {
```

```
}
```

```
@Override
```

```
public boolean onFling(@NonNull MotionEvent e1, @NonNull MotionEvent e2, float  
velocityX, float velocityY) {
```

```
    Fragment currentFragment =  
    getSupportFragmentManager().findFragmentById(R.id.mainFragmentContainer);
```

```
    FragmentTransaction fragmentTransaction =  
    getSupportFragmentManager().beginTransaction();
```

```
    if(Math.abs(e2.getX() - e1.getX()) > MIN_SWIPE_LENGTH){
```

```
        //swipe
```

```

if(e2.getX() > e1.getX()){
    //right swipe
    if(currentFragment instanceof JoinGameFragment){
        fragmentTransaction.replace(R.id.mainFragmentContainer,
newGameFragment).commit();
        navigationView.setSelectedItemId(R.id.newGameFragment);
    }
    else if(currentFragment instanceof NewGameFragment){
        fragmentTransaction.replace(R.id.mainFragmentContainer,
scoreFragment).commit();
        navigationView.setSelectedItemId(R.id.scoreFragment);
    }
    else if(currentFragment instanceof SettingsFragment){
        fragmentTransaction.replace(R.id.mainFragmentContainer,
joinGameFragment).commit();
        navigationView.setSelectedItemId(R.id.joinGameFragment);
    }
}
else{
    //left swipe
    if(currentFragment instanceof ScoreFragment){
        fragmentTransaction.replace(R.id.mainFragmentContainer,
newGameFragment).commit();
        navigationView.setSelectedItemId(R.id.newGameFragment);
    }
    else if(currentFragment instanceof NewGameFragment){
        fragmentTransaction.replace(R.id.mainFragmentContainer,
joinGameFragment).commit();
        navigationView.setSelectedItemId(R.id.joinGameFragment);
    }
}

```

```

        else if(currentFragment instanceof JoinGameFragment){
            fragmentTransaction.replace(R.id.mainFragmentContainer,
settingFragment).commit();
            navigationView.setSelectedItemId(R.id.settingsFragment);
        }
    }
}

return true;
}

@Override
public boolean onTouchEvent(MotionEvent event) {
    swipeDetector.onTouchEvent(event);
    return super.onTouchEvent(event);
}
}

```

:activity_main_menu.xml:

```

<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"

```

```
android:layout_height="match_parent"
```

```
tools:context=".MainMenuActivity">
```

```
<androidx.fragment.app.FragmentContainerView
```

```
    android:id="@+id/mainFragmentContainer"
```

```
    android:name="androidx.navigation.fragment.NavHostFragment"
```

```
    android:layout_width="match_parent"
```

```
    android:layout_height="match_parent"
```

```
    android:layout_marginBottom="56dp"
```

```
    app:defaultNavHost="true"
```

```
    app:layout_constraintBottom_toBottomOf="parent"
```

```
    app:layout_constraintEnd_toEndOf="parent"
```

```
    app:layout_constraintHorizontal_bias="1.0"
```

```
    app:layout_constraintStart_toStartOf="parent"
```

```
    app:layout_constraintTop_toTopOf="parent"
```

```
    app:navGraph="@navigation/main_nav" />
```

```
<com.google.android.material.bottomnavigation.BottomNavigationView
```

```
    android:id="@+id/mainNavigationView"
```

```
    android:layout_width="match_parent"
```

```
    android:layout_height="56dp"
```

```
    android:background="#e1e3e5"
```

```
    app:layout_constraintBottom_toBottomOf="parent"
```

```
    tools:layout_editor_absoluteX="165dp"
```

```
    app:menu="@menu/main_menu"/>
```

```
</androidx.constraintlayout.widget.ConstraintLayout>
```

:fragment_wait_to_enemy.xml:

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<androidx.constraintlayout.widget.ConstraintLayout
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=".WaitToEnemyFragment"
```

```
android:background="@color/white">
```

```
<ProgressBar
```

```
android:id="@+id/progressBar"
```

```
style="?android:attr/progressBarStyle"
```

```
android:layout_width="match_parent"
```

```
android:layout_height="match_parent"
```

```
app:layout_constraintBottom_toBottomOf="parent"
```

```
app:layout_constraintEnd_toEndOf="parent"
```

```
app:layout_constraintStart_toStartOf="parent"
```

```
app:layout_constraintTop_toTopOf="parent" />
```

```
<TextView
```

```
android:id="@+id/textView"
```

```
android:layout_width="wrap_content"
```

```
android:layout_height="wrap_content"
```

```
android:text="Waiting for\nenemy..."
```

```
android:textAlignment="center"
```

```
android:textAppearance="@style/TextAppearance.AppCompat.Display2"
```

```
app:layout_constraintBottom_toBottomOf="parent"
```

```

    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>

```

:activity_login.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=".LoginActivity"
    android:orientation="vertical">

    <LinearLayout
        android:gravity="bottom"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="5"
        android:layout_marginHorizontal="60sp"
        android:orientation="vertical">

        <EditText
            android:id="@+id/loginUsernameTxt"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"

```

```

        android:ems="10"
        android:inputType="text"
        android:hint="Username" />

```

```

<EditText
    android:id="@+id/loginPasswordTxt"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:ems="10"
    android:inputType="textPassword"
    android:hint="Password" />

```

```

<EditText
    android:id="@+id/loginPasswordAgainTxt"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:ems="10"
    android:inputType="textPassword"
    android:hint="Password again" />

```

```

<Button
    android:id="@+id/loginBtn"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Log In"
    android:orientation="vertical"/>

```

```

<CheckBox
    android:id="@+id/loginRememberMeCb"

```



```

        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Remember Me"
        android:checked="false"/>
</LinearLayout>

```

```

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="3"
    android:gravity="bottom|center"
    android:orientation="vertical"
    android:paddingBottom="20dp">

```

```

<TextView
    android:id="@+id/loginStatusLbl"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text=""
    android:textColor="@color/red"
    android:textSize="16sp"
    android:layout_marginBottom="30dp"
    android:textAlignment="center"/>

```

```

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

```

```

        android:text="want to toggle mode?"
        android:textSize="20sp"/>
    <TextView
        android:id="@+id/toggleLoginModeLink"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="CLICK HERE!"
        android:textStyle="bold"
        android:textSize="18sp"/>
</LinearLayout>

</LinearLayout>

:recycler_view_score.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:gravity="center"
    android:orientation="horizontal">

    <TextView
        android:id="@+id/placeLbl"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="1"

```

```
android:textAlignment="center" />
```

```
<TextView
```

```
    android:id="@+id/usernameLbl"
```

```
    android:layout_width="0dp"
```

```
    android:layout_height="wrap_content"
```

```
    android:layout_weight="3"
```

```
    android:text="username"
```

```
    android:textAlignment="center" />
```

```
<TextView
```

```
    android:id="@+id/scoreLbl"
```

```
    android:layout_width="0dp"
```

```
    android:layout_height="wrap_content"
```

```
    android:layout_weight="2"
```

```
    android:text="0"
```

```
    android:textAlignment="center" />
```

```
</LinearLayout>
```

```
:fragment_loading.xml:
```

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
```

```
    xmlns:tools="http://schemas.android.com/tools"
```

```
    android:layout_width="match_parent"
```

```
    android:layout_height="match_parent"
```

```
    tools:context=".LoadingFragment"
```

```
android:background="#FFFFFF">
```

```
<ProgressBar
```

```
    android:id="@+id/progressBar2"
```

```
    style="?android:attr/progressBarStyle"
```

```
    android:layout_width="match_parent"
```

```
    android:layout_height="match_parent"
```

```
    android:layout_gravity="center"/>
```

```
<TextView
```

```
    android:id="@+id/textView2"
```

```
    android:layout_width="wrap_content"
```

```
    android:layout_height="wrap_content"
```

```
    android:layout_gravity="center"
```

```
    android:text="Loading..."
```

```
    android:textAppearance="@style/TextAppearance.AppCompat.Display3" />
```

```
</FrameLayout>
```

:activity_game.xml:

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<FrameLayout 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=".GameActivity"
```

```
    android:orientation="vertical"
```

```
android:id="@+id/gameLayout">
```

```
<androidx.constraintlayout.widget.ConstraintLayout
```

```
    android:layout_width="match_parent"
```

```
    android:layout_height="match_parent">
```

```
<androidx.constraintlayout.widget.ConstraintLayout
```

```
    android:id="@+id/linearLayout5"
```

```
    android:layout_width="match_parent"
```

```
    android:layout_height="wrap_content"
```

```
    android:orientation="horizontal"
```

```
    app:layout_constraintEnd_toEndOf="parent"
```

```
    app:layout_constraintStart_toStartOf="parent"
```

```
    app:layout_constraintTop_toTopOf="parent">
```

```
<!--TODO: home onclick-->
```

```
<ImageButton
```

```
    android:id="@+id/homeBtn"
```

```
    android:layout_width="50dp"
```

```
    android:layout_height="50dp"
```

```
    android:scaleType="fitCenter"
```

```
    android:padding="6dp"
```

```
    app:layout_constraintBottom_toBottomOf="parent"
```

```
    app:layout_constraintStart_toStartOf="parent"
```

```
    app:layout_constraintTop_toTopOf="parent"
```

```
    app:srcCompat="@drawable/ic_home" />
```

```
<TextView
    android:id="@+id/currentQuestionLbl"
    android:layout_width="wrap_content"
    android:layout_height="match_parent"
    android:text="10/15"
    android:textSize="30dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

```
<TextView
    android:id="@+id/questionLbl"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_marginStart="20dp"
    android:layout_marginTop="10dp"
    android:layout_marginEnd="20dp"
    android:text="this is the text of the question. it's lenth isvaious, so make suer it has
enogh space... text text text text text text text and mor text ssssss"
    android:textAppearance="@style/TextAppearance.AppCompat.Large"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/linearLayout5"></TextView>
```

```
<LinearLayout
    android:id="@+id/linearLayout"
    android:layout_width="340dp"
```

```

android:layout_height="307dp"
android:layout_marginStart="40dp"
android:layout_marginTop="30dp"
android:layout_marginEnd="40dp"
android:orientation="vertical"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toBottomOf="@+id/questionLbl">

```

```

<Button
    android:id="@+id/answer0Btn"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="1"
    android:text="This is the first answer" />

```

```

<Button
    android:id="@+id/answer1Btn"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="1"
    android:text="This is the second answer" />

```

```

<Button
    android:id="@+id/answer2Btn"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="1"

```

```
android:text="This is the third answer" />
```

```
<Button
```

```
    android:id="@+id/answer3Btn"
```

```
    android:layout_width="match_parent"
```

```
    android:layout_height="0dp"
```

```
    android:layout_weight="1"
```

```
    android:text="This is the fourth answer" />
```

```
</LinearLayout>
```

```
<ImageButton
```

```
    android:id="@+id/recordImgBtn"
```

```
    android:layout_width="110dp"
```

```
    android:layout_height="110dp"
```

```
    android:layout_marginStart="156dp"
```

```
    android:layout_marginEnd="156dp"
```

```
    android:backgroundTint="#00000000"
```

```
    android:scaleType="centerInside"
```

```
    app:layout_constraintBottom_toTopOf="@+id/progressImg"
```

```
    app:layout_constraintEnd_toEndOf="parent"
```

```
    app:layout_constraintHorizontal_bias="0.545"
```

```
    app:layout_constraintStart_toStartOf="parent"
```

```
    app:srcCompat="@drawable/ic_microphone" />
```

```
<ImageView
```

```
    android:id="@+id/progressImg"
```

```
    android:layout_width="0dp"
```

```
    android:layout_height="3dp"
```



```

app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintHorizontal_bias="0.0"
app:layout_constraintStart_toStartOf="parent"
app:srcCompat="@drawable/ic_launcher_background" />

```

```

</androidx.constraintlayout.widget.ConstraintLayout>

```

```

</FrameLayout>

```

:fragment_end_game.xml:

```

<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
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=".EndGameFragment"
    android:background="#E9FFFFFF"
    android:paddingBottom="60dp"
>

```

```

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

```

```

android:layout_marginTop="-10dp"
android:text="99999"
android:textColor="#FF007777"
android:textSize="64dp"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toBottomOf="@+id/textView3" />

```

```
<TextView
```

```

    android:id="@+id/textView3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="41dp"
    android:text="Your score:"

    android:textColor="#FF00BBBB"
    android:textSize="22dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/winnerLbl" />

```

```
<TextView
```

```

    android:id="@+id/enemyScoreCountLbl"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="-10dp"
    android:text="9999"
    android:textColor="#FF007777"

```

```

    android:textSize="64dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/enemyScoreLbl" />

```

```
<TextView
```

```

    android:id="@+id/enemyScoreLbl"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="16dp"

    android:text="Enemy's score:"
    android:textColor="#FF00BBBB"
    android:textSize="22dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/yourScoreCountLbl" />

```

```
<TextView
```

```

    android:id="@+id/textView4"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="13dp"
    android:text="Not bad at all..."
    android:textColor="#008D60"

    android:textSize="28dp"
    app:layout_constraintEnd_toEndOf="parent"

```

```
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toBottomOf="@+id/enemyScoreCountLbl" />
```

```
<LinearLayout
    android:id="@+id/linearLayout6"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginHorizontal="70dp"
    android:layout_marginTop="35dp"
    android:gravity="center"
    android:orientation="horizontal"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView4">
```

```
<ImageButton
    android:id="@+id/endGameReplayBtn"
    android:layout_width="0dp"
    android:layout_height="60dp"
    android:layout_weight="1"
    android:background="#00000000"
    android:padding="0dp"
    android:scaleType="fitCenter"
    android:src="@drawable/ic_end_game_play_again" />
```

```
<ImageButton
```

```

        android:id="@+id/endGameHomeBtn"
        android:layout_width="0dp"
        android:layout_height="110dp"
        android:layout_weight="1"
        android:background="#00000000"
        android:padding="0dp"
        android:scaleType="fitCenter"
        android:src="@drawable/ic_end_game_home" />

```

```

<ImageButton
    android:id="@+id/endGameShareBtn"
    android:layout_width="0dp"
    android:layout_height="60dp"
    android:layout_weight="1"
    android:background="#00000000"
    android:padding="0dp"
    android:scaleType="fitCenter"
    android:src="@drawable/ic_end_game_share" />

```

```

</LinearLayout>

```

```

<TextView
    android:id="@+id/winnerLbl"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="40dp"
    android:text="Username won!"
    android:textSize="36dp"
    android:textColor="#FF007777"

```

```
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent" />
```

```
</androidx.constraintlayout.widget.ConstraintLayout>
```

:fragment_game_id.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
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=".GameIdFragment"
android:background="@color/white">
```

```
<TextView
    android:id="@+id/startGameGameIdLbl"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="TextView"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    android:textSize="64dp"/>
```

```
</androidx.constraintlayout.widget.ConstraintLayout>
```

:fragment_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:gravity="center"
    android:orientation="vertical"
    android:paddingHorizontal="40dp"
    tools:context=".SettingsFragment">

    <TextView
        android:id="@+id/usernameLbl"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:text="USERNAME"
        android:textColor="@color/main_color_700"
        android:textSize="40dp" />

    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="30dp"
        android:layout_marginTop="10dp"
```

android:gravity="center"

android:orientation="horizontal">

<ImageButton

android:id="@+id/deleteUserImgBtn"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:backgroundTint="#00FFFFFF"

android:src="@android:drawable/ic_menu_delete" />

<ImageButton

android:id="@+id/editUsernameImgBtn"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:backgroundTint="#00FFFFFF"

android:paddingLeft="20dp"

android:src="@android:drawable/ic_menu_edit" />

<ImageButton

android:id="@+id/logoutImgBtn"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:backgroundTint="#00FFFFFF"

android:paddingLeft="20dp"

android:src="@android:drawable/ic_menu_set_as" />

</LinearLayout>


```
<LinearLayout
```

```
    android:layout_width="match_parent"
```

```
    android:layout_height="wrap_content"
```

```
    android:gravity="center"
```

```
    android:orientation="horizontal">
```

```
<ImageButton
```

```
    android:id="@+id/muteImgBtn"
```

```
    android:layout_width="80dp"
```

```
    android:layout_height="80dp"
```

```
    android:backgroundTint="#00FFFFFF"
```

```
    android:clickable="true"
```

```
    android:scaleType="centerCrop"
```

```
    android:src="@drawable/ic_volume" />
```

```
<SeekBar
```

```
    android:id="@+id/volumeSb"
```

```
    android:layout_width="match_parent"
```

```
    android:layout_height="wrap_content" />
```

```
</LinearLayout>
```

```
<LinearLayout
```

```
    android:layout_width="match_parent"
```

```
    android:layout_height="wrap_content"
```

```
    android:layout_marginTop="60dp"
```

```
    android:orientation="horizontal">
```

```
<Button
```

```

        android:id="@+id/exitSettingsBtn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Discard" />

```

```

<Button
    android:id="@+id/saveSettingsBtn"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:text="Save" />

```

```

</LinearLayout>

```

```

</LinearLayout>

```

:fragment_new_game.xml:

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center"
    android:orientation="vertical"
    tools:context=".NewGameFragment">

```

```
<TextView
```

```
    android:id="@+id/categoryLbl"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Category"
    android:textSize="24sp" />
```

```
<LinearLayout
```

```
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">
```

```
<Button
```

```
    android:id="@+id/allCategoriesBtn"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="1"
    android:text="All"
    android:background="#FFFFFF"
    android:textAllCaps="false"
    android:textSize="22sp"/>
```

```
<Button
```

```
    android:id="@+id/generalKnowledgeCategoryBtn"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="1"
```

```
android:text="General Knowledge"
```

```
android:background="#FFFFFF"
```

```
android:textAllCaps="false"
```

```
android:textSize="22sp"/>
```

```
<Button
```

```
    android:id="@+id/scienceCategoryBtn"
```

```
    android:layout_width="match_parent"
```

```
    android:layout_height="0dp"
```

```
    android:layout_weight="1"
```

```
    android:text="Science"
```

```
    android:background="#FFFFFF"
```

```
    android:textAllCaps="false"
```

```
    android:textSize="22sp"/>
```

```
<Button
```

```
    android:id="@+id/computerScienceCategoryBtn"
```

```
    android:layout_width="match_parent"
```

```
    android:layout_height="0dp"
```

```
    android:layout_weight="1"
```

```
    android:text="Computer Science"
```

```
    android:background="#FFFFFF"
```

```
    android:textAllCaps="false"
```

```
    android:textSize="22sp"/>
```

```
</LinearLayout>
```

```
<TextView
```

```

        android:id="@+id/difficultyLevelLbl"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Difficulty Level"
        android:layout_marginTop="15dp"
        android:textSize="24sp" />

```

```
<LinearLayout
```

```

        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="horizontal">

```

```
<Button
```

```

        android:id="@+id/easyDifficultyLevelBtn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:paddingVertical="20dp"
        android:text=" Easy "
        android:textSize="24dp"
        android:backgroundTint="#4CAF50"/>

```

```
<Button
```

```

        android:id="@+id/mediumDifficultyLevelBtn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:paddingVertical="20dp"
        android:text="Medium"
        android:textSize="24dp"

```

```
android:backgroundTint="#FFC107"/>
```

```
<Button
```

```
    android:id="@+id/hardDifficultyLevelBtn"
```

```
    android:layout_width="wrap_content"
```

```
    android:layout_height="wrap_content"
```

```
    android:paddingVertical="20dp"
```

```
    android:text=" Hard "
```

```
    android:textSize="24dp"
```

```
    android:backgroundTint="#F44336"/>
```

```
</LinearLayout>
```

```
<TextView
```

```
    android:id="@+id/questionCountLbl"
```

```
    android:layout_width="wrap_content"
```

```
    android:layout_height="wrap_content"
```

```
    android:text="Number Of Questions"
```

```
    android:layout_marginTop="15dp"
```

```
    android:textSize="24sp" />
```

```
<LinearLayout
```

```
    android:layout_width="wrap_content"
```

```
    android:layout_height="wrap_content"
```

```
    android:orientation="horizontal"
```

```
    android:gravity="center">
```

```
<Button
```

```
    android:id="@+id/questionCountDecBtn"
```

```

android:layout_width="60dp"
android:layout_height="60dp"
android:text="-"
android:textAlignment="center"
android:textSize="30sp"
android:textStyle="bold"/>

```

```

<TextView
    android:id="@+id/questionCountValueLbl"
    android:layout_width="60dp"
    android:layout_height="60dp"
    android:text="0"
    android:textSize="42sp"
    android:layout_marginHorizontal="20dp"
    android:textAlignment="center"/>

```

```

<Button
    android:id="@+id/questionCountIncBtn"
    android:layout_width="60dp"
    android:layout_height="60dp"
    android:text="+"
    android:textAlignment="center"
    android:textSize="30sp"
    android:textStyle="bold"/>

```

```

</LinearLayout>

```

```

<Button

```

```

        android:id="@+id/playBtn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Play!"
        android:textSize="64sp"
        android:layout_marginTop="20dp"/>
</LinearLayout>

```

:fragment_join_game.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="wrap_content"
    tools:context=".JoinGameFragment"
    android:orientation="vertical"
    android:gravity="center"
    android:layout_gravity="center">

    <TextView
        android:id="@+id/gameIdLbl"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Enter game ID" />

```



```
<EditText
```

```
    android:id="@+id/gameIdTxt"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:ems="10"
    android:inputType="number"
    android:textAlignment="center"
    android:text=""
    android:hint="ID"/>
```

```
<Button
```

```
    android:id="@+id/joinGameBtn"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:text="JOIN"/>
```

```
</LinearLayout>
```

:fragment_score.xml:

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<androidx.constraintlayout.widget.ConstraintLayout
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:gravity="center"
android:orientation="vertical"
tools:context=".ScoreFragment"
android:id="@+id/scoreLayout">

```

```
<LinearLayout
```

```

    android:id="@+id/linearLayout3"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="-5dp"
    android:orientation="horizontal"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/totalScoreLbl">

```

```
<TextView
```

```

    android:id="@+id/textView15"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:text="Correct Answers"
    android:textAlignment="center"
    android:textAppearance="@style/TextAppearance.AppCompat.Medium" />

```

```
<TextView
```

```

    android:id="@+id/textView16"
    android:layout_width="0dp"

```

```

        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Wrong Answers"
        android:textAlignment="center"
        android:textAppearance="@style/TextAppearance.AppCompat.Medium" />
</LinearLayout>

```

```

<LinearLayout
    android:id="@+id/linearLayout2"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="-3dp"
    android:orientation="horizontal"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/linearLayout3">

```

```

<TextView
    android:id="@+id/totalCorrectLbl"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:text="-1"
    android:textAlignment="center"
    android:textAppearance="@style/TextAppearance.AppCompat.Display2" />

```

```

<TextView
    android:id="@+id/totalWrongLbl"

```

```

        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="-1"
        android:textAlignment="center"
        android:textAppearance="@style/TextAppearance.AppCompat.Display2" />
    </LinearLayout>

```

```

<com.mikhaellopez.circularprogressbar.CircularProgressBar
    android:id="@+id/successPercentagePb"
    android:layout_width="135dp"
    android:layout_height="135dp"

    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.498"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/linearLayout2">

</com.mikhaellopez.circularprogressbar.CircularProgressBar>

```

```

<TextView
    android:id="@+id/successPercentageLbl"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="-1%"
    android:textAppearance="@style/TextAppearance.AppCompat.Display2"
    app:layout_constraintBottom_toBottomOf="@+id/successPercentagePb"
    app:layout_constraintEnd_toEndOf="@+id/successPercentagePb"

```

```

app:layout_constraintHorizontal_bias="0.493"
app:layout_constraintStart_toStartOf="@+id/successPercentagePb"
app:layout_constraintTop_toTopOf="@+id/successPercentagePb"
app:layout_constraintVertical_bias="0.435"
/>

```

```
<TextView
```

```

    android:id="@+id/textView19"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="144dp"
    android:layout_marginTop="-10dp"
    android:layout_marginEnd="144dp"
    android:text="Correct"
    app:layout_constraintBottom_toBottomOf="@+id/successPercentagePb"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.493"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/successPercentageLbl"
    app:layout_constraintVertical_bias="0.0" />

```

```
<LinearLayout
```

```

    android:id="@+id/linearLayout4"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_marginTop="15dp"
    android:orientation="horizontal"
    app:layout_constraintEnd_toEndOf="parent"

```

```
app:layout_constraintStart_toStartOf="parent"
```

```
app:layout_constraintTop_toBottomOf="@+id/successPercentagePb">
```

```
<TextView
```

```
    android:id="@+id/textView24"
```

```
    android:layout_width="0dp"
```

```
    android:layout_height="wrap_content"
```

```
    android:layout_weight="1"
```

```
    android:text="" />
```

```
<TextView
```

```
    android:id="@+id/textView28"
```

```
    android:layout_width="0dp"
```

```
    android:layout_height="wrap_content"
```

```
    android:layout_weight="3"
```

```
    android:text="Username"
```

```
    android:textAlignment="center" />
```

```
<TextView
```

```
    android:id="@+id/textView27"
```

```
    android:layout_width="0dp"
```

```
    android:layout_height="wrap_content"
```

```
    android:layout_weight="2"
```

```
    android:text="Score"
```

```
    android:textAlignment="center" />
```

```
</LinearLayout>
```

```
<androidx.recyclerview.widget.RecyclerView
```

```

        android:id="@+id/scoreRv"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/linearLayout4" />

```

<TextView

```

        android:id="@+id/textView29"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:layout_marginEnd="4dp"
        android:text="Your\nScore\nIs"
        android:textAlignment="textEnd"
        android:textSize="16sp"
        app:layout_constraintBottom_toTopOf="@+id/linearLayout3"
        app:layout_constraintEnd_toStartOf="@+id/totalScoreLbl"
        app:layout_constraintTop_toTopOf="@+id/totalScoreLbl"
        app:layout_constraintVertical_bias="0.47" />

```

<TextView

```

        android:id="@+id/totalScoreLbl"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="10000"

```

```

        android:textSize="84sp"
        android:textAppearance="@style/TextAppearance.AppCompat.Display4"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

```

```

<TextView
    android:id="@+id/textView31"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="3dp"
    android:layout_marginTop="50dp"
    android:text="Pts."
    android:textAppearance="@style/TextAppearance.AppCompat.Display1"
    app:layout_constraintStart_toEndOf="@+id/totalScoreLbl"
    app:layout_constraintTop_toTopOf="parent" />

```

```

</androidx.constraintlayout.widget.ConstraintLayout>

```

:strings.xml:

```

<resources>

    <string name="app_name">Trivia</string>
    <string name="title_activity_statistics">StatisticsActivity</string>
    <!-- Strings used for fragments for navigation -->
    <string name="first_fragment_label">First Fragment</string>

```



```

<string name="second_fragment_label">Second Fragment</string>
<string name="next">Next</string>
<string name="previous">Previous</string>

<string name="hello_first_fragment">Hello first fragment</string>
<string name="hello_second_fragment">Hello second fragment. Arg: %1$s</string>
<string name="title_activity_login">LoginActivity</string>
<string name="prompt_email">Email</string>
<string name="prompt_password">Password</string>
<string name="action_sign_in">Sign in or register</string>
<string name="action_sign_in_short">Sign in</string>
<string name="welcome">"Welcome !"</string>
<string name="invalid_username">Not a valid username</string>
<string name="invalid_password">Password must be >5 characters</string>
<string name="login_failed">"Login failed"</string>
<string name="hello_blank_fragment">Hello blank fragment</string>
</resources>

```

:dimens.xml:

```

<resources>

    <dimen name="fab_margin">16dp</dimen>

    <!-- Default screen margins, per the Android Design guidelines. -->
    <dimen name="activity_horizontal_margin">16dp</dimen>
    <dimen name="activity_vertical_margin">16dp</dimen>

</resources>

```

:themes.xml:

```
<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Theme.Trivia" parent="Theme.MaterialComponents.Light.NoActionBar">
        <!-- Primary brand color. -->
        <item name="colorPrimary">@color/main_color_500</item>
        <item name="colorPrimaryVariant">@color/main_color_700</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">?attr/colorPrimaryVariant</item>
        <!-- Customize your theme here. -->

    </style>
    <style name="Theme.Trivia.NoActionBar">
        <item name="windowActionBar">false</item>
        <item name="windowNoTitle">true</item>
    </style>
    <style name="Theme.Trivia.AppBarOverlay"
    parent="ThemeOverlay.AppCompat.Dark.ActionBar" />
    <style name="Theme.Trivia.PopupOverlay" parent="ThemeOverlay.AppCompat.Light" />
</resources>
```

: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">#FFFFFFF</color>

  <color name="red">#FFFF0000</color>
  <color name="green">#FF00FF00</color>
  <color name="blue">#FF0000FF</color>

  <color name="main_color_200">#5ADBEB</color>
  <color name="main_color_500">#009FB4</color>
  <color name="main_color_700">#006572</color>

</resources>
```