



My Fantasy 2020



מגיש הפרויקט: נעם כירם

ת.ז: 322568718

שם המנחה: שגיא פייגין

בית הספר: מקיף א' גימנסיה ריאלית ראשון לציון

אפריל 2020

תוכן העניינים

4.....	מבוא
6.....	ארכיטקטורה
6.....	כללי:
7.....	צד לקוח
7.....	תרשים מסכים
9.....	יחידות מרכזיות - פירוט המסכים באפליקציה
9.....	מסך פתיחה
10.....	מסך הרשמה
11.....	המסך הראשי
12.....	מסך רשימת הליגות:
13.....	מסך בנייה קבוצה
15.....	מסך בחירת שחקן:
16.....	מסך מילוי פרטי ליגה וקבוצה
18.....	מסך רשימת קבוצות בליגה
19.....	תרשים UML
20.....	יחידות מרכזיות ומדריך למפתח – מחלקות
21.....	המחלקה Buid_team.java
24.....	המחלקה CallMyServer.java
25.....	המחלקה Full_details_creation.java
26.....	המחלקה Leagues_listView
27.....	המחלקה Lv_Teams_In_League.java
28.....	המחלקה Main_menu.java
30.....	המחלקה MainActivity.java
33.....	המחלקה MyFirebaseMessagingService
35.....	המחלקה Pl_players_LV.java
36.....	המחלקה Splash_screen
37.....	צד שרת
37.....	יחידות מרכזיות ומדריך למפתח – מחלקות
38.....	המחלקה PremierLeauge
41.....	המחלקה NotifyClient
42.....	המחלקה Leauge
44.....	המחלקה Teams
46.....	המחלקה users
49.....	תיאור פרוטוקולי תקשורת
49.....	שרת לקוח

60.....	הממשק מול הליגה האנגלית
61.....	מבנה הנתונים
61.....	תרשים ERD
62.....	הטבלאות בבסיס הנתונים
62.....	טבלת משתמש
63.....	טבלת קבוצה
65.....	טבלת ליגה
66.....	טבלת שחקן - קבוצה
67.....	טבלת משתמש - ליגה
68.....	שאליות
71.....	טכנולוגיות מרכזיות בפרויקט
71.....	FLASK
71.....	FIREBASE CLOUD MESSAGING
71.....	MySQL
72.....	Glide
72.....	OkHttp
72.....	Gson
73.....	מדריך למשתמש
74.....	סיכום ורפלקציה
75.....	ביבליוגרפיה
76.....	הקוד המלא בפרויקט
76.....	צד לקוח
76.....	קבצי ג'אווה
124.....	קבצי XML – עיצוב מסכים
148.....	צד שרת

מבוא

אפליקציה My Fantasy 2020 היא משחק שבו המשתתפים מרכיבים קבוצות כדורגל דמיוניות עם שחקנים אמיתיים מהליגה האנגלית הבכירה בכדורגל ומנהלים ליגות על בסיס הנתונים הסטטיסטיים של השחקנים.

שאבתי השראה לפיתוח אפליקציה זו ממשחק ה-Fantasy הרשמי של הפריימר ליג האנגלית¹, שמפתחת מנהלת הליגה האנגלית הבכירה בשיתוף עם חברת EA. משחק זה קיים כמשחק מחשב או אפליקציה לטלפון הנייד. אני אוהד כדורגל ונחשפתי למשחק זה לפני שלוש שנים. הצד הטכני של המשחק עניין אותי ורציתי לפתח אפליקציה דומה שבה יוכלו לשחק אוהדי כדורגל המתעניינים בליגה האנגלית וכך ליצור חיבור בין אוהדים אלו. הפרויקט מעניק מבט מעמיק על עונת הכדורגל האחרונה באמצעות נתונים סטטיסטיים על שחקני הפריימר ליג.

ייחודה של האפליקציה הוא בעובדה שהנבחרות במשחק מורכבות פעם אחת בלבד ולא נעשים חילופי שחקנים לאחר בניית הנבחרת. בניית הנבחרת באפליקציה שפיתחתי מצריכה חשיבה אסטרטגית המבוססת על ניהול תקציב מוגבל ומוגדר מראש ונתונים סטטיסטיים של השחקנים בעונה הקודמת - חשיבה לאחור. לעומת זאת, באפליקציית הפאנטזי הרשמית חילופי השחקנים נמשכים לאורך כל העונה הנוכחית שנערכת במציאות מתוך חשיבה על מחזורי המשחקים העתידיים.

הפלטפורמה החדשה מאפשרת לשחקנים להרכיב לעצמם קבוצות כדורגל דמיוניות עם שחקני פרמייר ליג אמיתיים, ולנהל ליגות על בסיס הנתונים הסטטיסטיים של השחקנים. כל משתתף בונה לעצמו נבחרת עם שחקנים מעונת המשחקים הנוכחית. המשתמש יכול לאתגר חברים נוספים שברשותם האפליקציה לבנות גם הם נבחרת כדי שישחקו נגדו. השחקנים נבחרים לפי תפקידם על המגרש (שוער, מגן, קשר או חלוץ) מתוך רשימה מובנית.

לאחר בניית הקבוצה, כל נבחרת מקבלת ניקוד לפי מספר הנקודות שהשחקנים קיבלו במשחק הרשמי של הליגה האנגלית. המנצח הוא מי שקבוצתו זכתה בניקוד הכולל הגבוה ביותר.

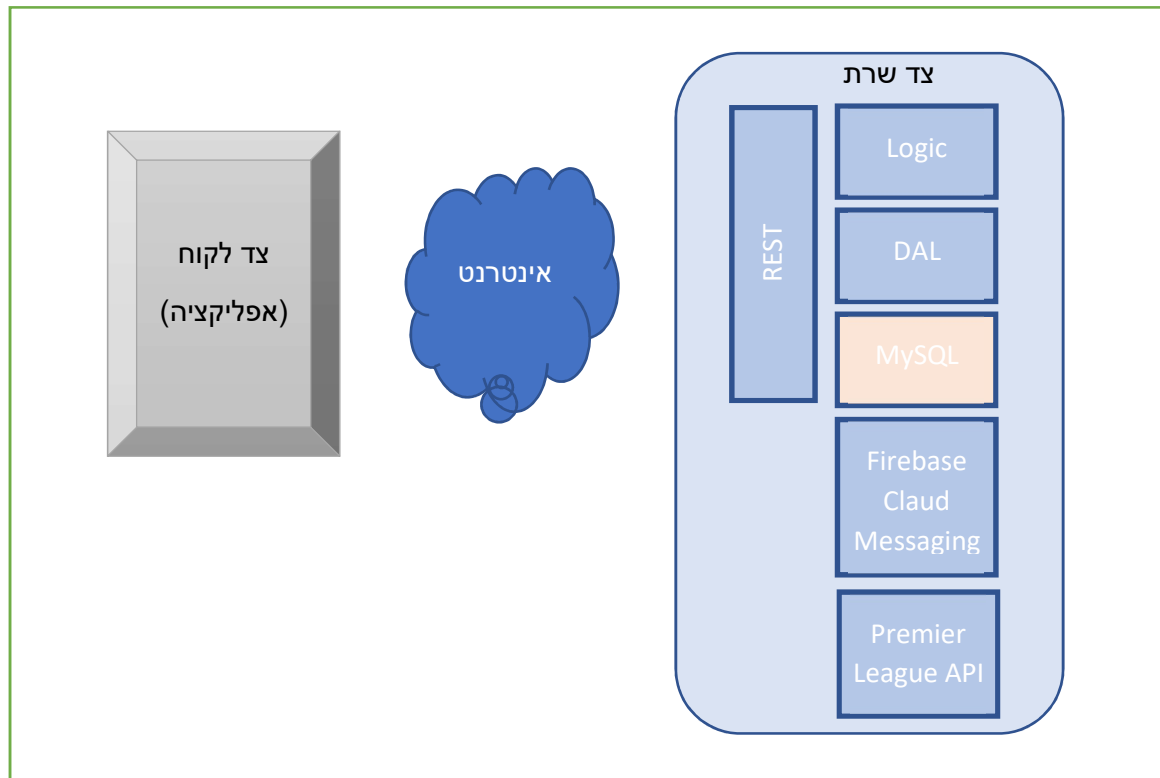
¹ משחק הפאנטזי הרשמי של הליגה האנגלית

בניית האפליקציה הצריכה התמודדות עם בעיות שונות, כגון קישור האפליקציה אל השרת, שמירת המידע באמצעות בסיס הנתונים, השגת המידע מבסיס הנתונים של הליגה האנגלית ושליחת הודעות למשתמשים השונים באפליקציה.

לשם כך, האפליקציה נעזרת בשרת שנכתב בשפת פייתון. השרת דוגם את אתר הפרמייר ליג בזמנים קבועים, מאחזר את הנתונים הסטטיסטיים של השחקנים ושומר אותם על גבי השרת. הנתונים מגיעים במקור מאתר הפרמייר ליג באמצעות שירותי REST. כמו כן, השרת שומר מידע על מצב המשתמשים שאותגרו (באיזה שלב באפליקציה נמצא כל משתמש - אתגר נשלח/ בונה קבוצה וכו'). צד הלקוח ממומש על ידי אפליקציית אנדרואיד בשפת JAVA.

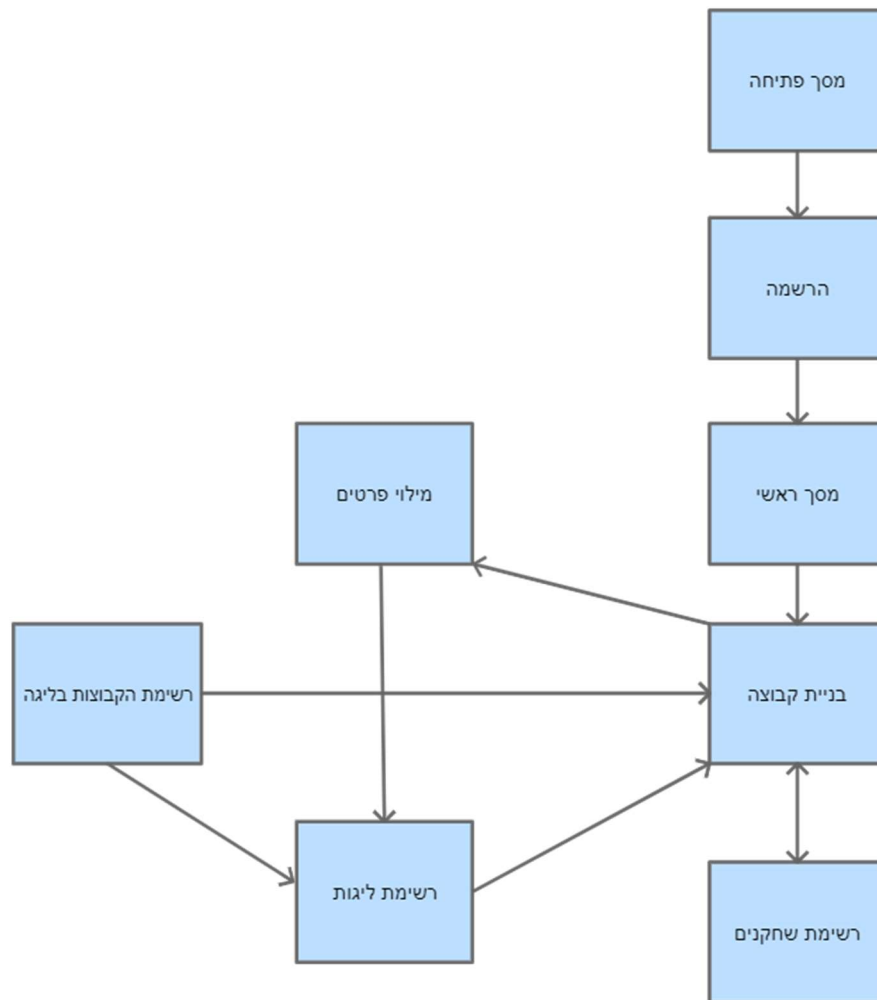
ארכיטקטורה

כללי:



צד לקוח

תרשים מסכים



הסבר כללי:

לאחר תהליך הרישום, מופנה המשתמש למסך הראשי.

במסך זה יש למשתמש שתי אפשרויות בחירה:

- הראשונה, יצירת ליגה עם קבוצה משלו.
- השנייה היא צפייה בליגות שבהן המשתמש הוזמן אליהן או שהמשתתף יצר אותן בעצמו בעבר.

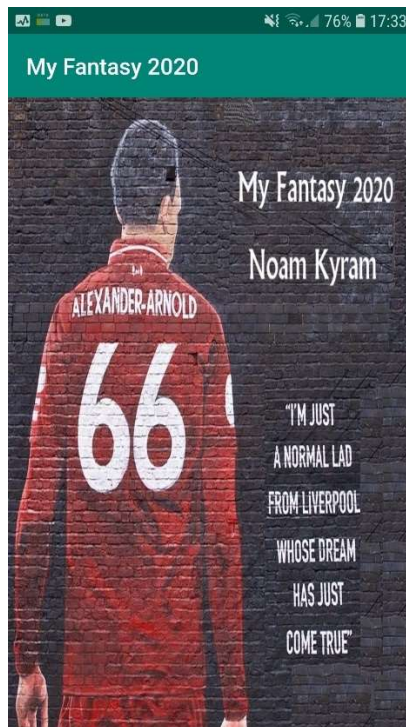
אם המשתמש בוחר ליצור ליגה משלו, המשתמש נדרש במסכים הבאים ליצור את הליגה ואת הקבוצה ולהזמין משתמשים נוספים להתחרות מולם: מסך בניית קבוצה, מסך מילוי פרטים ומסך בחירת שחקן מתוך רשימת שחקנים.

במידה והמשתמש בחר להיענות לאתגר שהוזמן אליו יופיע מסך רשימת הליגות שהמשתמש הוזמן אליהן. בהיענות לאתגר חדש, נדרש המשתמש להרכיב קבוצה שתשתתף בליגה שהוזמן אליה (במסך בניית קבוצה) ולתת שם לקבוצה (במסך מילוי פרטים).

לאחר צירוף הקבוצה לליגה ניתן לראות את רשימת הקבוצות בליגה שאליה נענה ואת השחקנים השייכים לקבוצות אלה.

יחידות מרכזיות - פירוט המסכים באפליקציה

מסך פתיחה



תפקיד המסך:

מסך זה מוצג למספר שניות ומוצגת התמונה, יש מדידת זמן של משך הצגת התמונה.

קלט:

לא מקבל

פלט:

לא מחזיר

[מסך הרשמה](#)
תפקיד המסך:

המסך בודק האם המשתמש נרשם לאפליקציה ומנהל את תהליך ההרשמה שלו במידה והמשתמש טרם נרשם.

קלט + פלט:

יש קריאה לשרת לפונקציית ה-API המחזירה האם שם המשתמש שהוכנס חוקי או לא.

קלט:

יש קלט מהמשתמש בהכנסת שם המשתמש בתיבת טקסט ולחיצה על כפתור לאישור שם המשתמש.

קלט + פלט:

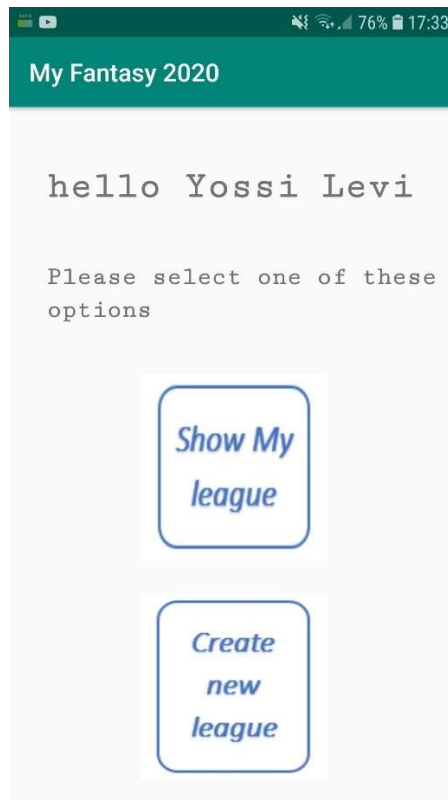
יש קריאה לשרת לפונקציית ה-API המוסיפה את שם המשתמש החוקי למבנה הנתונים. מוחזרת תשובה כאשר שם המשתמש נוסף בהצלחה.

פלט:

שם המשתמש שאושר נכתב לקובץ שם המשתמש.

פלט:

הטוקן של המשתמש נכתב לקובץ המתאים כאשר הוא נוסף למערכת.

[המסך הראשי](#)**תפקיד המסך:**

המסך מציג את האפשרויות של המשתמש לשחק במשחק וקולט את בחירתו של המשתמש.

קלט:

יש קלט של בחירתו של המשתמש.

הטוקן הכתוב בקובץ המכיל את הטוקן נקלט לצורך בדיקה האם יש צורך בעדכון מול השרת.

שם המשתמש נקלט מקובץ שם המשתמש על מנת להציגו בראש העמוד.

פלט:

במידה והמשתמש בוחר באפשרות של יצירת ליגה, הוא מועבר למסך בניית קבוצה שבו המשתמש יוצר ליגה.

מסך רשימת הליגות:

chipopo I	792
Created By: Yossi Levi	pts
2 team/s in league	2 place
IIII	567
Created By: Yossi Levi	pts
1 team/s in league	1 place
ggg	839
Created By: Yossi Levi	pts
1 team/s in league	1 place
III	519
Created By: Yossi Levi	pts
1 team/s in league	1 place
hjh	918
Created By: Yossi Levi	pts

תפקיד המסך:

המסך מציג את רשימת הליגות שהמשתמש הוזמן אליהם או יצר אותן. המסך קולט את בחירתו של המשתמש ובודק האם בנה קבוצה בליגה.

פלט + קלט:

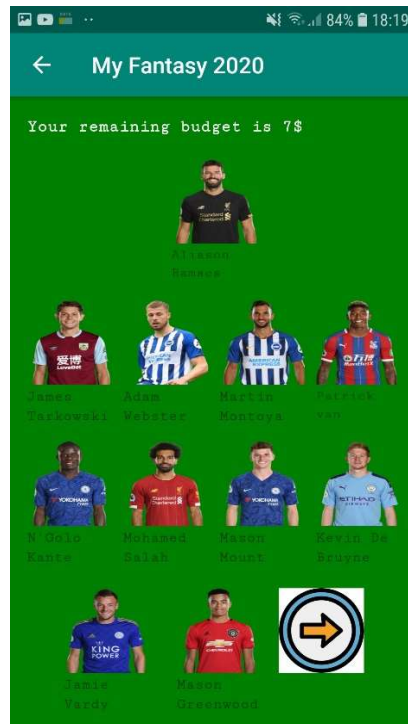
יש קריאה לשרת לפונקציית ה-API שמחזירה את רשימת הליגות.

קלט:

הליגה הנבחרת ע"י המשתמש נקלטת באמצעות לחיצה על איבר ברשימה.

פלט:

במידה והמשתמש טרם נענה לליגה, הוא יועבר למסך בניית קבוצה ביחד עם המספר המזהה של הליגה והסטטוס של הוספת קבוצה לליגה קיימת.

מסך בנייה קבוצה**תפקיד המסך:**

מאפשר למשתמש לבנות קבוצה מאחד עשר שחקני פרמייר ליג או לצפות בקבוצה שנבנתה באחת הליגות שהמשתמש משתתף בהן.

קלט:

במצב צפייה בקבוצה קיימת, מועבר ממסך הקבוצות הנמצאות בליגה מספרה המזהה של הקבוצה לצפייה.

קלט + פלט:

רשימת שחקנים מחזירה את המספרים המזהים של השחקנים לפי מספר מזהה של קבוצה מסוימת.

קלט + פלט:

יש קריאה לשרת לפונקציית ה-API שמחזירה את רשימת השחקנים המשחקים בפרמייר ליג.

קלט:

יש זיהוי לחיצה על עמדה של שחקן מהמשתמש.

קלט:

יש זיהוי לחיצה של המשתמש כאשר סיים לבנות את הקבוצה.

קלט:

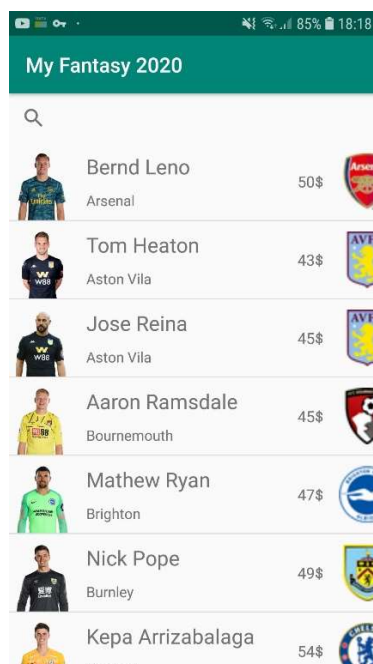
מהמסך הקודם מתקבל מספרה המזהה של הליגה אליה מצטרף המשתמש במידה וטרם בנה קבוצה בליגה שבחר.

פלט:

השחקנים מהעמדה שנבחרה מועברים למסך רשימת השחקנים, התקציב שנותר, מערך השחקנים מאותה עמדה כדי למנוע כפילויות.

קלט:

מועבר ממסך רשימת השחקנים מספרו המזהה של השחקן, המחיר שלו, המיקום בהרכב שבו נבחר השחקן.

מסך בחירת שחקן:


Player	Team	Price
Bernd Leno	Arsenal	50\$
Tom Heaton	Aston Villa	43\$
Jose Reina	Aston Villa	45\$
Aaron Ramsdale	Bournemouth	45\$
Mathew Ryan	Brighton	47\$
Nick Pope	Burnley	49\$
Kepa Arrizabalaga	Chelsea	54\$

תפקיד המסך:

מאפשר לשחקן לבחור את השחקן מהעמדה שנבחרה מהעמוד הקודם, מתוך רשימת השחקנים המשחקים בליגה מאותה עמדה.

קלט:

המידע מועבר למסך רשימת שחקנים: רשימת השחקנים מהעמדה שנבחרה, התקציב שנותר, מערך השחקנים מאותה עמדה כדי למנוע כפילויות.

פלט:

רשימת השחקנים מוצגת ברשימה.

קלט:

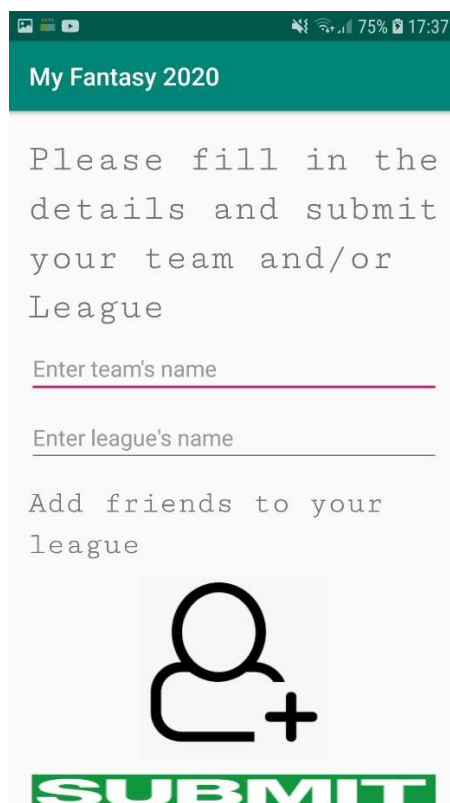
בחירת השחקן על ידי המשתמש מתבצעת באמצעות זיהוי לחיצה על איבר מהרשימה.

קלט + פלט:

בעת הקלדה של המשתמש בתיבת החיפוש את שמו של השחקן שאותו רוצה לבחור, שמם של השחקנים המכיל את הקלט שהוקלד יופיע ברשימה.

קלט + פלט:

יש קריאה לשרת לפונקציית ה-API שמחזירה את כתובות ה-URL המשמשות להשגת התמונות של השחקנים והקבוצות.

מסך מילוי פרטי ליגה וקבוצה

תפקיד המסך:

מסך זה מאפשר למשתמש להשלים את מילוי הפרטים האחרונים ביצירת קבוצה ו/או ליגה חדשה.

קלט:

מועבר למסך זה מהמסך הקודם האם זהו מצב יצירת ליגה וקבוצה חדשים או הוספת קבוצה לליגה קיימת והמספרים המזהים של אחד עשר השחקנים שנבחרו.

קלט:

במידה וזהו מצב הוספת קבוצה לליגה קיימת מועבר המספר המזהה של הקבוצה.

קלט + פלט:

יש קריאה לשרת לפונקציית ה-API שמחזירה את המשתמשים האחרים במשחק.

קלט:

המשתמש מזין דרך תיבת טקסט את שם הקבוצה החדשה.

קלט:

במידה וזהו מצב יצירת ליגה חדשה, על המשתמש להזין דרך תיבת טקסט נוספת את שם הליגה החדשה.

קלט:

בעת קליטת לחיצה של המשתמש על תמונה המופיעה ביצירת ליגה תיפתח חלונית עם רשימת המשתמשים האחרים במשחק ועל המשתמש להחליט אילו משתמשים הוא רוצה להזמין לליגה החדשה. בעת סיום בחירת המשתמשים תישמר רשימת המשתמשים שנבחרו.

קלט:

יש זיהוי לחיצה של המשתמש על כפתור אישור הנתונים.

קלט + פלט:

יש קריאה לשרת לפונקציית ה-API שמוסיפה את הליגה ו/או הקבוצה למבנה הנתונים. מוחזרת מהשרת תשובה כאשר הנתונים נשמרו בהצלחה.

מסך רשימת קבוצות בליגה

Rank	Team and Manager	points
1	real madirz Moshe Cohen	1352
2	wow its bad Yossi Levi	792

תפקיד המסך:

המסך מציג את רשימת הקבוצות המשתתפות בליגה שנבחרה על ידי המשתמש, את פרטיון וממוינות לפי מספר הנקודות שלהן.

קלט:

מסך זה מקבל כקלט מהמסך הקודם את רשימת הקבוצות ופרטיון מהליגה שנבחרה, ושם הליגה.

פלט:

המסך מציג את פרטי הקבוצות בתוך הרשימה.

קלט:

יש זיהוי של לחיצה של המשתמש על קבוצה מתוך הרשימה.

פלט:

מסך זה מעביר למסך בניית קבוצה את המספר המזהה של הקבוצה כדי שיוציג את תמונות השחקנים המשתתפים בקבוצה.

תרשים UML



יחידות מרכזיות ומדריך למפתח – מחלקות

java	64.2 KB	res	796.3 KB
└─ com	64.2 KB	└─ drawable	444.7 KB
└─ example	64.2 KB	└─ add_people.png	2.6 KB
└─ myfantasy2020	64.2 KB	└─ court.jpg	111.6 KB
└─ Build_team.java	13.0 KB	└─ create_btn.png	5.2 KB
└─ CallMyServer.java	1.9 KB	└─ done2.png	5.9 KB
└─ Full_details_creation.java	8.4 KB	└─ empty_shirt.png	9.8 KB
└─ GettingDeviceTokenService.java	559 Bytes	└─ empty2.png	6.7 KB
└─ League_LV_adapter.java	2.6 KB	└─ ic_launcher_background.xml	4.8 KB
└─ League_obj.java	1.2 KB	└─ next2.png	5.8 KB
└─ Leagues_listView.java	4.9 KB	└─ show_league_btn.png	5.2 KB
└─ List_PLs_obj.java	181 Bytes	└─ submit_blue.png	6.6 KB
└─ Lv_Teams_In_League.java	2.7 KB	└─ submit_team.png	2.8 KB
└─ Main_menu.java	2.9 KB	└─ title.png	12.5 KB
└─ MainActivity.java	5.2 KB	└─ trent_image.jpg	265.3 KB
└─ MyFirebaseMessagingService.j...	2.6 KB	└─ drawable-v24	1.9 KB
└─ PL_Player_Adapter.java	4.4 KB	└─ layout	28.4 KB
└─ Pl_players_LV.java	5.6 KB	└─ activity_build_team.xml	11.8 KB
└─ Player_layout.java	1.0 KB	└─ activity_full_details_creation.xml	2.3 KB
└─ Plplayer_obj.java	1.9 KB	└─ activity_leagues_list_view.xml	568 Bytes
└─ Rootobject.java	178 Bytes	└─ activity_lv_teams_in_league.xml	2.2 KB
└─ Splash_screen.java	884 Bytes	└─ activity_main.xml	1.8 KB
└─ Team_Adapter.java	1.9 KB	└─ activity_main_menu.xml	1.6 KB
└─ Team_obj.java	2.3 KB	└─ activity_pl_players_lv.xml	720 Bytes
		└─ activity_splash_screen.xml	576 Bytes
		└─ league_row.xml	2.4 KB
		└─ one_item_list.xml	892 Bytes
		└─ pl_player_row.xml	1.9 KB
		└─ team_row.xml	1.8 KB

Buid_team.java המחלקה

שם קובץ:

תפקיד המחלקה:

תפקידה לטפל בענייני מסך בניית הקבוצה, ניהול אירועי לחיצה על פקדים, הצגת הפקדים, ניהול המידע המוחזר מהמסכים העוקבים והקודמים אל מסך זה ובדיקת תקינות של הקבוצה בסיום בנייה.

תכונות

11 תמונות לחיצות המציגות את 11 השחקנים בקבוצה. בלחיצה עליהן יוצג מסך בחירת שחקן. בתחילת בחירת הקבוצה יוצג תמונת ברירת מחדל.

11 תוויות טקסט המציגות את שמות השחקנים. בכניסה אל המסך יוצג טקסט ברירת מחדל.

תוויות טקסט tvBudget המציגה את התקציב שנשאר. התקציב ההתחלתי הוא \$750.

מחרוזת status ששומרת את אופן הצגת המסך: לבניית ליגה וקבוצה או צפייה בקבוצה קיימת.

leagueId מטיפוס מספר שלם. שומר את מספר הליגה במידה והמשתמש מוסיף קבוצה משלו לליגה קיימת שהוזמן אליה.

Starting11 מערך של עצמים מטיפוס Player_layout שכל איבר המערך מייצג את סידור של שחקן מתוך הקבוצה.

Plsarr מערך של עצמים מטיפוס שחקן פרמייר ליג, משמש להצגת קבוצה מסוימת.

Budget מטיפוס מספר שלם, שומר את כמות התקציב שנשאר.

examplephotoUrl מטיפוס מחרוזת, שומר את כתובת ה-url שמציגה תמונת שחקן.

פעולות

1. onCreate

טענת כניסה: אין

טענת יציאה: אין

תפקיד הפעולה: הפעולה מציגה את הפקדים על המסך בהתאם לסטטוס שהתקבל מהמסך הקודם: אם הסטטוס הוא הצגת קבוצה יוצגו תמונות השחקנים ולא יאפשר מעבר אל המסך הבא לא יוצג תקציב. אחרת תהיה אפשרות בחירת שחקנים והצגת תקציב שנותר.

2. onClick

טענת כניסה: הפקד שנלחץ על ידי המשתמש.

טענת יציאה: אין

תפקיד הפעולה:

ניהול הפקדים שנלחצו בהתאם לסטטוס.

3. onActivityResult

טענת כניסה:

resultCode מטיפוס מספר שלם, שומר את הסטטוס מהמסך הקודם.

dataIntent מטיפוס Intent מאחסן את כל המידע מהמסך הקודם.

תפקיד הפעולה:

הפעולה משיגה את פרטי השחקן שנבחר במסך הקודם, מציגה את תמונתו, שומרת את פרטיו ומציגה את התקציב החדש.

4. **onBackPressed**

טענת כניסה: אין

טענת יציאה: אין

תפקיד הפעולה: הפעולה מטפלת באירוע לחיצה על מקש חזרה למסך הקודם.

המחלקה CallMyServer.java

תפקיד המחלקה:

לטפל בחיבור בין הלקוח אל השרת, שליחת בקשות באמצעות url וקבלת תשובה מהשרת באמצעות מחרוזת.

תכונות המחלקה:

Address מטיפוס מחרוזת, שומר את כתובת ip של השרת.
 Port מטיפוס מספר שלם, שומר את הפורט שבו ניתן לתקשר עם השרת.
 Msg מטיפוס מחרוזת, שומר את מבנה של תחילת url של הבקשה מהשרת. מורכב מהפורט ומכתובת ip של השרת.
 Client מטיפוס OkHttpClient מטפל בניהול צד הלקוח בקשר מול השרת.
 R מטיפוס response שומר את התשובה מהשרת בתבנית.
 Res מטיפוס מחרוזת, שומר את התשובה שהתקבלה מהשרת.
 Req מטיפוס Request, שומר את פרטי הבקשה מהשרת בתבנית.

פעולות עיקריות:

1. getFromServer

טענת כניסה:

-Params מטיפוס מחרוזת, הפרמטרים המרכיבים חלק מכתובת url כמו הפעולה המתבקשת מהשרת, והפרמטרים הדרושים בשביל הפעולה.

טענת יציאה:

הפעולה מחזירה מחרוזת שהיא גוף התשובה המתקבלת מהשרת.

תפקיד הפעולה:

הפעולה יוצרת את החיבור עם השרת, שולחת את הבקשה שהתקבלה כפרמטר, ומנתחת את התשובה שהתקבלה מהשרת.

[Full details creation.java המחלקה](#)

תפקיד המחלקה:

המחלקה מטפלת במסך מילוי הפרטים שמופיע אחרי מסך בניית הקבוצה, קבלת המידע המתקבל מהמסך הקודם, אישור הנתונים הממולאים ע"י המשתמש, טיפול בחיצה על פקדים וסיום תהליך יצירת הקבוצה או הליגה.

תכונות המחלקה:

2 תיבות טקסט המשמשות למילוי שם הקבוצה והליגה (בעת יצירת ליגה)

תמונה לחיצה המציגה את דיאלוג בחירת משתמשים לליגה החדשה.

תמונה לחיצה לאישור הנתונים.

מחרוזת status ששומר את מצב תצוגת המסך – יצירת ליגה או הוספת קבוצה לליגה.

מערך של מחרוזות team_codes שכל איבר בו מציין את המספר המזהה של השחקנים שנבחרו במסך הקודם.

מערך של מחרוזות השומר את המשתמשים שהוזמנו לליגה.

League_id מטיפוס מספר שלם, מספר הליגה שאליה מוסיפים את הקבוצה החדשה במידה והסטטוס הוא הוספת קבוצה לליגה קיימת.

פעולות עיקריות:

1. onCreate

תפקיד הפעולה:

הפעולה מציגה את הפקדים על המסך, מגדירה פעולת ניהול לחיצה על פקדים בהתאם לסטטוס המתקבל מהמסך הקודם.

המחלקה Leagues listView

תפקיד המחלקה:

המחלקה מציגה את מסך הליגות שבהן המשתמש משתתף, מנהלת את המסך, ומטפל באירוע לחיצה על אחת הליגות מהרשימה.

תכונות המחלקה:

Lv_my_leagues מטיפוס ListView הפקד שאחראי להצגת הרשימה.

Leagues_adapter מטיפוס League_LV_adapter, מחלקת עזר העוזרת להקצות לכל איבר ברשימה layout משלו ברשימה, ומגדירה פעולת ניהול לחיצה על איבר ברשימה.

פעולות עיקריות:

onCreate

תפקיד המחלקה: הפעולה מאתחלת את רשימת הליגות על ידי פנייה למחלקה הפונה אל השרת CallMyServer ומגדירה פעולה לניהול לחיצה על איבר ברשימה.

setOnClickListener

תפקיד הפעולה:

הפעולה מתמודדת עם אירוע לחיצה על איבר ברשימה – היא מזהה את האיבר שנלחץ, בודקת האם המשתמש יצר קבוצה בליגה – אם לא יבנה קבוצה, אחרת הוא יצפה ברשימת הקבוצות בליגה שנבחרה.

[Lv Teams In League.java](#) המחלקה

תפקיד המחלקה:

לנהל את המסך של הקבוצות שנמצאות בליגה שנבחרה במסך הקודם, לטפל באירועי לחיצות על איברים ברשימה ולחיצות על לחצני חזרה לעמוד הקודם.

תכונות המחלקה:

lvTeamsInLeague מטיפוס ListView, סוג הפקד שבו מוצגת רשימת הקבוצות.

tvLeagueName מטיפוס תווית טקסט, מוצג בו שם הליגה של הקבוצה.

פעולות המחלקה:

1. onCreate

תפקיד הפעולה:

הפעולה מגדירה את הפקדים ומגדירה להם פעולה המאזינה על לחיצות עליהם.

2. setOnItemClickListener

תפקיד הפעולה:

הפעולה מאזינה ללחיצה על אחד מאיברי הרשימה ובעת לחיצה על קבוצה בליגה היא מאתרת את הקבוצה שנבחרה ושולחת את המספרים המזהים של שחקני הקבוצה אל מסך בניית הקבוצה כדי שתציג את השחקנים השייכים לקבוצה שנבחרה.

3. onBackPressed

תפקיד הפעולה:

הפעולה מטפלת באירועי לחיצה על כפתור חזרה לעמוד הקודם.

[המחלקה Main_menu.java](#)

תפקיד המחלקה:

מחלקה זאת מציגה את המסך הראשי – בו המשתמש יכול לבחור האם ליצור ליגה חדשה או האם לצפות בליגות קיימות. במסך זה יש גם בדיקה האם הטוקן של האפליקציה מול Google Firebase השתנה.

תכונות המחלקה:

תווית טקסט המציגה את שם המשתמש בראש המסך

שתי תמונות לחיצות שהן שתי האפשרויות של המשתמש - האם ליצור ליגה חדשה או האם לצפות בליגות קיימות.

פעולות המחלקה:

1. onCreate

תפקיד הפעולה:

מטפלת באירוע היווצרות המסך, מגדירה פעולות ניהול לחיצה על הפקדים המתאימים, מציגה את שם המשתמש על המסך ע"י קריאה מקובץ המשתמש ובודקת את בדיקה האם הטוקן של האפליקציה מול Google Firebase השתנה.

2. onClick

טענת כניסה:

View v- הפקד שנלחץ

טענת יציאה: אין

תפקיד הפעולה:

מטפלת באירועי הלחיצה על הפקדים הלחיצים, בודקת איזה מהפקדים נלחץ ומעבירה למסך המתאים עם המידע הרלוונטי.

checkToken .3

טענת כניסה: שם המשתמש, בצורת מחרוזת.

טענת יציאה: אין

תפקיד הפעולה: הפעולה קוראת מן הקובץ שבודק האם הטוקן מול גוגל פיירבייס הוא תוכן קובץ הטוקן של המשתמש: אם אין התאמה, נשלחת בקשה לשרת לעדכן את הטוקן של שם המשתמש.

המחלקה MainActivity.java

תפקיד המחלקה:

המחלקה מטפלת בניהול מסך ההרשמה של האפליקציה וניהול תהליך ההרשמה אצל המשתמש.

תכונות המחלקה:

ibRegister מטיפוס תמונה לחיצה. משמש ככפתור לאישור הנתונים.

etNewUser מטיפוס תיבת טקסט. משמש להזנת שם המשתמש החדש.

file_user_key משתנה סטטי מטיפוס מחרוזת. מכיל את שם הקובץ המכיל את שם המשתמש.

file_token_key משתנה סטטי מטיפוס מחרוזת. מכיל את שם הקובץ המכיל את הטוקן של אפליקציית המשתמש.

Path – משתנה סטטי המכיל את כתובת הקבצים על המכשיר. מטיפוס מחרוזת.

פעולות המחלקה:

1. **onCreate**

תפקיד הפעולה: בדיקה האם המשתמש טרם נרשם. אם המשתמש רשום במשק, יש לעבור למסך הראשי. אם טרם, הפעולה מציגה את המסך כדי להתחיל את תהליך ההשלמה.

2. **goNextScreen**

טענת כניסה: אין

טענת יציאה: אין

תפקיד הפעולה: לעבור למסך הראשי.

3. onClick

טענת כניסה:

View v - הפקד שנלחץ

טענת יציאה: אין

תפקיד הפעולה:

מטפלת באירועי הלחיצה על אישור הנתונים. יש זיהוי הטקסט שהזין המשתמש, ובדיקה אצל השרת האם קיים שם משתמש עם שם זהה: אם לא קיים שם משתמש כזה במבנה הנתונים יתווסף המשתמש החדש בשרת, אחרת תוצג הודעה מתאימה והמשתמש יתבקש להזין שם משתמש חדש.

4. isFileExists

טענת כניסה: שם הקובץ לבדיקה, מטיפוס מחרוזת.

טענת יציאה: הפעולה מחזירה אמת אם הקובץ קיים או שקר אחרת.

תפקיד הפעולה:

משתמשים בפעולה זאת עבור 2 הקבצים שנמצאים על המכשיר במקומות שונים בפרויקט. בקובץ המכיל את שם המשתמש יש בדיקה האם המשתמש נרשם ע"י שימוש בפעולה זאת. בקובץ המכיל את הטוקן של האפליקציה של המשתמש יש בדיקה האם הקובץ קיים כאשר רוצים לכתוב את הטוקן לקובץ (על מנת לא לכתוב לקובץ ריק).

5. setToFile

טענת כניסה:

Filename – שם הקובץ שאליו נכתוב, מטיפוס מחרוזת.

Data – תוכן הקובץ החדש שאותו נרצה לכתוב.

טענת יציאה: אין

תפקיד הפעולה:

הפעולה כותבת את תוכן הקובץ שמועבר כפרמט לקובץ ששמו הועבר כפרמטר עבור 2 הקבצים במקומות שונים בפרויקט. לקובץ המכיל את שם המשתמש כאשר המשתמש מסיים את תהליך ההרשמה ולקובץ המכיל את הטוקן של המשתמש כאשר המשתמש מקבל אותו פעם ראשונה.

4. readFile

טענת כניסה:

Filename – מכיל את שם הקובץ לקריאה. מטיפוס מחרוזת.

טענת יציאה:

הפעולה מחזירה את תוכן הקובץ שהתקבל ששמו התקבל כפרמטר.

תפקיד הפעולה:

הפעולה קוראת את תוכן קובץ שם המשתמש במקומות שונים בפרויקט כאשר צריך את שם המשתמש ואת תוכן קובץ הטוקן כדי להשוות האם הטוקן הנוכחי של המשתמש שונה מזה שכתוב על הקובץ ובשרת.

5. isValidUser

טענת כניסה:

הפעולה מקבלת מחרוזת שהוא שם המשתמש לבדיקה.

טענת יציאה:

הפעולה מחזירה אמת או שקר האם שם המשתמש שהוזן קיים בשרת.

תפקיד הפעולה:

לוודא שלא יהיו כפילויות של שמות המשתמשים באפליקציה. הפעולה בודקת מול השרת באמצעות המחלקה המתאימה האם יש כפילויות בשם.

המחלקה `MyFirebaseMessagingService`

תפקיד המחלקה:

המחלקה היא יורשת מ-`Service` מותאם של `Google Firebase` (`FirebaseMessagingService`) המאפשר ניהול קבלת הודעות `push` מן `Google Firebase` והצגתן אל המשתמש.

תכונות המחלקה:

`NOTIFICATION_ID` - משתנה סטטי מטיפוס `שלם`. אחראי לשמירת המספר הסידורי של ההודעה.

פעולות המחלקה:

`onMessageReceived`

טענת כניסה:

`remoteMessage` – משנה מטיפוס `RemoteMessage` שהוא מבנה ההודעה המתקבל מגוגל פיירבייס.

טענת יציאה: אין

תפקיד הפעולה:

מגיעה דרך המחלקת האב של מחלקה זאת מאפשרת ניהול אירוע קבלת ההודעה בעת קבלתה.

generateNotification

טענת כניסה:

גוף ההודעה וכותרת ההודעה מטיפוס מחרוזת.

טענת יציאה: אין.

תפקיד הפעולה: ניהול הצגת ההודעה אל המשתמש בעת קבלת ההודעה.

[המחלקה LV.java players Pl](#)

תפקיד המחלקה: הצגת מסך בחירת השחקנים, בדיקה וניהול אירוע בחירת שחקן מהרשימה.

תכונות המחלקה:

Lv – הפקד שאחראי להצגת הרשימה. מטיפוס ListView

Index – מטיפוס מספר שלם. שומר את אינדקס השחקן שנבחר בעמוד הקודם.

player_adapter – משתנה מטיפוס PL_Player_Adapter שהיא מחלקת עזר העוזרת להקצות לכל איבר ברשימה layout בפקד lv.

arrCheckDup - מערך מספרים שלמים המכילים את המספרים המזהים של השחקנים מאותה עמדה של האינדקס שנבחר. משמש לבדיקת כפילויות של שחקן.

Sv מטיפוס SearchView הפקד שאחראי לחיפוש על הרשימה.

tmp1 שהיא רשימה של שחקני פרמייר ליג, רשימת שחקני הפרמייר ליג ותכונותיהן.

פעולות המחלקה:

onCreate

תפקיד הפעולה: טעינת המידע מהמסך הקודם בעת הופעתו, הצגת הרשימה של העמדה שנבחרה על המסך.

setQueryTextListener

תפקיד הפעולה:

מאזינה להקלדה של תיבת בחיפוש והצגת הרשימה בהתאם למידע שהוקלד.

setOnClickListener

תפקיד הפעולה:

הפעולה מאזינה ללחיצה על הרשימה, בודקת האם הבחירה היא חוקית: האם השחקן נבחר כבר והאם נשאר תקציב עבור השחקן.

המחלקה Splash screen

תפקיד המחלקה:

להציג את מסך הפתיחה למספר שניות מוגבל.

תכונות המחלקה: אין

פעולות המחלקה:

1. **onCreate**

תפקיד הפעולה: להציג את התמונה לחמש שניות ולעבור את למסך הבא. בפעולה קודם מוצג המסך ואז יופעל תהליך שנמשך 5 שניות ובסיומו יהיה מעבר למסך הבא.

צד שרת

יחידות מרכזיות ומדריך למפתח – מחלקות

D:\Noam\Programing\SchoolPrograming\yg2\fant...	35.9 MB
.idea	29.8 KB
__pycache__	14.1 KB
venv	35.8 MB
[9 Files]	21.1 KB
apiPrenierLeauge.py	2.3 KB
cloud_messaging.py	2.3 KB
getFromClient.py	4.1 KB
league.py	4.0 KB
my-fantasy-2020-firebase-adminsdk-n2h7g-a...	2.3 KB
teams.py	1.9 KB
tstDbSql.py	445 Bytes
tstjson.py	439 Bytes
users.py	3.4 KB

[המחלקה PremierLeague](#)

תפקיד המחלקה הוא לטפל בממשק מול הליגה האנגלית.

תכונות חשובות של המחלקה:

lastUpdated – אחראי לציון התאריך האחרון שבו נדגמו נתוני הליגה האנגלית, הזמן נבדק בכל פנייה למחלקה, במידה וחלפו 24 שעות מתבצעת פנייה לממש של הליגה האנגלית לעדכון הנתונים, מטיפוס תאריך.

Current_user_data – תוכן הפעם האחרונה שבו נדגם הממשק, בצורה מותאמת למשתמש, בצורת json.

All_data – תוכן הפעם האחרונה שבו נדגם הממשק בצורתו המלאה, בצורת json.

dict_points – מילון שהמפתח שלו הוא קוד השחקן בצורת מחרוזת והערך הוא מספר הנקודות שעשה במשחק.

פעולות המחלקה:

1. Bring_players

טענת כניסה: אין

טענת יציאה:

הפעולה מחזירה את המידע על כל השחקנים בליגה האנגלית כפי שהוא מגיע מהממשק, בפרומט json בצורת מחרוזת.

תפקיד הפעולה:

הפעולה פונה אל הממשק של הליגה האנגלית ומבקשת את המידע על כל השחקנים בליגה האנגלית, ומעדכנת את תאריך העדכון האחרון.

2. Calc11

טענת כניסה:

הפעולה מקבלת רשימה של 11 מספרים המזהים של השחקנים בליגה האנגלית שנבחרות לקבוצה מסוימת, ומחשבת את הניקוד הכולל של השחקנים במשחק דרך המילון שמוגדר במחלקה.

טענת יציאה:

הפעולה מחזירה את הסכום של הניקוד של כל השחקנים ברשימה, בצורת מספר שלם.

תפקיד הפעולה:

הפעולה מחשבת את הסכום של כל השחקנים ברשימה כאשר נוספת קבוצה חדשה למבנה הנתונים ביחד עם הניקוד הכולל של הקבוצה.

3. bringForUser

טענת כניסה: אין

טענת יציאה:

הפעולה מחזירה את השחקנים במשחק והמאפיינים שלהם שהמשתמש צריך, בפורמט json בצורת מחרוזת.

תפקיד הפעולה:

הפעולה מחזירה את השחקנים במשחק והמאפיינים שלהם שהמשתמש צריך כאשר המשתמש צריך לבחור שחקנים לקבוצה שלו. במידה ועברו 24 שעות מהפעם האחרונה שהממשק של הפרמייר ליג הוא יידגם שוב, אחרת יוחזרו המידע ששמור אצל השרת.

bringPlayerPhoto .4

טענת כניסה: אין

טענת יציאה:

הפעולה מחזירה את ה-url המשמשת להשגת תמונה של שחקן עם קוד מסוים דרך ה-api של הפרמייר ליג.

תפקיד הפעולה:

הפעולה מחזירה את ה-url המשמשת להשגת תמונה של שחקן כאשר המשתמש בוחר שחקן התמונה שלו מופיעה ברשימת השחקנים וכששחקן מסוים נבחר תמונתו מופיעה במסך בניית הקבוצה.

bringTeamPhoto .5

טענת כניסה: אין

טענת יציאה:

הפעולה מחזירה את ה-url המשמשת להשגת תמונה של קבוצה עם קוד מסוים דרך ה-api של הפרמייר ליג.

תפקיד הפעולה:

הפעולה מחזירה את ה-url המשמשת להשגת תמונה של קבוצה כאשר המשתמש בוחר שחקן התמונה של קבוצתו מופיעה ברשימת השחקנים, מטיפוס מחרוזת.

Update_dict .6

טענת כניסה: אין

טענת יציאה: אין

תפקיד הפעולה:

פעולה פנימית של המחלקה, הפעולה מעדכנת את המילון שהמפתח שלו הוא קוד השחקן בצורת מחרוזת והערך הוא מספר הנקודות שעשה במשחק בכל פעם שה-api של הפרמייר ליג נדגם מחדש.

[המחלקה NotifyClient](#)

תפקיד המחלקה להתריע משתמשים שהזמנו לליגות דרך שליחת הודעה מתאימה, תוך שימוש בספרייה של Google Firebase ושמה `firebase_admin`.

תכונות חשובות של המחלקה: אין.

פעולות:

1. `Send_multicast`

טענת כניסה:

`Registration_tokens` – רשימה של טוקנים בצורת מחרוזות של משתמשים הקיימים במערכת שהוזמנו לליגה מסוימת.

`League_owner` – שם המשתמש של יוצר הליגה החדשה, מטיפוס מחרוזת.

`League_name` – שם הליגה החדשה, מטיפוס מחרוזת.

טענת יציאה: אין

תפקיד הפעולה: הפעולה מתריעה למשתמשים שהוזמנו דרך הודעה מותאמת למכשירי android דרך Google Firebase.

2. `android_message`

טענת כניסה:

`Registration_tokens` – רשימה של טוקנים בצורת מחרוזות של משתמשים הקיימים במערכת שהוזמנו לליגה מסוימת.

`League_owner` – שם המשתמש של יוצר הליגה החדשה, מטיפוס מחרוזת.

`League_name` – שם הליגה החדשה, מטיפוס מחרוזת.

טענת יציאה:

הפעולה מחזירה את תבנית של הודעה שמותאמת למכשירי android על פי הפרמטרים שהתקבלו.

תפקיד הפעולה: הפעולה מתריעה למשתמשים שהוזמנו דרך הודעה מותאמת למכשירי android דרך Google Firebase.

המחלקה Leauge

תפקיד המחלקה:

המחלקה מטפלת באירועים הקשורים לליגות במשחק.

תכונות המחלקה: אין.

פעולות המחלקה:

1. `add_league`

טענת כניסה:

`dbConnection` – אחראי על החיבור עם הטבלאות במבנה הנתונים.

`username` – שם המשתמש של יוצר הליגה, מטיפוס מחרוזת.

`league_name` – שם הליגה שהמשתמש רוצה להוסיף, מטיפוס מחרוזת.

`teamName` – שם הקבוצה שהמשתמש מצרף לליגה, מטיפוס מחרוזת.

`teamList` – רשימת השחקנים הנמצאים בקבוצה של יוצר הליגה, רשימת שלמים.

`usersList` – רשימת המשתמשים המוזמנים לליגה, מיטפוס רשימת של מחרוזות.

טענת יציאה: הפעולה מחזירה אמת כאשר תהליך הוספת הליגה למבנה הנתונים הסתיים.

תפקיד הפעולה: הפעולה מוסיפה את הליגה החדשה למבנה הנתונים, מזמנת את הפעולה המוסיפה את הקבוצה של יוצר הליגה למבנה הנתונים.

2. `get_my_leagues`

טענת כניסה:

Username – שם המשתמש שמבקש את הליגות שהוא יכול להשתתף בהן.

טענת יציאה:

הפעולה מחזירה את הליגות שהמשתמש יכול להשתתף בהן ופרטיותן בפורמט json בצורת מחרוזת.

תפקיד הפעולה:

הפעולה מחזירה את כל הליגות שהמשתמש יצר או שהוזמן אליהן על מנת שהמשתמש יהיה יכול להיענות לליגות שהוזמן אליהן או לראות את מצב המשתמשים שהמשתמש הזמין לליגות שיצר.

המחלקה Teams

תפקיד המחלקה:

המחלקה מטפלת באירועים הקשורים לקבוצות במשחק.

תכונות המחלקה: אין.

פעולות המחלקה:

add_team .1

טענת כניסה:

dbConnection – אחראי על החיבור עם הטבלאות במבנה הנתונים.

Id_user – המספר המזהה של המשתמש שיצר את הליגה, מטיפוס מספר שלם.

Team_name- שם הקבוצה שהמשתמש רוצה להוסיף למערכת, מטיפוס מחרוזת.

Id_league – המספר המזהה של הליגה שאליה המשתמש מצרף את הקבוצה שלו.

Starting11_codes – רשימת המספרים המזהים של השחקנים שנבחרו על ידי המשתמש, מטיפוס מספרים שלמים.

טענת יציאה: אין

תפקיד הפעולה:

הפעולה מוסיפה את הקבוצה החדשה למבנה הנתונים ואת השחקנים השייכים לקבוצה.

2. Calc11

טענת כניסה:

רשימת השחקנים שהמשתמש בחר לקבוצה שלו, מטיפוס רשימת שלמים.

טענת יציאה:

סכום הניקוד של השחקנים בקבוצה, מטיפוס מספר שלם.

תפקיד הפעולה:

חישוב הניקוד הכולל של הקבוצה.

3. getPlayers

טענת כניסה:

dbConnection – אחראי על החיבור עם הטבלאות במבנה הנתונים.

idTeam – המספר המזהה של הקבוצה שרוצים את השחקנים שלה

טענת יציאה:

הפעולה מחזירה רשימת מספרים שלמים של השחקנים ששייכים לקבוצה.

תפקיד הפעולה:

הפעולה מחזירה את רשימת השחקנים השייכים לקבוצה כאשר משתמש רוצה לצפות באחת הקבוצות הנמצאות בליגות שלו.

[המחלקה users](#)

תפקיד המחלקה:

המחלקה מנהלת את הפעולות הקשורות אצל המשתמשים.

תכונות המחלקה: אין

פעולות המחלקה:

1. add_user

טענת כניסה:

Name – שם המשתמש החדש שנוסף למערכת.

dbConnection – אחראי על החיבור עם הטבלאות במבנה הנתונים.

Token – הטוקן של המשתמש החדש מול Google Firebase

טענת יציאה: הפעולה מחזירה אמת כאשר תהליך ההוספה הושלם.

תפקיד הפעולה:

הפעולה מוסיפה את המשתמש החדש למערכת.

2. nameTold

טענת כניסה:

dbConnection – אחראי על החיבור עם הטבלאות במבנה הנתונים.

Name – שם המשתמש שנרצה להמיר את שמו למספר המזהה שלו.

טענת יציאה:

הפעולה מחזירה את המספר המזהה של המשתמש שהתקבל כפרמטר.

תפקיד הפעולה: להמיר את שם המשתמש למספר המזהה במקרה הצורך.

3. idToName

טענת כניסה:

dbConnection – אחראי על החיבור עם הטבלאות במבנה הנתונים.

idUser – המספר המזהה של המשתמש שאותו נרצה להמיר.

טענת יציאה:

הפעולה מחזירה את שם המשתמש שבעל המספר המזהה שהתקבל כפרמטר.

תפקיד הפעולה:

הפעולה ממירה את המספרים המזהים לשמות משתמשים במקומות שונים בתכנית

4. getOthers

טענת כניסה:

dbConnection – אחראי על החיבור עם הטבלאות במבנה הנתונים.

Name – שם המשתמש שנרצה להמיר את שמו למספר המזהה שלו.

טענת יציאה:

הפעולה מחזירה את רשימת המשתמשים ששמן לא שווה לשם המשתמש שהתקבל כפרמטר.

תפקיד הפעולה:

הפעולה מחזירה את רשימת המשתמשים ששמן לא שווה לשם המשתמש שהתקבל כפרמטר כאשר המשתמש צריך לבחור אילו משתמשים הוא רוצה להזמין לליגה שעשה.

טענת כניסה:

Username- שם המשתמש שאליו נרצה לעדכן את ה-token, מטיפוס מחרוזת.

New_token- ה-token החדש שאותו נרצה לשים במקום ה-token הקיים של אותו המשתמש, מטיפוס מחרוזת.

טענת יציאה:

הפעולה שולחת הודעה מתאימה כאשר תהליך העדכון התסיים.

תפקיד הפעולה:

הפעולה מעדכנת את ה-token החדש שאותו נרצה לשים במקום ה-token הקיים של אותו המשתמש. משתמשים בפעולה זאת כאשר המשתמש מקבל token חדש מול Google Firebase על מנת שיוכל לקבל התראות.

5. getTokenByUser

טענת כניסה:

dbConnection – אחראי על החיבור עם הטבלאות במבנה הנתונים.

lstUsers – רשימה של שמות משתמשים, מטיפוס מחרוזות.

טענת יציאה:

הפעולה מחזירה רשימה של tokens, בצורת מחרוזת.

תפקיד הפעולה:

הפעולה מחזירה רשימה של tokens, בצורת מחרוזת כאשר צריך להודיע לכל המשתמשים שהוזמנו לליגה חדשה.

תיאור פרוטוקולי תקשורת

שרת לקוח

המחלקה `getFromClient`

תפקיד מחלקה זו הוא טיפול בפניות הלקוח לצרכים שונים. הפניות ממומשות על ידי הפעולות השונות במחלקה. הפניית הבקשות לפעולות השונות נעשית באמצעות השירות `.flask`.

להלן פירוט הפעולות:

1. `editToken`

טענת כניסה:

Username- שם המשתמש שאליו נרצה לעדכן את ה-token, מטיפוס מחרוזת.

New_token- ה-token החדש שאותו נרצה לשים במקום ה-token הקיים של אותו המשתמש, מטיפוס מחרוזת.

טענת יציאה:

הפעולה שולחת הודעה מתאימה כאשר תהליך העדכון התסיים.

תפקיד הפעולה:

הפעולה מעדכנת את ה-token החדש שאותו נרצה לשים במקום ה-token הקיים של אותו המשתמש. משתמשים בפעולה זאת כאשר המשתמש מקבל token חדש מול Google Firebase על מנת שיוכל לקבל התראות.

דוגמה לפנייה:

<http://192.168.1.18:5000/editToken/Yossi>

[Levi/fgekb5JJRS8:APA91bGKyO3oiCU8FeCqzDPRqdItN6b1u93a18bW5CCZqDUg1u6zeNJaDnF0IXfVpDk85erPs-](#)

[Az9si8KDHZOwjR7unRu_5bCYuvBWh3Hv_Fi8tGxgz7oHlnEV8lwl8G2Qr2srGCSNi](#)

דוגמה לתשובה אפשרית:

Updated

2. isUsernameExists

טענת כניסה:

Username - הפעולה מקבלת שם משתמש מסוים, מטיפוס מחרוזת.

טענת יציאה:

הפעולה מחזירה הודעה "valid name" אם שם המשתמש שהתקבל לא קיים במבנה הנתונים, אחרת מוחזר "name already exists" אם השם קיים במערכת.

תפקיד הפעולה:

משתמשים בפעולה בתהליך ההרשמה כאשר שם המשתמש של הנרשם צריך להיות ייחודי ולא קיים כבר במערכת.

דוגמה לפנייה:

<http://192.168.1.18:5000/isUsernameExists/Moshe>

דוגמה לתשובה אפשרית:

valid name

3. addUser

טענת כניסה:

Username - שם משתמש שאינו קיים במערכת, מטיפוס מחרוזת.

Token – מזהה של האפליקציה של המשתמש מול Google Firebase, מטיפוס מחרוזת.

טענת יציאה:

הפעולה מחזירה הדעה מתאימה במידה ונוסף למערכת.

תפקיד הפעולה:

הפעולה מוסיפה משתמש חדש למבנה הנתונים לאחר שסיים להירשם.

דוגמה לפניה:

<http://192.168.1.18:5000/addUser/Moshe/token123>

תשובה אפשרית:

added

4. getUserLeagues

טענת כניסה:

שם משתמש הקיים במערכת, מטיפוס מחרוזת.

טענת יציאה:

הפעולה מחזירה את כל הליגות שהמשתמש יצר או שהוזמן אליהן בצורת Json.

תפקיד הפעולה:

הפעולה מחזירה את כל הליגות שהמשתמש יצר או שהוזמן אליהן על מנת שהמשתמש

יהיה יכול להיענות לליגות שהוזמן אליהן או לראות את מצב המשתמשים שהמשתמש

הזמין לליגות שיצר.

דוגמה לפנייה:

<http://192.168.1.18:5000/getMyLeagues/Yossi Levi>

תשובה אפשרית:

```
{
  "leagues": [
    {
      "name": "chipopo I",
      "idLeague": 50,
      "idOwner": 75,
      "creator_username": "Yossi Levi",
      "teams": [
        {
          "idTeam": 56,
          "idOwner": 75,
          "teamName": "wow its bad",
          "usernameOwner": "Yossi Levi",
          "points": 792
        },
        {
          "idTeam": 57,
          "idOwner": 74,
          "teamName": "real madirz",
          "usernameOwner": "Moshe Cohen",
          "points": 1352
        }
      ]
    },
    {
      "name": "IIII",
      "idLeague": 51,
      "idOwner": 75,
      "creator_username": "Yossi Levi",
      "teams": [
        {
          "idTeam": 58,
          "idOwner": 75,
          "teamName": "dhfhf",
          "usernameOwner": "Yossi Levi",
          "points": 567
        }
      ]
    },
    {
      "name": "ggg",
      "idLeague": 52,
      "idOwner": 75,
      "creator_username": "Yossi Levi",
      "teams": [
        {
          "idTeam": 59,
          "idOwner": 75,
          "teamName": "fff",
          "usernameOwner": "Yossi Levi",
          "points": 839
        }
      ]
    },
    {
      "name": "III",
      "idLeague": 53,
      "idOwner": 75,
      "creator_username": "Yossi Levi",
      "teams": [
        {
          "idTeam": 60,
          "idOwner": 75,
          "teamName": "yyy",
          "usernameOwner": "Yossi Levi",
          "points": 519
        }
      ]
    },
    {
      "name": "hjh",
      "idLeague": 54,
      "idOwner": 75,
      "creator_username": "Yossi Levi",
      "teams": [
        {
          "idTeam": 61,
          "idOwner": 75,
          "teamName": "ttt",
          "usernameOwner": "Yossi Levi",
          "points": 918
        }
      ]
    },
    {
      "name": "nnn",
      "idLeague": 55,
      "idOwner": 75,
      "creator_username": "Yossi Levi",
      "teams": [
        {
          "idTeam": 62,
          "idOwner": 75,
          "teamName": "ghh",
          "usernameOwner": "Yossi Levi",
          "points": 704
        }
      ]
    },
    {
      "name": "nnn",
      "idLeague": 56,
      "idOwner": 75,
      "creator_username": "Yossi Levi",
      "teams": [
        {
          "idTeam": 63,
          "idOwner": 75,
          "teamName": "uuu",
          "usernameOwner": "Yossi Levi",
          "points": 913
        }
      ]
    }
  ]
}
```

5. addLeague

טענת כניסה:

Username - שם משתמש הקיים במערכת, מטיפוס מחרוזת.

leagueName – שם הליגה שאותה המשתמש רוצה להוסיף למערכת מטיפוס מחרוזת.

teamName – שם הקבוצה שהמשתמש מצרף לליגה, מטיפוס מחרוזת.

teamList – רשימת השחקנים הנמצאים בקבוצה של יוצר הליגה, רשימת שלמים.

usersList – רשימת המשתמשים המוזמנים לליגה, מטיפוס רשימת של מחרוזות.

טענת יציאה:

הפונקציה שולחת "added" כאשר תהליך הוספת הקבוצה לליגה הושלם.

תפקיד הפעולה:

הפעולה מוסיפה את הליגה החדשה למבנה הנתונים, את המשתמשים שהוזמנו לליגה, ומתריעה למשתמשים שהוזמנו שייענו לבקשה.

דומה לפנייה:

<http://192.168.1.18:5000/addLeague/Yossi>
[Levi/bestLeague/myEleven/\[\"116535\", \"74230\", \"172850\", \"110735\", \"41328\", \"114283\", \"159533\", \"47247\", \"61366\", \"102057\", \"54694\"\]/\[\"Moshe Cohen\"\]](http://192.168.1.18:5000/addLeague/Levi/bestLeague/myEleven/[\)

תשובה אפשרית:

Added

6. addTeamToLeague

טענת כניסה:

Username - שם המשתמש שמוסיף את הקבוצה שלו.
 leagueId – המספר המזהה של הליגה שמשתמש מוסיף את הקבוצה, מטיפוס שלם.
 teamName – שם הקבוצה שהמשתמש רוצה להוסיף, מטיפוס מחרוזת.
 teamList - רשימת השחקנים הנמצאים בקבוצה של יוצר הליגה, רשימת שלמים.

טענת יציאה:

הפונקציה שולחת "added" כאשר תהליך הוספת הקבוצה לליגה הושלם.

תפקיד הפעולה:

הפעולה מוסיפה את הקבוצה החדשה למבני הנתונים, ומחשבת את הניקוד שלה.

דוגמה לפנייה:

[http://192.168.1.18:5000/addTeamToLeague/Noam/56/joventus/\[\"116535\",\"74230\",\"172850\",\"110735\",\"41328\",\"114283\",\"159533\",\"47247\",\"61366\",\"102057\",\"54694\"\]](http://192.168.1.18:5000/addTeamToLeague/Noam/56/joventus/[\)

7. getPLPlayers

טענת כניסה: אין

טענת יציאה:

הפעולה מחזירה את רשימת השחקנים מהפרמייר ליג ומאפייניהם בצורת json.

תפקיד הפעולה:

הפעולה מחזירה את רשימת השחקנים במשחק ומאפייניהם כאשר המשתמש בונה קבוצה וצריך לבחור שחקנים לקבוצה שלו.

דוגמה לפנייה:

<http://192.168.1.18:5000/getPLPlayers>

תשובה אפשרית:

```
"}plPlayers": [{"fullName": "Shkodran Mustafi", "position": 2, "cost": 51, "code": 69140, "team": 3}, {"fullName": "Hector Bellerin", "position": 2, "cost": 55, "code": 98745, "team": 3}, {"fullName": "Sead Kolasinac", "position": 2, "cost": 52, "code": 111457, "team": 3}, {"fullName": "Ainsley Maitland-Niles", "position": 2, "cost": 45, "code": 154043, "team": 3}]}
```

8. getPlayerPhoto

טענת כניסה: אין

טענת יציאה:

הפעולה מחזירה את ה-url המשמשת להשגת תמונה של שחקן עם קוד מסוים דרך ה-api של הפרמייר ליג.

תפקיד הפעולה:

הפעולה מחזירה את ה-url המשמשת להשגת תמונה של שחקן כאשר המשתמש בוחר שחקן התמונה שלו מופיעה ברשימת השחקנים וכששחקן מסוים נבחר תמונתו מופיעה במסך בניית הקבוצה.

דוגמה לפניה:

<http://192.168.1.18:5000/getPlayerPhoto>

תשובה אפשרית:

<https://platform-static-files.s3.amazonaws.com/premierleague/photos/players/110x140/p152760.png>

9. getTeamPhoto

טענת כניסה: אין

טענת יציאה:

הפעולה מחזירה את ה-url המשמשת להשגת תמונה של קבוצה עם קוד מסוים דרך ה-api של הפרמייר ליג.

תפקיד הפעולה:

הפעולה מחזירה את ה-url המשמשת להשגת תמונה של קבוצה כאשר המשתמש בוחר שחקן התמונה של קבוצתו מופיעה ברשימת השחקנים, מטיפוס מחרוזת.

דוגמה לפניה:

<http://192.168.1.18:5000/getTeamPhoto>

תשובה אפשרית:

https://fantasy.premierleague.com/dist/img/badges/badge_14_40.png

10. getOtherUsers

טענת כניסה:

You- שם משתמש מטיפוס מחרוזת

טענת יציאה:

הפעולה מחזירה את המשתמשים האחרים במערכת בצורת רשימת מחרוזות.

תפקיד הפעולה:

הפעולה מחזירה את המשתמשים האחרים במערכת כאשר המשתמש צריך להזמין משתמשים

אחרים לליגה שהוא יוצר.

דוגמה לפנייה:

<http://192.168.1.18:5000/getOtherUsers/Noam>

תשובה אפשרית:

["Moshe Cohen", "Yossi Levi"]

11. getSomeoneTeam

טענת כניסה:

idTeam – מספר הקבוצה המבוקשת, מטיפוס מספר שלם.

טענת יציאה:

הפעולה מחזירה את המספרים המזהים של השחקנים השייכים לקבוצה שהתקבלה כפרמטר בצורת רשימת מספרים שלמים.

תפקיד הפעולה:

הפעולה מחזירה את המספרים המזהים של השחקנים השייכים לקבוצה כאשר רוצים לצפות אילו שחקנים שייכים לקבוצה מסוימת באחת הליגות שהמשתמש משתתף בהן.

דוגמה לפנייה:

<http://127.0.0.1:5000/getSomeoneTeam/60>

תשובה אפשרית:

[8432, 223911, 17761, 69140, 111457, 152551, 89274, 148508, 40564, 54694, 94245]

הממשק מול הליגה האנגלית

על מנת להשיג מידע על נתוני הפאנטזי של הליגה האנגלית נדרש לפנות לשרות REST של הפרמייר ליג.

בפרויקט זה יש שימוש בשלושה סוגים של גישות למבנה הנתונים של הליגה האנגלית.

השגת רשימת השחקנים מהעונה האחרונה והמידע עליהם:

כאשר המשתמש בונה קבוצה, הוא צריך לבחור שחקן מרשימת השחקנים. הוא בוחר עמדה, ורשימת השחקנים בעמדה המבוקשת מופיעה במסך רשימת השחקנים. השרת דוגם את מבנה הנתונים במקרה זה באמצעות REST לפי התבנית הבאה:

<https://fantasy.premierleague.com/api/bootstrap-static/>

אתר הליגה האנגלית מחזיר את רשימת השחקנים בפורמט Json והשרת מחזיר את המידע באותו אופן עם הפרטים הנחוצים למשתמש.

השגת התמונה של השחקנים:

כאשר המשתמש בוחר בעמדה מסוימת במסך בניית הקבוצה, תמונות השחקנים מופיעות במסך רשימת השחקנים. לאחר בחירת שחקן תמונתו תופיע במסך בניית הקבוצה בעמדה שבחר קודם לכן. השרת מחזיר את הכתובת בתור מחרוזת.

<https://platform-static-files.s3.amazonaws.com/premierleague/photos/players/110x140/p{{code}}.png>

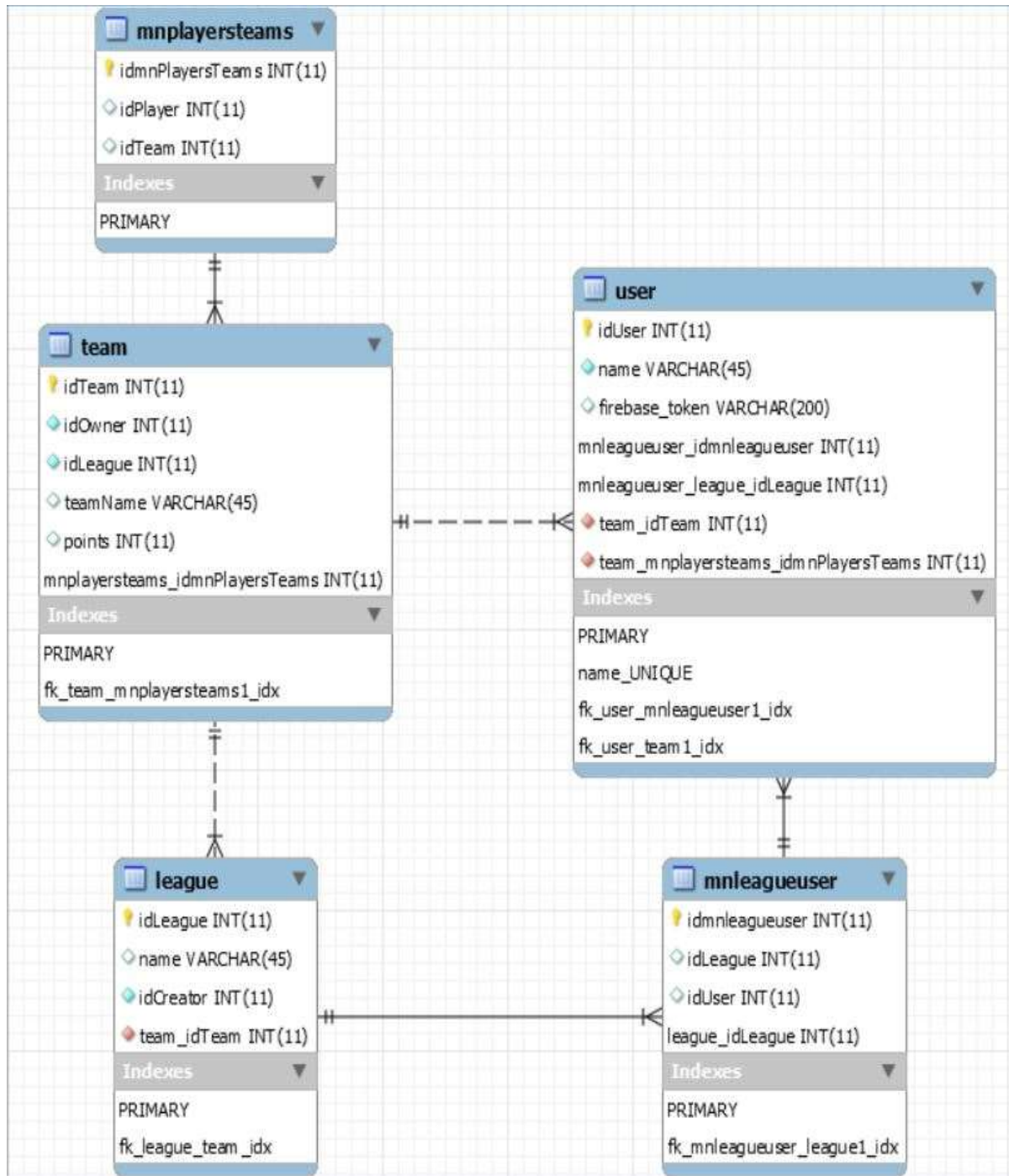
השגת הלוגואים של הקבוצות:

כאשר המשתמש בוחר בעמדה מסוימת במסך בניית הקבוצה, הסמלים של קבוצות השחקנים יופיעו במסך בחירת השחקנים. השרת מחזיר את הכתובת בתור מחרוזת.

https://fantasy.premierleague.com/dist/img/badges/badge_{{team_code}}_40.png

מבנה הנתונים

תרשים ERD



הטבלאות בבסיס הנתונים

טבלת משתמש

שם – user.

תיאור – טבלה זאת משמשת לשמור את נתוני המשתמשים הרשומים לאפליקציה.

עמודות –

א. שם – idUser

תיאור – המזהה המספרי של כל משתמש.

טיפוס – int

תכונות:

האם מפתח ראשי – כן

האם יכול להיות null – לא

Auto Increasment – כן

ב. שם – name

תיאור – שם המשתמש הייחודי של כל משתמש.

טיפוס – VARCHAR

תכונות:

האם מפתח ראשי – לא

האם יכול להיות null – לא

ג. שם – firebase_token

תיאור – המזהה של האפליקציה של כל משתמש מול Google Firebase.

טיפוס – VARCHAR

תכונות:

האם מפתח ראשי – לא

האם יכול להיות null – לא

טבלת קבוצה

שם - team

תיאור – טבלה זו שומרת את נתוני הקבוצות הנבנות בתוך ליגות.עמודות –

א. שם – idTeam

תיאור – המזהה המספרי של כל קבוצה.טיפוס – intתכונות:האם מפתח ראשי – כןהאם יכול להיות null – לאAuto Increasment – כן

ב. שם – idOwner

תיאור – מכיל את המזהה של המשתמש שבנה את הקבוצהטיפוס – intתכונות:האם מפתח ראשי – לא. יכול להיות שמשתמש אחד בנה מספר קבוצות.האם יכול להיות null – לא

ג. שם – idLeague

תיאור – מכיל את המזהה של הליגה שבה הקבוצה נבנתה.טיפוס – intתכונות:האם מפתח ראשי – לאהאם יכול להיות null – לא

ד. שם – teamName

תיאור – מכיל את שם הקבוצה.

טיפוס – VARCHAR

תכונות:

האם מפתח ראשי – לא

האם יכול להיות null – כן (מתבצעת בדיקת תקינות של שם הקבוצה בצד הלקוח כך שהשם לא יהיה null).

ה. שם – points

תיאור – יכיל את מספר הנקודות של הקבוצה.

טיפוס – int

תכונות:

האם מפתח ראשי – לא

Auto Increasment – לא

טבלת ליגה

שם – league

תיאור – מכילה את פרטי הליגות הנבנות על ידי המשתמשים.

עמודות –

א. שם – leaugeld

תיאור – מכיל את המספר המזהה של הליגה .

טיפוס – int

תכונות:

האם מפתח ראשי – כןהאם יכול להיות null – לאAuto Increasment – כן

ב. שם – name

תיאור – מכיל את שם הליגה.

טיפוס – VARCHAR

תכונות:

האם מפתח ראשי – לאהאם יכול להיות null – כן (מתבצעת בדיקת תקינות של שם הקבוצה בצד הלקוח כך שהשם לא יהיה null).

ג. שם – idCreator

תיאור – מכיל את המספר המזהה של המשתמש שיצר את הליגה

טיפוס – int

תכונות:

האם מפתח ראשי – לאהאם יכול להיות null – לא

טבלת שחקן - קבוצה

שם – mnplayersteams

תיאור – שמירה של השחקנים שנבחרו על ידי הקבוצות תוך שמירה על עקרונות הנרמול, מאחר שאותו שחקן יכול להיות שייך לכמה קבוצות ובאותה קבוצה יש 11 שחקנים.

עמודות –

א. שם – idmnplayersteams

תיאור - מזהה ייחודי לשורה בטבלה.תכונות:האם מפתח ראשי – כןהאם יכול להיות null – לאAuto Increasment – כן

ב. שם – idPlayer

תיאור – מכיל את קוד השחקן שמוענק לשחקן לפי ה-API של הליגה האנגלית בכדורגל.

תכונות:האם מפתח ראשי – לאהאם יכול להיות null – לא

ג. שם – idTeam

תיאור – מכיל את מזהה הקבוצה שאליה שייך השחקן.תכונות:האם מפתח ראשי – לאהאם יכול להיות null – לא

טבלת משתמש - ליגה

שם - mnleagueuser

תיאור – שמירה של הליגות שבהן נמצאות המשתמשים תוך שמירה על עקרונות הנרמול. מאחר ומשתמש אחד יכול להיות במספר ליגות ובליגה אחת יש מספר משתמשים.

עמודות –

א. שם - idmnleagueuser

תיאור - מזהה ייחודי לשורה בטבלה.תכונות:האם מפתח ראשי – כןהאם יכול להיות null – לאAuto Increment – כן

ב. שם - idLeague

תיאור – מכיל את המזהה של הליגה שאליו שייך המשתמש.טיפוס – intתכונות:האם מפתח ראשי – לאהאם יכול להיות null – לא

ג. שם - idUser

תיאור – מכיל את המספר המזהה של המשתמש ששייך לליגה.טיפוס – intתכונות:האם מפתח ראשי – לאהאם יכול להיות null – לא

שאלות

שאלתה המוסיפה משתמש חדש לטבלת משתמשים.

השאלתה מקבלת את שם המשתמש החדש ביחד עם המזהה של אפליקציית המשתמש עם Google Firebase .

```
insert into my_fantasy_pl.user (name, firebase_token) VALUES
("Noam","token123");
```

שאלתה המחזירה האם קיים משתמש עם שם משתמש מסוים.

השאלתה מקבלת את שם המשתמש לבדיקה ומחזירה את רשימת המשתמשים עם אותו שם משתמש. בשאלתה זו יש שימוש בתהליך ההרשמה כאשר יש בדיקה האם שם המשתמש שהוכנס כבר קיים במערכת. אם לא קיים, ניתן להמשיך את תהליך ההרשמה.

```
select * from my_fantasy_pl.user where name = "Noam"
```

שאלתה שממירה שם משתמש למספר מזהה. השאלתה מקבלת שם משתמש ומחזירה את המספר המזהה של המשתמש.

```
select my_fantasy_pl.user.idUser from my_fantasy_pl.user where name = "Noam"
```

שאלתה הממירה מספר מזהה לשם משתמש. השאלתה מקבלת מספר מזהה של משתמש ומחזירה את שם המשתמש.

```
Select user.name from user where idUser = 30;
```

שאלתה המחזירה את המשתמשים האחרים במשחק שבעלי שם משתמש שונה ממשתמש מסוים. השאלתה מקבלת שם משתמש ומחזירה את המשתמשים האחרים במשחק.

```
select name from user where name != "Noam";
```

שאיילתה המעדכנת token של משתמש בטבלת המשתמשים.

השאיילתה מקבלת שם משתמש וגם token חדש ומעדכנת token חדש.

```
UPDATE user SET user.firebase_token = "token456" WHERE user.name = "Noam" ;
```

שאיילתה המקבלת token ומחזירה את שם המשתמש שאליו שייך ה-token.

1. שאיילתה זו משמשת לשלוח הודעה למשתמש המתאים שהוזמן לליגה חדשה.

השאיילתה מקבלת שם משתמש ומחזירה את המזהה של האפליקציה שלו מול Google
Firebase.

```
select firebase_token from user where name = "Noam";
```

שאיילתה המקבלת קבוצה ומוסיפה אותה לטבלת הקבוצות.

השאיילתה מקבלת את שם המשתמש של בונה הקבוצה, מספר הליגה שאליה שייכת
הקבוצה ושם הקבוצה.

```
INSERT INTO `my_fantasy_pl`.`team` (`idOwner`, `idLeague`, `teamName`, `points`)  
VALUES (30, 40, "Good Team F.C" , 751);
```

שאיילתה המוסיפה קוד של שחקן השייך לקבוצה לטבלת שחקנים – קבוצות

```
insert into mnplayersteams(idPlayer,idTeam) values(75624,68);
```

שאיילתה המחזירה את קודי השחקנים השייכים לקבוצה מסוימת.

השאיילתה מקבלת מספר מזהה של קבוצה ומחזירה את הקודים של השחקנים
בקבוצה.

```
select idPlayer from my_fantasy_pl.mnplayersteams where  
mnplayersteams.idTeam= 30;
```

שאילתה המוסיפה ליגה חדשה לטבלת ליגות.

השאילתה מקבלת את שם הליגה ואת המספר המזהה של יוצר הליגה.

```
INSERT INTO `my_fantasy_pl`.`league` (`name`, `idCreator`) VALUES ("Winners",3);
```

שאילתה המוסיפה משתמש לליגה באמצעות הוספתו לטבלת משתמשים – ליגות.

הליגה מקבלת מספר מזהה של משתמש ומספר מזהה של ליגה ומוסיפה אותו לטבלה.

```
insert into my_fantasy_pl.mnleagueuser(idLeague,idUser) values(30, 40);
```

שאילתה המחזירה את פרטי הליגות שבהן משתתף משתמש מסוים.

השאילתה מקבלת מספר מזהה של משתמש ומחזירה את פרטי הליגות שבהן רשום המשתמש.

```
select * from (select tbl.idLeague,tbl.idUser,league.name,league.idCreator from
(select * from my_fantasy_pl.mnleagueuser where idUser = 74) as tbl inner join
league on tbl.idLeague = league.idLeague) as sub_q inner join user on
sub_q.idCreator = user.idUser;
```

שאילתה המחזירה את פרטי המשתמשים שהספיקו לבנות קבוצה בליגה שאליה הוזמן המשתמש.

השאילתה מקבלת מספר ליגה ומחזירה את פרטי המשתמשים.

```
select * from team inner join user on team.idOwner = idUser where idLeague = 45;
```

טכנולוגיות מרכזיות בפרויקט

FLASK

הסבר כללי:

זוהי ספרייה בשפת פייתון שעוזרת לממש פעולות Rest תחת http. הספרייה עוזרת לנתח את כתובת ה-URL שהתקבלה ויודעת לחלק אותה לפי פעולות שכל אחת מטפלת באופן שונה בהתאם למידע שהתקבל מהלקוח.

שימוש בפרויקט:

בצד השרת יש שימוש בספרייה זו במחלקה `getFromClient` כאשר הפעולות עוזרות לנתח את המידע שהתקבל מהלקוח, עוזרת לדלות מהמידע את הפעולה הדרושה לביצוע ואת הפרמטרים.

FIREBASE CLOUD MESSAGINGS

הסבר כללי:

זהו שירות של חברת גוגל המאפשר שליחת הודעות אל אפליקציות במכשירים השונים הנרשמים אליה.

שימוש בפרויקט:

ראשית, בצד הלקוח מוקצה מזהה ייחודי של האפליקציה של המשתמש מול גוגל פיירבייס ומועבר בנוסף גם אל השרת ונשמרים עם פרטי הלקוח בעת ההרשמה.

לאחר מכן, בצד השרת נשלחת הודעה למשתמשים שהוזמנו על ידי משתמש שיצר ליגה חדשה באמצעות שירות זה. ובצד הלקוח יש מחלקה הדואגת לקבל את המידע, לנתח ולהציג אותו בפני המוזמנים שייענו לליגה שאליה הוזמנו.

MySQL

הסבר כללי:

זהו מסד נתונים טבלאי המאפשר אחסון מידע.

שימוש בפרויקט:

הנתונים המוזנים מהמשתמשים השונים במשחק נשמרים באמצעות MySQL. ראה פרק מסד נתונים.

Glide

הסבר כללי:

זוהי ספרייה המאפשרת הטענת תמונות לאפליקציות אנדרואיד מתוך כתובת URL או מתוך תמונה קיימת הנמצאת על המכשיר.

שימוש בפרויקט:

בצד הלקוח, כדי לטעון את התמונות של השחקנים והקבוצות מהליגה האנגלית יש שימוש בספרייה זו באמצעות העברת הכתובת שבה נמצאת התמונה ברשת והפקד שבה התמונה תהיה מוכלת.

OkHttp

הסבר כללי:

זוהי ספרייה העוזרת לייצג את הלקוח בתקשורת מול שרת http מסוים בקריאה לשירותי REST – היא עוזרת לו לשלוח את הבקשה לפי כתובת מסוימת ועוזרת לו לנתח את התשובה המוחזרת ממנו.

שימוש בפרויקט:

בצד הלקוח, באפליקציה, יש שימוש בספרייה זו כאשר יש צורך לשלוח מידע אל השרת ולנתח את התשובה המוחזרת ממנו במחלקה CallMyServer.

:Gson

הסבר כללי:

זוהי ספרייה של חברת Google העוזרת להפוך מידע מסוים לפורמט json (serialize) ולנתח מידע שהתקבל בצורת json אל תוך משתנים ומחלקות (deserialize).

שימוש בפרויקט:

כאשר השרת מחזיר אובייקטים מורכבים (שאינם משתנים פשוטים) הוא מעביר אותו בצורת json ויש צורך בצד הלקוח לנתח אותו כדי לייצג אותו בצורה מתאימה. בנוסף, כאשר המשתמש שולח אל השרת מידע מורכב כמו רשימה, הוא מכניס אותו בכתובת ה-URL בצורת Json כך שהשרת יהיה יכול לנתח את המידע.

מדריך למשתמש

כניסה לאפליקציה

לאחר מסך הפתיחה, במידה וטרם נרשמת לאפליקציה, יופיע מסך הרשמה. עליך לרשום את שם המשתמש החדש וללחוץ על כפתור אישור.

לאחר ההרשמה יופיע המסך הראשי. במסך זה יש 2 כפתורים: כפתור יצירת ליגה חדשה ומעבר לרשימת הליגות הקיימות, יש לבחור אחת מן האפשרויות.

1. יצירת אתגר חדש

א. יש לבחור שחקנים בכל עמדה. בעת בחירת השחקנים תועבר למסך בחירת שחקן כדי לבחור את השחקן מהעמדה שנבחרה.

ב. לאחר בניית הקבוצה, יופיע מסך מילוי פרטים ובו יש לבחור את שם הקבוצה, שם הליגה, ואת המשתמשים המוזמנים לליגה החדשה מתוך רשימת המשתמשים.

ג. לאחר שעשינו כל זאת, יש לאשר את הליגה ולשלוח אותה.

ד. לאחר יצירת האתגר המשתמש יועבר למסך רשימת הליגות.

2. צפייה ברשימת הליגות

א. על המשתמש לבחור בליגה מן רשימת הליגות שברצונו לצפות או להוסיף קבוצה אליה.

ב.

- במידה וטרם בנית קבוצה בליגה שנבחרה תועבר למסך בניית קבוצה (ולמסך בחירת שחקן).

- לאחר בניית הקבוצה יש למלא את שם הקבוצה במסך מילוי הפרטים.

- לאחר מכן יש ללחוץ על כפתור אישור הקבוצה.

- בסיום תהליך זה תופיע רשימת הליגות.

ג.

- במידה והמשתמש בנה קבוצה בליגה שנבחרה הוא יועבר לרשימת הקבוצות בליגה שנבחרה.

- יש ללחוץ על אחת הקבוצות בליגה כדי לצפות בשחקני הקבוצה.

סיכום ורפלקציה

בפרויקט זה למדתי הרבה כלים שימושיים שיכולים לעזור לי מאוד במקצועות המחשב בעתיד ובצבא.

למדתי כיצד לכתוב אפליקציה באנדרואיד – כלי חשוב מאוד ושימושי בעולם שבו הרבה ממה שאנחנו עושים מנוהל במכשיר הסמארטפון שלנו. בפעם הראשונה התנסיתי בכתיבה של פרויקט עם מספר גדול של מחלקות, כיצד להעביר ביניהן מידע בצורה אמינה ונכונה וכיצד לתמך ולחלק את קטעי הקוד בין המחלקות והמסכים השונים.

בנוסף, למדתי כיצד לקשר את האפליקציה לשרת, כלי שיכול לעזור מאוד אם רוצים לקשר בין אנשים ולשמור את המידע מכל מקום בעולם. הקישור בין השרת ללקוח גרם לי לחשוב כיצד אני פועל ומתווך בין שני הצדדים כדי שהתקשורת תהיה ברורה ונכונה.

בעבודה התנסיתי בפעם הראשונה גם כיצד לשמור נתונים מהמשתמש בצורה טבלאית. שמירת המידע בצורה זו גרמה לי לחשוב באופן שונה כיצד צריך לשמור את המידע כדי לשמור על עקרונות הנרמול ועל מנת שיהיה נוח לגשת אל מידע זה.

בפרויקט זה התנסיתי לראשונה בכלי של גוגל שנקרא Google Firebase שעוזר לשלוח מידע למשתמשים באפליקציה. בכלי זה השתמשתי בשרת כך שיידעתי את המשתמשים שהוזמנו לליגה בשפת פייתון. כלי זה מאוד שימושי באפליקציות הקיימות בשוק.

גם כתיבת תיק הפרויקט נחוצה למקצועות המחשב בעתיד כי העבודות והפרויקטים שאעשה בעתיד יוגשו ויאורגנו באותה צורה.

הייתי רוצה להודות לשגיא שעזר מאוד בהנחיית הפרויקט והוצאתו לפועל, על שהיה עבורי יועץ אסטרטגי וסייע בפתרון בעיות. תודה רבה על ההנחיה שעזרה לי מאוד בהתנהלות של העבודה, ולמדתי לחשוב בצורה נכונה, מסודרת, פשוטה ואלגנטית. לדעתי זהו כלי חשוב מאוד שיעזור לי בעתיד בתור מתכנת.

ביבליוגרפיה

Google Firebase Cloud Messaging

<https://firebase.google.com/docs/cloud-messaging>

Google Firebase Python

https://github.com/firebase/firebase-admin-python/blob/9805758ef936b882f12f08cd78cc2589985235b7/snippets/messaging/cloud_messaging.py#L24-L40

Fantasy Premier League

<https://fantasy.premierleague.com/>

Glide

<https://bumptech.github.io/glide/>

MySql

<https://www.mysql.com/>

flask

<https://flask.palletsprojects.com/en/1.1.x/>

OkHttp

<https://square.github.io/okhttp/>

Gson

<https://github.com/google/gson>

הקוד המלא בפרויקט

צד לקוח

קבצי ג'אווה

Build_team

```
package com.example.myfantasy2020;

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

import android.content.Intent;
import android.os.Bundle;
import android.view.MenuItem;
import android.view.View;
import android.widget.ImageButton;
import android.widget.TextView;
import android.widget.Toast;

import com.bumptech.glide.Glide;
import com.bumptech.glide.load.engine.bitmap_recycle.IntegerArrayAdapter;
import com.bumptech.glide.request.RequestOptions;
import com.google.gson.Gson;

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

public class Build_team extends AppCompatActivity implements
View.OnClickListener {

    String examplephotoUrl;
    int budget;

    Player_layout [] starting11;
    ImageButton btnGK, btnDef1, btnDef2,btnDef3,btnDef4,
    btnMid1,btnMid2,btnMid3,btnMid4, btnStr1, btnStr2, btnDone;

    TextView
    tvGk,tvDef1,tvDef2,tvDef3,tvDef4,tvMid1,tvMid2,tvMid3,tvMid4,tvStr1,tvS
    tr2;
    TextView tvBudget;

    String status;
    Plplayer_obj[] plsarr;
    int league_id;

    @Override
```

```

protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_build_team);

    CallMyServer cms = new CallMyServer();

    try {
        examplephotoUrl = cms.getFromServer("getPlayerPhoto");
    } catch (Exception e) {
        e.printStackTrace();
    }

    // DEFINING TEXT VIEWS AND IMAGE BUTTONS

    tvBudget = (TextView)findViewById(R.id.tv_budget_build);

    btnGK = (ImageButton)findViewById(R.id.img_gk);
    btnDef1 = (ImageButton)findViewById(R.id.def1);
    btnDef2 = (ImageButton)findViewById(R.id.def2);
    btnDef3 = (ImageButton)findViewById(R.id.def3);
    btnDef4 = (ImageButton)findViewById(R.id.def4);
    btnMid1 = (ImageButton)findViewById(R.id.mid1);
    btnMid2 = (ImageButton)findViewById(R.id.mid2);
    btnMid3 = (ImageButton)findViewById(R.id.mid3);
    btnMid4 = (ImageButton)findViewById(R.id.mid4);
    btnStr1 = (ImageButton)findViewById(R.id.st1);
    btnStr2 = (ImageButton)findViewById(R.id.st2);

    tvGk = (TextView)findViewById(R.id.tv_Gk);
    tvDef1 = (TextView)findViewById(R.id.tv_def1);
    tvDef2 = (TextView)findViewById(R.id.tv_def2);
    tvDef3 = (TextView)findViewById(R.id.tvDef3);
    tvDef4 = (TextView)findViewById(R.id.tvDef4);
    tvMid1 = (TextView)findViewById(R.id.tv_mid1);
    tvMid2 = (TextView)findViewById(R.id.tv_mid2);
    tvMid3 = (TextView)findViewById(R.id.tv_mid3);
    tvMid4 = (TextView)findViewById(R.id.tv_mid4);
    tvStr1 = (TextView)findViewById(R.id.tv_str1);
    tvStr2 = (TextView)findViewById(R.id.tv_str2);

    // SET ARRAY OF PLAYER LAYOUTS

    // EACH INDEX AT THE ARRAY INCLUDES LAYOUT OF PLAYER (TEXT
    VIEW, PLAYER CODE AND IMAGE VIEW)

    this.starting11 = new Player_layout[11];
    starting11[0] = new Player_layout(tvGk,btnGK,0);
    starting11[1] = new Player_layout(tvDef1,btnDef1,0);
    starting11[2] = new Player_layout(tvDef2,btnDef2,0);
    starting11[3] = new Player_layout(tvDef3,btnDef3,0);
    starting11[4] = new Player_layout(tvDef4,btnDef4,0);
    starting11[5] = new Player_layout(tvMid1,btnMid1,0);
    starting11[6] = new Player_layout(tvMid2,btnMid2,0);
    starting11[7] = new Player_layout(tvMid3,btnMid3,0);
    starting11[8] = new Player_layout(tvMid4,btnMid4,0);

```

```

starting11[9] = new Player_layout(tvStr1,btnStr1,0);
starting11[10] = new Player_layout(tvStr2,btnStr2,0);

// SET CLICK LISTENER TO EACH IMAGE VIEW

for(int i=0;i<starting11.length;i++){
    starting11[i].getIbPhoto().setOnClickListener(this);
}

this.budget = 750; // INITIALIZE BUDGET
tvBudget.setText("Your remaining budget is " +
String.valueOf(this.budget)+"$");

Intent intent = getIntent();
this.status = intent.getStringExtra("status"); // GET STATUS OF
VIEW FROM PREVIOUS SCREEN
this.league_id = intent.getIntExtra("league id",-1 );

btnDone = findViewById(R.id.ib_done_build); // DEFINING AND
SETTING CLICK LISTENER TO DONE BUILDING BUTTON
btnDone.setOnClickListener(this);

if(status.equals("create league")){

btnDone.setImageDrawable(getResources().getDrawable(R.drawable.next2));
}

if (status.equals("view team")){
    // IF VIEWING TEAM MODE, SHOW THE PLAYERS FROM THE SELECTED
TEAM

    btnDone.setBackground(null);
    btnDone.setEnabled(false);

    int id = intent.getIntExtra("team id",0); // GET THE ID OF
THE SELECTED TEAM FROM THE PREVIOUS SCREEN
    String playersJson = null;
    if (id!=0){
        CallMyServer cms1 = new CallMyServer();
        try {
            playersJson =
cms1.getFromServer("getSomeoneTeam/"+String.valueOf(id)); // CALL THE
SERVER TO GET THE TEAM'S PLAYERS
        } catch (Exception e) {
            e.printStackTrace();
        }

        Gson gson = new Gson();

        int [] ids = gson.fromJson(playersJson, int[].class);
// PARSING THE JSON TO ARRAY OF INTS
        tvBudget.setText("");

        // SET IMAGES OF PLAYERS BY THEIR ID

```

```

        for (int i = 0; i < starting11.length; i++) {
            String fixedurl =
examplephotoUrl.replace("152760", String.valueOf(ids[i]));

            starting11[i].getIbPhoto().setBackground(null);

Glide.with(this).load(fixedurl).into(starting11[i].getIbPhoto());

            starting11[i].getTvname().setText("");
        }
    }
}

if(!status.equals("view team")){

    // CREATING LEAGUE OR ADDING TEAM TO LEAGUE MODE

    CallMyServer cms1 = new CallMyServer();
    String jsonPlayers = null; // CALL THE SERVER TO GET THE
PREMIER LEAGUE'S PLAYERS
    try {
        jsonPlayers = cms1.getFromServer("getPLPlayers");
    } catch (Exception e) {
        e.printStackTrace();
    }

    Gson gson = new Gson();

    if(jsonPlayers != null) {

        // PARSING THE JSON TO LIST OF
        // LEAGUE PLAYERS

        List_PLs_obj players = (List_PLs_obj)
gson.fromJson(jsonPlayers, List_PLs_obj.class);

        this.plsarr = players.getPlPlayers();

    }
}

getSupportActionBar().setDisplayHomeAsUpEnabled(true); // SET
BACK PAGE BUTTON ON ACTION BAR
getSupportActionBar().setDisplayShowHomeEnabled(true);
}

@Override
public void onClick(View v) {

    // handle click event

```

```

        if(this.status.equals("view team")){
            return;
        }

        if(v == btnDone){
            if(!this.status.equals("view team")){
                // CHECK IF YOU FILLED ALL THE SQUAD
                boolean b = true;
                for (int i = 0; i< starting11.length && b;i++){
                    if(starting11[i].getCode_player() == 0){ // WHEN
YOU DIDN'T FILL ALL THE SQUAD
                        Toast.makeText(this, "You didn't fill all the
squad", Toast.LENGTH_LONG).show();
                        b = false;
                    }
                }
                // IF FILLED ALL THE SQUAD
                if(b){
                    // go to next screen
                    // pass: the starting's 11 codes, STATUS, AND
LEAGUE ID (IN CREATING LEAGUE MODE)
                    String [] codes = new String[11];
                    for(int i=0;i<starting11.length;i++){
                        codes[i] =
String.valueOf(starting11[i].getCode_player());
                    }
                    Intent intent = new
Intent(this,Full_details_creation.class);
                    intent.putExtra("team codes",codes);
                    intent.putExtra("status",this.status);
                    if(this.status.equals("add team")){
                        intent.putExtra("league id", this.league_id);
                    }
                    startActivity(intent); // PASS TO NEXT ACTIVITY
                }
            }
        }else{

            // WHEN SELECTING PLAYER

            // to players list:
            // role at field, budget

            int i=0;
            for(;i<starting11.length;i++){
                if (starting11[i].getIbPhoto() == v) {
                    break;
                }
            }

            // GO TO SCREEN OF SELECTING PLAYER FROM LIST

            Intent intent = new Intent(Build_team.this,
Pl_players_LV.class);

```



```

int [] arr = null;
// CHECK THE SELECTED POSITION
if(i == 0){ // IF SELECTED GOALKEEPER
    intent.putExtra("role",1);
}else if(i<=4){ // WHEN DEFENDER SELECTED
    intent.putExtra("role",2);
    arr = new int[4];

    for (int j = 1; j<=4; j++){
        arr[j-1] = starting11[j].getCode_player();
    }
    intent.putExtra("arr check dup",arr);

}else if(i<=8){ // WHEN SELECTED MIDFIELDER
    intent.putExtra("role",3);
    arr = new int[4];
    for (int j = 5; j<=8; j++){
        arr[j-5] = starting11[j].getCode_player();
    }
    intent.putExtra("arr check dup",arr);

}else { // WHEN STRIKER SELECTED
    intent.putExtra("role", 4);

    if (i >= 9) {
        arr = new int[2];
        arr[0] = starting11[9].getCode_player();
        arr[1] = starting11[10].getCode_player();
        intent.putExtra("arr check dup",arr);
    }

}

}

List<Plplayer_obj> plslst = Arrays.asList(plsarr);
ArrayList<Plplayer_obj> tmp = new ArrayList<>();
tmp.addAll(plslst);

intent.putParcelableArrayListExtra("PL players",tmp); //
PASS THE LIST OF OPTIONAL PLAYERS FROM THE SELECTED POSITION
intent.putExtra("budget", this.budget +
starting11[i].getCost_player()); // PASS THE REMAINED BUDGET
intent.putExtra("index",i); // PASS THE INDEX OF THE
SELECTED LAYOUT
startActivityForResult(intent,1);
}

}

// this screen gets from players list:
// name of player, code, and his price
// update starting 11 players, budget, set players photo

```

```

@Override
protected void onActivityResult(int requestCode, int resultCode,
Intent dataIntent) {
    super.onActivityResult(requestCode, resultCode, dataIntent);

    // The returned result data is identified by requestCode.
    switch (requestCode)
    {
        // This request code is set by startActivityForResult
        method.
        case 1:
            if(resultCode == RESULT_OK) {
                String namePlayerChosen =
dataIntent.getStringExtra("player name"); // GET THE NAME OF THE
SELECTED PLAYER
                int plcode = dataIntent.getIntExtra("code", 0); //
GET THE CODE OF THE SELECTED PLAYER
                int index = dataIntent.getIntExtra("index", -1); //
GET THE INDEX OF THE CHOSEN LAYOUT
                int plcost = dataIntent.getIntExtra("cost",0); //
GET THE COST OF THE SELECTED PLAYER

starting11[index].getTvname().setText(namePlayerChosen); // SET THE
NAME OF THE SELECTED PLAYER
                starting11[index].setCode_player(plcode); // SET
THE CODE OF THE SELECTED PLAYER
                starting11[index].setCost_player(plcost); // SET
THE COST OF THE SELECTED PLAYER

                // CALCULATE THE SUM OF THE SELECTED PLAYERS
                int sum = 0;
                for (int i=0;i<starting11.length;i++){
                    sum+=starting11[i].getCost_player();
                }

                this.budget = 750 - sum; // SUB FROM THE BUDGET

                tvBudget.setText("Your remaining budget is " +
String.valueOf(this.budget)+"$"); // SET TEXT TO BUDGET TEXT VIEW

                // SET PHOTO OF SELECTED PLAYER

                String fixedurl =
examplephotoUrl.replace("152760",String.valueOf(starting11[index].getCo
de_player()));

                starting11[index].getIbPhoto().setBackground(null);

Glide.with(this).load(fixedurl).into(starting11[index].getIbPhoto());
            }
        }
    }
}

```

```
// HANDLE BACK PAGE CLICKING
@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {

    if (android.R.id.home == item.getItemId()){
        onBackPressed();
        return true;
    }
    return true;
}

@Override
public void onBackPressed() {
    finish();
    super.onBackPressed();
}
}
```

CallMyServer

```

package com.example.myfantasy2020;

import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.Socket;
import java.net.URL;

import okhttp3.Call;
import okhttp3.Callback;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.Response;

public class CallMyServer implements Callback,Runnable {

    private static String address = "192.168.1.22";
    private static int port = 5000;
    private static String msg =
"http://" + address + ':' + String.valueOf(port);

    OkHttpClient client; // CLIENT FROM THE PACKAGE
    Response r;
    private String res;

    private Request req;

    public CallMyServer(){}

    public String getFromServer(String params) throws Exception {

        String retStr = null;
        try {
            getFromServerInt(params);
        } catch (Exception e) {
            e.printStackTrace();
        }

        do {
            Thread.sleep(10);
            retStr = res;
        }while (retStr == null);

        return retStr;
    }

    private void getFromServerInt(String params) throws Exception {

        String url = msg + '/' + params; // BUILD URL
        URL obj = new URL(url);
        // INITIALIZE CLIENT
    }

```

```

    client= new OkHttpClient();

    Request request = new Request.Builder()
        .url(url)
        .build();

    this.req = request;

    this.run(); // RUN THREAD OF CALL SERVER
}

@Override
public void onFailure(Call call, IOException e) {
    e.printStackTrace();
}

@Override
public void onResponse(Call call, Response response) throws
IOException {
    this.res = response.body().string(); // SET RESULT WHEN GETTING
DATA FROM SERVER
}

@Override
public void run(){
    client.newCall(this.req).enqueue(this); // MAKE NEW CALL TO
SERVER
}
}

```

Full_details_creation

```

package com.example.myfantasy2020;

import androidx.annotation.RequiresApi;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;

import android.content.DialogInterface;
import android.content.Intent;
import android.os.Build;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.ImageButton;
import android.widget.TextView;
import android.widget.Toast;

import com.google.gson.Gson;

import java.lang.reflect.Array;
import java.util.Arrays;
import java.util.List;

public class Full_details_creation extends AppCompatActivity {

    EditText et_team_name,et_league_name;
    ImageButton ib_add_friends;
    ImageButton ib_submit;
    String status;
    TextView tv_add;
    String [] teams_codes;
    String jsonUsers = null;
    String [] selected_users;
    String you;
    int leauge_id;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_full_details_creation);
        // DEFINING WIDGETS BY THE MODE (CREATING A LEAGUE OR ADDING
TEAM TO LEAGUE)
        Intent intent = getIntent();
        // GET SELECTED SQUAD AND STATUS FROM THE PREVIOUS SCREEN
        teams_codes = intent.getStringArrayExtra("team codes");
        this.status = intent.getStringExtra("status");

        et_team_name = (EditText)findViewById(R.id.et_team_name);
        et_league_name = (EditText)findViewById(R.id.et_league_name);
        tv_add = (TextView)findViewById(R.id.tv_add_friend);
        MainActivity mainActivity = new MainActivity();
        // READING THE USERNAME FILE TO THE USERNAME
        this.you = mainActivity.readFile(MainActivity.file_user_key);
    }
}

```

```

// GET LIST OF OTHER USERS WHO CAN ADD TO YOUR LEAGUE
if (this.status.equals("create league")){
    CallMyServer cms = new CallMyServer();
    // CALLING SERVER
    try {
        jsonUsers = cms.getFromServer("getOtherUsers/"+you);
    } catch (Exception e) {
        e.printStackTrace();
    }
}

// GET LEAGUE ID ON ADDING TEAM TO LEAGUE
if (this.status.equals("add team")){
    this.leauge_id = intent.getIntExtra("league id",-1);
}

// HANDLE CLICKING ON ADDING FRIENDS IMAGE VIEW
// AND SHOW THE MULTICHOOSSE DIALOG
ib_add_friends = (ImageButton)findViewById(R.id.add_people);
ib_add_friends.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (status.equals("create league")){

            // add friends
            // go to next screen

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

            Gson gson = new Gson();
            final String [] arrUsers =
gson.fromJson(jsonUsers,String[].class);
            // SET ARRAY OF BOOLEANS OF SELECTED USERS
            final boolean [] checked = new
boolean[arrUsers.length];
            Arrays.fill(checked, Boolean.FALSE);
            builder.setTitle("Select the users you want in your
league"); // SET TITLE
            // EVENT OF CHOOSING ITEM
            builder.setMultiChoiceItems(arrUsers, checked, new
DialogInterface.OnMultiChoiceClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int
which, boolean isChecked) {
                    checked[which] = isChecked;
                    String current_item = arrUsers[which]; // SET
IS CHECKED AT THE INDEX
                }
            });
            // HANDLE OK BUTTON CLICKING
            builder.setPositiveButton("Ok", new
DialogInterface.OnClickListener() {
                @Override

```

```

which) {

    public void onClick(DialogInterface dialog, int

        // COUNT SELECTED USERS

        int c = 0;
        for (int i=0;i<arrUsers.length;i++){
            if (checked[i] == true){
                c++;
            }
        }

        // MAKE ARRAY OF STRING WITH THE SELECTED USERS
        selected_users = new String[c];
        c = 0;
        for (int i=0;i<arrUsers.length;i++){
            if (checked[i] == true){
                selected_users[c] = arrUsers[i];
                c++;
            }
        }

    }

});
// CANCEL BUTTON
builder.setNegativeButton("Cancel", new
DialogInterface.OnClickListener() {
    @Override
    public void onClick(DialogInterface dialog, int

which) {

        // do nothing

    }

});

AlertDialog dialog = builder.create();
// show alert dialog
dialog.show();

}
}

});
// MAKE ADD FRIENDS BUTTON, LEAGUE NAME EDIT TEXT INVISIBLE
if (this.status.equals("add team")){
    ib_add_friends.setBackground(null);
    tv_add.setVisibility(View.INVISIBLE);
    et_league_name.setEnabled(false);
    et_league_name.setVisibility(View.INVISIBLE);
}

ib_submit = (ImageButton)findViewById(R.id.btn_submit_League);
// HANDLE SUBMIT BUTTON
ib_submit.setOnClickListener(new View.OnClickListener() {
    @RequiresApi(api = Build.VERSION_CODES.O)
    @Override
    public void onClick(View v) {

```



```

// GET DATA FROM THE USER
String teamName = et_team_name.getText().toString();
String leagueName =
et_league_name.getText().toString();

if (!isValidString(teamName)){
    return;
}
Gson gson = new Gson();
String jsonPlayers =
gson.toJson(teams_codes,String[].class);

if (status.equals("add team")){
    // build url
    // add team to league
    String params =
    "addTeamToLeague/"+you+"/"+String.valueOf(league_id)+"/"+teamName+"/"+j
sonPlayers;

    CallMyServer cms = new CallMyServer();
    String res = null;
    try {
        res = cms.getFromServer(params);
    } catch (Exception e) {
        e.printStackTrace();
    }

    Intent i = new Intent(Full_details_creation.this,
Leagues_listView.class);
    startActivity(i);
}

if(status.equals("create league")){

    // build url
    // call server to set league

    if(!isValidString(leagueName) || selected_users ==
null){

        return;
    }

    String jsonSelectedUsers =
gson.toJson(selected_users,String[].class);

    MainActivity ma = new MainActivity();
    // BUILD URL
    String param =
    "addLeague/"+ma.readFile(MainActivity.file_user_key)+"/"+leagueName+"/"+
+teamName+"/"+jsonPlayers+"/"+jsonSelectedUsers;
    // CALL THE SERVER TO CREATE LEAGUE
    CallMyServer cms = new CallMyServer();
    String s = null;
    try {
        s = cms.getFromServer(param);
    } catch (Exception e) {
        e.printStackTrace();
    }
}

```

```

    }

    if (s.equals("added")){
        Intent intent = new
Intent(Full_details_creation.this, Leagues_listView.class);
        startActivity(intent);
    }

    }

    });
}

// CHECK IF THE STRING CAN BE NAME OF LEAGUE OR TEAM
private boolean isValidString(String a){
    if(a == null || a.length() == 0 || a.trim().isEmpty()){
        return false;
    }
    return true;
}

}

```

GettingDeviceTokenService

```
package com.example.myfantasy2020;

import android.app.Service;
import android.content.Intent;
import android.os.IBinder;
import android.util.Log;

import com.google.firebase.iid.FirebaseInstanceId;
import com.google.firebase.iid.FirebaseInstanceIdService;

public class GettingDeviceTokenService extends
FirebaseInstanceIdService {
    @Override
    public void onTokenRefresh() {
        super.onTokenRefresh();
        String deviceToken =
FirebaseInstanceId.getInstance().getToken();
        Log.d("DEVICE TOKEN:", deviceToken);
    }
}
```

League_LV_adapter

```

package com.example.myfantasy2020;

import android.app.Activity;
import android.content.Context;
import android.graphics.Color;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ImageView;
import android.widget.TextView;

import androidx.annotation.NonNull;

import org.w3c.dom.Text;

import java.util.Arrays;
import java.util.List;

public class League_LV_adapter extends ArrayAdapter<League_obj> {

    // THIS CLASS HELPS INFLATE THE LIST OF LEAGUES TO LAYOUTS IN THE
    LIST VIEW

    Context context; // THE CONTEXT WHERE THE LIST VIEW PLACED IN
    List<League_obj> objects; // LIST OF LEAGUES

    public League_LV_adapter(@NonNull Context context, int resource,
int textViewResourceId, @NonNull List<League_obj> objects) {
        super(context, resource, textViewResourceId, objects);
        this.context = context;
        this.objects = objects;
    }

    @NonNull
    @Override
    public View getView(int position, View convertView, @NonNull
ViewGroup parent) {
        LayoutInflater inflater = ((Activity)
context).getLayoutInflater();
        View view = inflater.inflate(R.layout.league_row, parent,
false);

        League_obj temp = objects.get(position); // GET THE ITEM NEED
TO SET LAYOUT FIR

        // SET INFORMATION IN THE LAYOUT

        TextView tvName =
(TextView)view.findViewById(R.id.tv_id_league_row);
        tvName.setText(temp.getName());

        TextView tvCreatedBy =

```

```

(TextView)view.findViewById(R.id.tv_League_created_by_row);
    tvCreatedBy.setText("Created By: "+temp.getCreator_username());

    TextView tvNumTeamsRow = (TextView)
view.findViewById(R.id.tv_amount_teams_row);
    tvNumTeamsRow.setText(String.valueOf(temp.getTeams().length) +
" team/s in league");

    TextView tvCurrentPlace =
(TextView)view.findViewById(R.id.tv_current_place_in_League_row);
    TextView tvNumPointsInLeague =
(TextView)view.findViewById(R.id.tv_your_points_in_League_row);

    MainActivity ma = new MainActivity();
    String u = ma.readFile(MainActivity.file_user_key);
    Arrays.sort(temp.getTeams());

    int j = 1;
    int points = 0;

    for(Team_obj i : temp.getTeams()){
        if(i.getUsernameOwner().equals(u)){
            points = i.getPoints();
            break;
        }
        j++;
    }

    tvCurrentPlace.setText(String.valueOf(j) + " place");
    tvNumPointsInLeague.setText(String.valueOf(points)+" pts");

    return view;
}
}

```

League_obj

```

package com.example.myfantasy2020;

// LEAGUE WHICH THE USER INVITED FOR OR CREATED

public class League_obj {

    private String name; // LEAGUE'S NAME
    private int idLeague; // LEAGUES PRIMARY KEY
    private int idOwner; // PRIMARY KEY OF OWNER
    private Team_obj[] teams; // ARRAY OF TEAMS WHO WERE BUILT TO THE
    LEAGUE
    private String creator_username; // USERNAME OF THE LEAGUE'S
    CREATOR

    public String getCreator_username() {
        return creator_username;
    }

    public void setCreator_username(String creator_username) {
        this.creator_username = creator_username;
    }

    public String getName() {
        return name;
    }

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

    public int getIdLeague() {
        return idLeague;
    }

    public void setIdLeague(int idLeague) {
        this.idLeague = idLeague;
    }

    public int getIdOwner() {
        return idOwner;
    }

    public void setIdOwner(int idOwner) {
        this.idOwner = idOwner;
    }

    public Team_obj[] getTeams() {
        return teams;
    }

    public void setTeams(Team_obj[] teams) {
        this.teams = teams;
    }
}

```

Leagues_listView

```

package com.example.myfantasy2020;

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

import android.content.Intent;
import android.os.Bundle;
import android.telecom.Call;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ListView;
import android.widget.Toast;

import com.google.gson.Gson;

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

public class Leagues_listView extends AppCompatActivity{

    // A SCREEN OF THE LEAGUES THAT THE USER INVITED FOR OR CREATED

    ListView lv_my_leagues; // THE LIST VIEW WHICH CONTAINS THE LIST OF
    THE USER'S LEAGUES
    League_LV_adapter leaugesAdapter; // HELPS TO INFLATE THE LIST TO
    THE LIST VIEW

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

        lv_my_leagues = (ListView) findViewById(R.id.lv_Leagues);

        CallMyServer cms = new CallMyServer();
        MainActivity ma = new MainActivity();
        // CALL READ THE USERNAME FROM THE FILE
        String s = ma.readFile(MainActivity.file_user_key);
        // GET THE USERNAME'S LEAGUES

        try {

            String leaguesJson = cms.getFromServer("getMyLeagues/"+s);

            //cms.getFromServer("getMyLeagues/"+s);
            //String LeaguesJson = cms.getRes();

            //do{LeaguesJson = cms.getRes();}while (LeaguesJson ==
null);

```

```

// PARSING THE JSON FROM THE SERVER
Gson gson = new Gson();
Rootobject myLeagues = (Rootobject)
gson.fromJson(leaguesJson,Rootobject.class);

League_obj [] lgs = myLeagues.getLeagues();

List<League_obj> lst_lgs = Arrays.asList(lgs);
// INITIALIZE THE LIST VIEW FOM THE LIST
if (lst_lgs.size() != 0){
    leaugesAdapter = new League_LV_adapter(this, 0, 0,
lst_lgs);
    lv_my_leagues.setAdapter(leaugesAdapter);
}else{
    Toast.makeText(Leagues_listView.this, "No available
results", Toast.LENGTH_LONG).show();
}

// SET LISTENER TO BACK PAGE BUTTON
getSupportActionBar().setDisplayHomeAsUpEnabled(true);
getSupportActionBar().setDisplayShowHomeEnabled(true);

// CLICKING ON ITEM FROM THE LIST VIEW
lv_my_leagues.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> adapterView,
View view, int i, long l) {
        League_obj lastSelected = leaugesAdapter.getItem(i); //
GET THE OBJECT THE LIST VIEW
        MainActivity mainActivity = new MainActivity();
        String you =
mainActivity.readFile(MainActivity.file_user_key);

        // CHECK IF THE USER HAVE ANSWERED THE CHALLENGE
        boolean isDone = false;
        for(int j=0;j<lastSelected.getTeams().length &&
!isDone;j++){
            if
(lastSelected.getTeams()[j].getUsernameOwner().equals(you)){
                isDone = true;
            }
        }
        // HAS ANSWERED THE CHALLENGE ?
        if (isDone){
            // IF YES
            // PASS TO THE TEAMS IN LEAGUE SCREEN
            Intent intent = new Intent(Leagues_listView.this,
Lv_Teams_In_League.class);
            intent.putExtra("name", lastSelected.getName()); //
PASS THE LEAGUE'S NAME

            ArrayList <Team_obj> to = new ArrayList<>();

            Team_obj []aaa = lastSelected.getTeams();

```



```

        for (Team_obj t: aaa) {
            to.add(t);
        }

        intent.putParcelableArrayListExtra
("teams_in_league", to); // PASS THE ARRAY OF TEAMS
        startActivity(intent);
    }else{
        Intent intent1 = new Intent(Leagues_listView.this,
Build_team.class); // MOVE TO BUILDING SCREEN
        intent1.putExtra("status","add team"); // PASS MODE
        intent1.putExtra("league
id",lastSelected.getIdLeague()); // PASS THE ID OF THE SELECTED LEAGUE
        startActivity(intent1);
    }
    });

    } catch (Exception e) {
        e.printStackTrace();
    }

}

// MOVE TO MAIN MENU SCREEN
@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {

    if (android.R.id.home == item.getItemId()){
        onBackPressed();
        return true;
    }
    return true;
}

@Override
public void onBackPressed() {
    Intent a = new Intent(this,Main_menu.class);
    startActivity(a);
}
}

```

List_PLs_obj

```
package com.example.myfantasy2020;

public class List_PLs_obj {
    Plplayer_obj[] plPlayers;

    public Plplayer_obj[] getPlPlayers() {
        return plPlayers;
    }
}
```

Lv_Teams_In_League

```

package com.example.myfantasy2020;

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

import android.content.Intent;
import android.os.Bundle;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ListView;
import android.widget.TextView;
import java.util.ArrayList;

// SCREEN OF TEAMS OF THE CHOSEN LEAGUES

public class Lv_Teams_In_League extends AppCompatActivity {

    ListView lvTeamsInLeague;
    TextView tvLeagueName;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_lv_teams_in_League);
        lvTeamsInLeague = findViewById(R.id.lv_teams_inLeague);

        final Intent prev = getIntent();
        if (prev.getExtras() != null) {
            String name = prev.getExtras().getString("name"); // SET
LEAGUE'S NAME

            tvLeagueName = findViewById(R.id.tv_League_name_teamsLST);
            tvLeagueName.setText(name);

            // SET THE LIST OF TEAM'S FROM THE PREVIOUS SCREEN
            ArrayList<Team_obj> objects =
prev.getParcelableArrayListExtra("teams_in_League");
            final Team_Adapter teamAdapter = new Team_Adapter(this, 0,
0, objects);
            try {
                lvTeamsInLeague.setAdapter(teamAdapter);
            } catch (Exception e) {
                e.printStackTrace();
            }

            // MOVE TO BUILD TEAM SCREEN AND SHOW THE CHOSEN TEAM
            lvTeamsInLeague.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
                @Override
                public void onItemClick(AdapterView<?> adapterView,
View view, int i, long l) {
                    Team_obj lastSelected = teamAdapter.getItem(i);

```

```

        Intent intent = new
Intent(Lv_Teams_In_League.this, Build_team.class);
        intent.putExtra("status", "view team");
        intent.putExtra("team id", lastSelected.getIdTeam());
        startActivity(intent);
    }
});

}

// CREATE PREVIOUS PAGE BUTTON
getSupportActionBar().setDisplayHomeAsUpEnabled(true);
getSupportActionBar().setDisplayShowHomeEnabled(true);

}

// HANDLE CLICKING ON BACK PAGE BUTTON

@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {

    if (android.R.id.home == item.getItemId()){
        onBackPressed();
        return true;
    }
    return true;
}

@Override
public void onBackPressed() {
    finish();
    super.onBackPressed();
}

}

```

Main_menu

```

package com.example.myfantasy2020;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.ImageButton;
import android.widget.TextView;
import android.widget.Toast;

import com.google.firebase.iid.FirebaseInstanceId;

public class Main_menu extends AppCompatActivity implements
View.OnClickListener {

    TextView tv1;
    ImageButton ib_show_leagues;
    ImageButton ib_create_league;

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

        tv1 = (TextView)findViewById(R.id.hello_tv);

        MainActivity m = new MainActivity();
        String you = m.readFile(MainActivity.file_user_key);
        tv1.setText("hello " + you); // SET NAME OF THE USERNAME FROM
THE FILE

        ib_show_leagues =
(ImageButton)findViewById(R.id.btn_show_leagues);
        ib_show_leagues.setOnClickListener(this); // SET CLICK LISTENER
TO THIS BUTTON

        ib_create_league =
(ImageButton)findViewById(R.id.btn_create_league);
        ib_create_league.setOnClickListener(this); // SET CLICK
LISTENER TO THIS BUTTON

        checkToken(you); // CHECK IF THE TOKEN IS UPDATED

    }

    @Override
    public void onClick(View v) {
        switch (v.getId())
        {
            case R.id.btn_create_league:
                goNextScreen("build team screen");
        }
    }

```

```

        break;
    case R.id.btn_show_Leagues:
        goNextScreen("show leagues screen");
        break;
    }
}

// MOVE TO THE CHOSEN SCREEN BY IT'S NAME
private void goNextScreen(String nextScreen){
    if(nextScreen.equals("build team screen")){
        Intent intent = new Intent(this, Build_team.class);
        intent.putExtra("status", "create league");
        startActivity(intent);
    }else{
        Intent intent = new Intent(this, Leagues_listView.class);
        startActivity(intent);
    }
}

// CHECK IF THE TOKEN IS UPDATED
// IF TOKEN HAS CHANGE: WRITE NEW TOKEN TO FILE
private void checkToken(String you){
    String newToken = FirebaseInstanceId.getInstance().getToken();
    MainActivity mainActivity = new MainActivity();

    String current_token =
mainActivity.readFile(MainActivity.file_token_key);
    if (!current_token.equals(newToken)){
        CallMyServer cms = new CallMyServer();
        String res = null;
        try {

            res = cms.getFromServer("editToken/"+you+'/' +newToken);
        } catch (Exception e) {
            e.printStackTrace();
        }

mainActivity.setToFile(MainActivity.file_token_key,newToken);
    }
}

// DO NOTHING ON BACK PRESS
@Override
public void onBackPressed() {
    return;
}
}

```

MainActivity

```

package com.example.myfantasy2020;

import androidx.appcompat.app.AppCompatActivity;

import android.app.Activity;
import android.content.Context;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.os.Environment;
import android.preference.PreferenceManager;
import android.util.Log;
import android.view.View;
import android.widget.EditText;
import android.widget.ImageButton;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;

import com.google.firebase.iid.FirebaseInstanceId;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.util.Scanner;

public class MainActivity extends Activity implements
View.OnClickListener {
    ImageButton ibRegister;
    EditText etNewUser;

    String inpUser;

    // DEFINING NAMES OF FILES
    static final String file_user_key = "your_user.txt"; // USERNAME
FILE
    static final String file_token_key = "your_token.txt"; // TOKEN
FILE

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

        etNewUser = (EditText)findViewById(R.id.etNewUser);
        inpUser = etNewUser.getText().toString();

        ibRegister = (ImageButton) findViewById(R.id.btn_submit_user);
        ibRegister.setOnClickListener(this);

```

```

// IF USERNAME HAS ALREADY REGISTERED MOVE TO NEXT SCREEN
boolean b = isFileExists(MainActivity.file_user_key);

if(b){
    goNextScreen();
}

}

// MOVE TO MAIN MENU SCREEN
private void goNextScreen(){
    Intent intent = new Intent(this, Main_menu.class);
    startActivity(intent);
}

@Override
public void onClick(View view) {

    // WHEN USERNAME SUBMITTED A USERNAME
    switch (view.getId())
    {
        case R.id.btn_submit_user:
            Toast.makeText(MainActivity.this, "This is my Toast
message!",
                        Toast.LENGTH_LONG).show();

            String inp = etNewUser.getText().toString();

            boolean check = this.isValidUser(inp);

            if(!check){break;} // already exists

            setToFile(MainActivity.file_user_key,inp);

            String first_token =
FirebaseInstanceId.getInstance().getToken();

            setToFile(MainActivity.file_token_key,first_token);

            CallMyServer cms = new CallMyServer();
            String res = null;
            try {
                res =
cms.getFromServer("addUser/"+inp+"/"+first_token);
            } catch (Exception e) {
                e.printStackTrace();
            }
            if (res.equals("added")){
                goNextScreen();
                break;
            }
        }
    }
}

```



```

    }

    private boolean isFileExists(String file_key){
        File file = this.getFileStreamPath(file_key);
        if(file == null || !file.exists()) {
            return false;
        }
        return true;
    }

    public void setToFile(String filename, String data) {

        try {
            FileOutputStream fout =
openFileOutput(filename,MODE_PRIVATE);
            fout.write(data.getBytes());
            fout.close();
            File fdir = new
File(getFilesDir(),MainActivity.file_user_key);
            //Toast.makeText(getApplicationContext(),"file saved at:
"+fdir,Toast.LENGTH_LONG).show();
        }
        catch (Exception e) {
            Log.e("Exception", "File write failed: " + e.toString());
        }
    }

    private boolean isValidUser(String s){

        // call server, write to database and return answer
        // if username already exists user should write new username

        CallMyServer c = new CallMyServer();
        String res = null;
        try {
            res = c.getFromServer("isUsernameExists/"+s);
            if(res.equals("valid name")){
                return true;
            }
            if(res.equals("exists")){
                return false;
            }
        } catch (Exception e) {
            e.printStackTrace();
        }

        return false;
    }

    final static String path =
Environment.getExternalStorageDirectory().getAbsolutePath();
    final static String TAG = MainActivity.class.getName();

```

```
// RETURNS THE CONTENT OF FILE BY IT'S NAME
public String readFile(String file_name) {
    String line = "";
    int c;

    try {
        File f = new File(
            "/data/data/com.example.myfantasy2020/files", file_name);
        FileInputStream fin = new FileInputStream(f);

        while ((c = fin.read())!=-1){
            line += Character.toString((char)c);
        }
        fin.close();

        return line;

    } catch (Exception ex) {
        ex.printStackTrace();
    }
    return "";
}

}
```

MyFirebaseMessagingService

```

package com.example.myfantasy2020;

import android.app.NotificationChannel;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.app.Service;
import android.content.Context;
import android.content.Intent;
import android.media.RingtoneManager;
import android.net.Uri;
import android.os.Build;
import android.os.IBinder;

import androidx.core.app.NotificationCompat;

import com.google.firebase.messaging.FirebaseMessagingService;
import com.google.firebase.messaging.RemoteMessage;

public class MyFirebaseMessagingService extends
    FirebaseMessagingService {

    public static int NOTIFICATION_ID = 1;

    @Override
    public void onMessageReceived(RemoteMessage remoteMessage){

        // EVENT OF GETTING MESSAGE FROM THE FIREBASE
        String orgtitle = remoteMessage.getNotification().getTitle();
        String orgbody = remoteMessage.getNotification().getBody();

        generateNotification(orgbody,orgtitle);

    }
    // GENERATES NOTIFICATION BY TITLE AND BODY
    private void generateNotification(String body, String title) {
        Uri soundUri =
RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION); //
DEFINE SOUND BY THE DEFAULT RINGTONE
        NotificationCompat.Builder notificationBuilder = new
NotificationCompat.Builder(this) // DEFINE BUILDER
            .setSmallIcon(R.mipmap.ic_launcher)
            .setContentTitle(title)
            .setContentText(body)
            .setAutoCancel(true)
            .setSound(soundUri);

        Intent intent = new Intent(this,MainActivity.class);
        PendingIntent pendingIntent =
PendingIntent.getActivity(this,0,intent,PendingIntent.FLAG_UPDATE_CURRE
NT);

        notificationBuilder.setContentIntent(pendingIntent);

        NotificationManager notificationManager =

```

```
(NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);

if (NOTIFICATION_ID > 1073741824){
    NOTIFICATION_ID = 0;
}

// HANDLE DIFFERENT ANDROID VERSIONS
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O)
{
    String channelId = "Your_channel_id";
    NotificationChannel channel = new NotificationChannel(
        channelId,
        "Channel human readable title",
        NotificationManager.IMPORTANCE_HIGH);
    notificationManager.createNotificationChannel(channel);
    notificationBuilder.setChannelId(channelId);
}

// NOTIFY USER

notificationManager.notify(NOTIFICATION_ID++,notificationBuilder.build(
));

}

}
```

PL_Player_Adapter

```

package com.example.myfantasy2020;

import android.app.Activity;
import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ImageView;
import android.widget.TextView;

import androidx.annotation.NonNull;

import com.bumptech.glide.Glide;
import com.squareup.picasso.Picasso;

import java.util.Arrays;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class PL_Player_Adapter extends ArrayAdapter<Plplayer_obj> {

    // THIS CLASS HELPS INFLATE THE LIST OF LEAGUES TO LAYOUTS IN THE
    LIST VIEW

    Context context; // THE CONTEXT WHERE THE LIST VIEW PLACED IN
    List<Plplayer_obj> objects; // LIST PL PLAYERS (PL = PREMIER
    LEAGUE)
    static String examplePhotoPlayer; // URL OF A PLAYER'S PHOTO FROM
    THE SERVER
    static String examplePhototeam; // URL OF A TEAM'S PHOTO FROM THE
    SERVER
    final Map<Integer,String> teamsnames= new HashMap<Integer,
    String>(); // HELPS CONVERTING TEAM'S CODE TO TEAM'S NAME

    public PL_Player_Adapter(@NonNull Context context, int resource,
    int textViewResourceId, @NonNull List<Plplayer_obj> objects) throws
    Exception {
        super(context, resource, textViewResourceId, objects);
        this.context = context;
        this.objects = objects;

        // DEFINE THE HASH MAP
        teamsnames.put(14, "Liverpool");
        teamsnames.put(91, "Bournemouth");
        teamsnames.put(3, "Arsenal");
        teamsnames.put(21, "West Ham");
        teamsnames.put(31, "Crystal Palace");
        teamsnames.put(57, "Watford");
        teamsnames.put(49, "Sheffield Utd");
        teamsnames.put(4, "Newcastle");
    }

```

```

        teamsnames.put(39, "Wolves");
        teamsnames.put(36, "Brighton");
        teamsnames.put(90, "Burnley");
        teamsnames.put(6, "Spurs");
        teamsnames.put(8, "Chelsea");
        teamsnames.put(11, "Everton");
        teamsnames.put(1, "Man Utd");
        teamsnames.put(43, "Man City");
        teamsnames.put(13, "Leicester");
        teamsnames.put(7, "Aston Vila");
        teamsnames.put(45, "Norwich");
        teamsnames.put(20, "Southampton");

        getURLS();

    }

    // SET THE LAYOUT TO EACH ITEM
    @NonNull
    @Override
    public View getView(int position, View convertView, @NonNull
    ViewGroup parent) {
        LayoutInflater inflater = ((Activity)
        context).getLayoutInflater();
        View view = inflater.inflate(R.layout.pl_player_row,
        parent, false);

        Plplayer_obj temp = objects.get(position);

        // SET INFORMATION IN THE LAYOUT

        TextView fullname = view.findViewById(R.id.pl_fullname_pl_row);
        fullname.setText(temp.getFullname());

        ImageView ivP = view.findViewById(R.id.pl_pic_pl_row);

        String url1 = PL_Player_Adapter.examplePhotoPlayer;

        url1 = url1.replace("152760",String.valueOf(temp.getCode()));

        //Picasso.get().Load(url1).into(ivP);

        Glide.with(this.context).load(url1).into(ivP);

        ImageView ivT = view.findViewById(R.id.pl_team_pic_pl_row);

        String url2 = PL_Player_Adapter.examplePhototeam;

        url2 = url2.replace("14",String.valueOf(temp.getTeam()));

        Glide.with(this.context).load(url2).into(ivT);

        TextView tv_price = view.findViewById(R.id.pl_cost_pl_row);

```

```

tv_price.setText(String.valueOf(temp.getCost()+tv_price.getText().toString()));

        TextView teams_name =
view.findViewById(R.id.teams_name_pl_row);
        int c = temp.getTeam();
        String s = this.teamsnames.get(c);
        teams_name.setText(s);

        return view;
    }

    private void getURLS() throws Exception {
        if (PL_Player_Adapter.examplePhotoPlayer == null ||
PL_Player_Adapter.examplePhototeam == null
            || PL_Player_Adapter.examplePhotoPlayer.equals("") ||
PL_Player_Adapter.examplePhotoPlayer.equals("")){
            // CALL SERVER TO GET URL PHOTO
            CallMyServer cms = new CallMyServer();
            PL_Player_Adapter.examplePhotoPlayer = null;
            PL_Player_Adapter.examplePhotoPlayer =
cms.getFromServer("getPlayerPhoto");

            // CALL SERVER TO GET URL TEAM'S PHOTO
            PL_Player_Adapter.examplePhototeam = null;

            CallMyServer cms2 = new CallMyServer();
            PL_Player_Adapter.examplePhototeam =
cms2.getFromServer("getTeamPhoto");

        }

    }
}

```

Pl_players_LV

```

package com.example.myfantasy2020;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.telecom.Call;
import android.view.Menu;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ListView;
import android.widget.SearchView;
import android.widget.TextView;
import android.widget.Toast;
import com.google.gson.Gson;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class Pl_players_LV extends AppCompatActivity{

    ListView lv; // LIST VIEW OF THE PLAYERS
    static int index; // INDEX OF SELECTED PLAYER
    PL_Player_Adapter player_adapter; // ADAPTER HELPS SET THE LIST
VIEW
    int [] arrCheckDup; // ARRAY OF PLAYERS' CODES TO CHECK IF YOU HAD
ALREADY CHOSE THIS PALYER
    //TextView tvBudget;
    SearchView sv; // SEARCH VIEW TO SEARCH PLAYER/S
    List<Plplayer_obj> tmp1; // LIST OF OBJECTS OF PL PLAYERS

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_pl_players__lv);
        //DEFINING WIDGETS
        lv = (ListView) findViewById(R.id.lv_pls);
        arrCheckDup = null;
        Intent intent = getIntent();

        ArrayList<Plplayer_obj> pls1 =
intent.getParcelableArrayListExtra("PL players");

        List<Plplayer_obj> pls = (List<Plplayer_obj>)pls1;

        tmp1 = (List<Plplayer_obj>) new ArrayList<Plplayer_obj>();

        int role = intent.getIntExtra("role",0);
        final int budget = intent.getIntExtra("budget",0);
        index = intent.getIntExtra("index",11);

        if (role > 1 && role <=4){
            arrCheckDup = intent.getIntArrayExtra("arr check dup");
        }
    }

```



```

        if(role!=0){
            for (Plplayer_obj i:pls) {
                if(i.getPosition() == role){
                    tmp1.add(i);
                }
            }
        }

        // SET LIST VIEW
        player_adapter = null;

        try {
            player_adapter = new PL_Player_Adapter(this, 0, 0, tmp1);
            lv.setAdapter(player_adapter);
        } catch (Exception e) {
            e.printStackTrace();
        }

        // DEFINING THE LIST VIEW
        sv = findViewById(R.id.sv_pls);
        // HANDLE SEARCH VIEW EVENTS
        sv.setOnQueryTextListener(new SearchView.OnQueryTextListener()
    {
        @Override
        public boolean onQueryTextSubmit(String query) {
            return false;
        }
        @Override
        public boolean onQueryTextChange(String newText) {
            List<Plplayer_obj> new_pls = new ArrayList<>();
            // MAKE A NEW LIST FROM THE CONDITION
            for(Plplayer_obj p : tmp1){

                if(p.getFullname().toLowerCase().contains(newText.toLowerCase())){
                    new_pls.add(p);
                }
            }
            try {
                player_adapter = new
                PL_Player_Adapter(Pl_players_LV.this, 0, 0, new_pls);
            } catch (Exception e) {
                e.printStackTrace();
            }
            lv.setAdapter(player_adapter);
            if(new_pls.size() == 0){
                Toast.makeText(Pl_players_LV.this, "No available
                results", Toast.LENGTH_LONG).show();
            }
            return false;
        }
    });
}

```

```

        lv.setOnItemClickListener(new AdapterView.OnItemClickListener()
        {
            @Override
            public void onItemClick(AdapterView<?> parent, View view,
            int position, long id) {

                // RETURN THE CHOSEN PLAYER TO BUILDING TEAM SCREEN
                Intent intent = new Intent();
                Plplayer_obj lastSelected =
                player_adapter.getItem(position);

                // CHECK IF YOU HAVE ENOUGH BUDGET
                if (lastSelected.getCost() > budget){
                    Toast.makeText(Pl_players_LV.this, "You do not have
                    enough budget", Toast.LENGTH_LONG).show();
                }else{
                    // CHECK IF YOU HAVE ALREADY CHOSE THE PLAYER
                    int k = 0;
                    if (arrCheckDup != null){
                        for(int i = 0; i<arrCheckDup.length;i++){
                            if (arrCheckDup[i] == lastSelected.getCode()){
                                k++;
                            }
                        }
                        if (k>0){
                            Toast.makeText(Pl_players_LV.this, "You have
                            selected this player", Toast.LENGTH_LONG).show();
                        }else{
                            // WHEN VALID PLAYER
                            // RETURN RESULT TO PREVIOUS ACTIVITY
                            intent.putExtra("index",Pl_players_LV.index);
                            intent.putExtra("cost",lastSelected.getCost());
                            intent.putExtra("code",lastSelected.getCode());
                            intent.putExtra("player
                            name",lastSelected.getFullname());

                            setResult(RESULT_OK, intent);
                            finish();
                        }
                    }else {
                        // WHEN YOU CHOSE THE A GOALKEEPER
                        intent.putExtra("index",Pl_players_LV.index);
                        intent.putExtra("cost",lastSelected.getCost());
                        intent.putExtra("code",lastSelected.getCode());
                        intent.putExtra("player
                        name",lastSelected.getFullname());
                        setResult(RESULT_OK, intent);
                        finish();
                    }
                }
            }
        });
    }
}

```

Player_layout

```
package com.example.myfantasy2020;

import android.widget.ImageButton;
import android.widget.TextView;

public class Player_layout {

    // A PLAYER LAYOUT IN BUILD TEAM ACTIVITY
    private TextView tvname;
    private ImageButton ibPhoto;
    private int code_player;
    private int cost_player;

    public Player_layout(TextView tvname, ImageButton ibPhoto, int
code_player) {
        this.tvname = tvname;
        this.ibPhoto = ibPhoto;
        this.code_player = code_player;
        this.cost_player = 0;
    }

    public void setCode_player(int code_player) {
        this.code_player = code_player;
    }

    public TextView getTvname() {
        return tvname;
    }

    public ImageButton getIbPhoto() {
        return ibPhoto;
    }

    public int getCode_player() {
        return code_player;
    }

    public void setCost_player(int cost_player) {
        this.cost_player = cost_player;
    }

    public int getCost_player() {
        return cost_player;
    }
}
```

Plplayer_obj

```

package com.example.myfantasy2020;
import android.os.Parcel;
import android.os.Parcelable;

public class Plplayer_obj implements Parcelable {

    private String fullname;
    private int position;
    private int cost;
    private int code;
    private int team;

    public Plplayer_obj() {
    }

    protected Plplayer_obj(Parcel in) {
        fullname = in.readString();
        position = in.readInt();
        cost = in.readInt();
        code = in.readInt();
        team = in.readInt();
    }

    public static final Creator<Plplayer_obj> CREATOR = new
    Creator<Plplayer_obj>() {
        @Override
        public Plplayer_obj createFromParcel(Parcel in) {
            return new Plplayer_obj(in);
        }

        @Override
        public Plplayer_obj[] newArray(int size) {
            return new Plplayer_obj[size];
        }
    };

    public String getFullname() {
        return fullname;
    }

    public void setFullname(String fullname) {
        this.fullname = fullname;
    }

    public int getPosition() {
        return position;
    }

    public void setPosition(int position) {
        this.position = position;
    }

    public int getCost() {
        return cost;
    }

```

```
}

public void setCost(int cost) {
    this.cost = cost;
}

public int getCode() {
    return code;
}

public void setCode(int code) {
    this.code = code;
}

public int getTeam() {
    return team;
}

public void setTeam(int team) {
    this.team = team;
}

@Override
public int describeContents() {
    return 0;
}

@Override
public void writeToParcel(Parcel dest, int flags) {
    dest.writeString(fullname);
    dest.writeInt(position);
    dest.writeInt(cost);
    dest.writeInt(code);
    dest.writeInt(team);
}
}
```

Rootobject

```
package com.example.myfantasy2020;

public class Rootobject
{
    private League_obj[] leagues;

    public League_obj[] getLeagues() {
        return leagues;
    }
}
```

Splash_screen

```

package com.example.myfantasy2020;

import androidx.appcompat.app.AppCompatActivity;

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

public class Splash_screen extends AppCompatActivity {

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

        Thread t = new Thread(new Runnable() {
            @Override
            public void run() {
                try {
                    Thread.sleep(5000);
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }

                Intent i = new Intent(Splash_screen.this,
MainActivity.class);
                startActivity(i);

            }
        });

        // START COUNTING 5 SECONDS AND MOVE OT NEXT SCREEN
        t.start();

    }
}

```

Team_Adapter

```

package com.example.myfantasy2020;

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

import androidx.annotation.NonNull;

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

public class Team_Adapter extends ArrayAdapter<Team_obj> {

    // THIS CLASS HELPS INFLATE THE LIST OF LEAGUES TO LAYOUTS IN THE
    LIST VIEW

    Context context; // THE CONTEXT WHERE THE LIST VIEW PLACED IN
    ArrayList<Team_obj> objects; // LIST OF TEAMS' PROJECT

    public Team_Adapter(@NonNull Context context, int resource, int
textViewResourceId, @NonNull ArrayList<Team_obj> objects) {
        super(context, resource, textViewResourceId, objects);
        this.context = context;
        this.objects = objects;
    }

    // SET LAYOUT TO TEAM
    @NonNull
    @Override
    public View getView(int position, View convertView, @NonNull
ViewGroup parent) {
        LayoutInflater inflater = ((Activity)
context).getLayoutInflater();
        View view = inflater.inflate(R.layout.team_row, parent,
false);

        Team_obj temp = objects.get(position);

        TextView tvName =
(TextView)view.findViewById(R.id.teams_name_team_row);
        tvName.setText(temp.getTeamName());

        TextView tvPlace =
view.findViewById(R.id.place_in_league_team_row);
        int p = position+1; // SET POSITION TO EACH TEAM
        tvPlace.setText(String.valueOf(p));

        TextView teamsManager =

```



```
view.findViewById(R.id.teams_manager_team_row);
    String u = temp.getUsernameOwner();
    teamsManager.setText(u);

    TextView teamsPoints =
view.findViewById(R.id.teams_points_team_row);
    teamsPoints.setText(String.valueOf(temp.getPoints()));

    return view;
}
}
```

Team_obj

```

package com.example.myfantasy2020;

import android.os.Parcel;
import android.os.Parcelable;

import java.util.Comparator;

public class Team_obj implements Parcelable, Comparable<Team_obj>
{
    private int idTeam; // TEAM'S ID
    private int idOwner; // USER'S ID
    private String teamName; // TEAM'S NAME
    private String usernameOwner;
    private int points;

    // HELPS TO PASS ARRAY OF OBJECTS BETWEEN ACTIVITY
    protected Team_obj(Parcel in) {
        idTeam = in.readInt();
        idOwner = in.readInt();
        teamName = in.readString();
        usernameOwner = in.readString();
        points = in.readInt();
    }

    public static final Creator<Team_obj> CREATOR = new
    Creator<Team_obj>() {
        @Override
        public Team_obj createFromParcel(Parcel in) {
            return new Team_obj(in);
        }

        @Override
        public Team_obj[] newArray(int size) {
            return new Team_obj[size];
        }
    };

    public int getIdTeam() {
        return idTeam;
    }

    public void setIdTeam(int idTeam) {
        this.idTeam = idTeam;
    }

    public int getIdOwner() {
        return idOwner;
    }

    public void setIdOwner(int idOwner) {
        this.idOwner = idOwner;
    }

    public String getTeamName() {

```

```

        return teamName;
    }

    public void setTeamName(String teamName) {
        this.teamName = teamName;
    }

    public String getUsernameOwner() {
        return usernameOwner;
    }

    public void setUsernameOwner(String usernameOwner) {
        this.usernameOwner = usernameOwner;
    }

    public int getPoints() {
        return points;
    }

    public void setPoints(int points) {
        this.points = points;
    }

    // HELPS TO PASS ARRAY OF OBJECTS BETWEEN ACTIVITY
    @Override
    public int describeContents() {
        return 0;
    }

    @Override
    public void writeToParcel(Parcel dest, int flags) {
        dest.writeInt(idTeam);
        dest.writeInt(idOwner);
        dest.writeString(teamName);
        dest.writeString(usernameOwner);
        dest.writeInt(points);
    }

    // HELPS SORTING TEAM'S BY THEIR POINTS
    @Override
    public int compareTo(Team_obj o) {
        return o.getPoints() - this.getPoints();
    }
}

```

קבצי XML – עיצוב מסכים**Activity_build_team**

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/green_court"
    android:orientation="vertical"
    tools:context=".Build_team">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="15dp"
        android:layout_marginTop="15dp"
        android:fontFamily="serif-monospace"
        android:text="Your Remaining budget is:$"
        android:textColor="@color/my_white"
        android:textSize="15sp"
        android:textStyle="bold"
        android:id="@+id/tv_budget_build"></TextView>

    <LinearLayout
        android:layout_width="75dp"
        android:layout_height="110dp"
        android:orientation="vertical"
        android:layout_marginTop="20dp"

        android:layout_gravity="center">

        <ImageButton
            android:id="@+id/img_gk"
            android:layout_width="75dp"
            android:layout_height="75dp"
            android:background="@drawable/empty2"
            android:scaleType="fitXY"></ImageButton>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:fontFamily="serif-monospace"
            android:text="namepl"
            android:autoSizeTextType="uniform"
            android:textSize="20sp"
            android:id="@+id/tv_Gk">
    </TextView>

```

```
</LinearLayout>
```

```
<LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="15dp"
    android:layout_marginBottom="15dp"
    android:orientation="horizontal"
    android:layout_gravity="center">

    <LinearLayout
        android:layout_width="75dp"
        android:layout_height="110dp"
        android:orientation="vertical">

        <ImageButton
            android:id="@+id/def1"
            android:layout_width="75dp"
            android:layout_height="75dp"
            android:background="@drawable/empty2"
            android:scaleType="fitXY"></ImageButton>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:fontFamily="serif-monospace"
            android:text="namepl"
            android:autoSizeTextType="uniform"
            android:textSize="20sp"
            android:id="@+id/tv_def1"></TextView>

    </LinearLayout>
```

```
<LinearLayout
    android:layout_width="75dp"
    android:layout_height="110dp"
    android:layout_marginLeft="10dp"
    android:orientation="vertical">

    <ImageButton
        android:id="@+id/def2"
        android:layout_width="75dp"
        android:layout_height="75dp"
        android:background="@drawable/empty2"
        android:scaleType="fitXY"></ImageButton>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:fontFamily="serif-monospace"
        android:text="namepl"
```

```

        android:autoSizeTextType="uniform"
        android:textSize="20sp"
        android:id="@+id/tv_def2"></TextView>

</LinearLayout>

<LinearLayout
    android:layout_width="75dp"
    android:layout_height="110dp"
    android:layout_marginLeft="10dp"
    android:orientation="vertical">

    <ImageButton
        android:id="@+id/def3"
        android:layout_width="75dp"
        android:layout_height="75dp"
        android:background="@drawable/empty2"
        android:scaleType="fitXY"></ImageButton>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:fontFamily="serif-monospace"
        android:text="namepl"
        android:autoSizeTextType="uniform"
        android:textSize="20sp"
        android:id="@+id/tvDef3"></TextView>

</LinearLayout>

<LinearLayout
    android:layout_width="75dp"
    android:layout_height="110dp"
    android:layout_marginLeft="10dp"
    android:orientation="vertical">

    <ImageButton
        android:id="@+id/def4"
        android:layout_width="75dp"
        android:layout_height="75dp"
        android:background="@drawable/empty2"
        android:scaleType="fitXY"></ImageButton>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:fontFamily="serif-monospace"
        android:text="namepl"
        android:autoSizeTextType="uniform"
        android:textSize="20sp"
        android:id="@+id/tvDef4">
    </TextView>

</LinearLayout>

```

```
</LinearLayout>
```

```
<LinearLayout
```

```
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginBottom="15dp"
    android:orientation="horizontal"
    android:layout_gravity="center">
```

```
    <LinearLayout
```

```
        android:layout_width="75dp"
        android:layout_height="110dp"
        android:orientation="vertical">
```

```
        <ImageButton
```

```
            android:id="@+id/mid1"
            android:layout_width="75dp"
            android:layout_height="75dp"
            android:background="@drawable/empty2"
            android:scaleType="fitXY"></ImageButton>
```

```
        <TextView
```

```
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:fontFamily="serif-monospace"
            android:text="namepl"
            android:autoSizeTextType="uniform"
            android:textSize="20sp"
            android:id="@+id/tv_mid1"></TextView>
```

```
</LinearLayout>
```

```
<LinearLayout
```

```
    android:layout_width="75dp"
    android:layout_height="110dp"
    android:layout_marginLeft="10dp"
    android:orientation="vertical">
```

```
    <ImageButton
```

```
        android:id="@+id/mid2"
        android:layout_width="75dp"
        android:layout_height="75dp"
        android:background="@drawable/empty2"
        android:scaleType="fitXY"></ImageButton>
```

```
    <TextView
```

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:fontFamily="serif-monospace"
        android:text="namepl"
        android:autoSizeTextType="uniform"
```

```

        android:id="@+id/tv_mid2"
        android:textSize="20sp"></TextView>

</LinearLayout>

<LinearLayout
    android:layout_width="75dp"
    android:layout_height="110dp"
    android:layout_marginLeft="10dp"
    android:orientation="vertical">

    <ImageButton
        android:id="@+id/mid3"
        android:layout_width="75dp"
        android:layout_height="75dp"
        android:background="@drawable/empty2"
        android:scaleType="fitXY"></ImageButton>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:fontFamily="serif-monospace"
        android:text="namepl"
        android:autoSizeTextType="uniform"
        android:id="@+id/tv_mid3"
        android:textSize="20sp"></TextView>

</LinearLayout>

<LinearLayout
    android:layout_width="75dp"
    android:layout_height="110dp"
    android:layout_marginLeft="10dp"
    android:orientation="vertical">

    <ImageButton
        android:id="@+id/mid4"
        android:layout_width="75dp"
        android:layout_height="75dp"
        android:background="@drawable/empty2"
        android:scaleType="fitXY"></ImageButton>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:fontFamily="serif-monospace"
        android:text="namepl"
        android:autoSizeTextType="uniform"
        android:id="@+id/tv_mid4"
        android:textSize="20sp"></TextView>

</LinearLayout>

```



```
</LinearLayout>
```

```
<LinearLayout
```

```
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="5dp"
    android:orientation="horizontal"
    android:layout_gravity="center">
```

```
    <LinearLayout
```

```
        android:layout_width="75dp"
        android:layout_height="110dp"
        android:orientation="vertical">
```

```
        <ImageButton
```

```
            android:id="@+id/st1"
            android:layout_width="75dp"
            android:layout_height="75dp"
            android:background="@drawable/empty2"
            android:scaleType="fitXY"></ImageButton>
```

```
        <TextView
```

```
            android:id="@+id/tv_str1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:fontFamily="serif-monospace"
            android:autoSizeTextType="uniform"
            android:text="namepl"
            android:textSize="20sp"></TextView>
```

```
</LinearLayout>
```

```
<LinearLayout
```

```
    android:layout_width="75dp"
    android:layout_height="110dp"
    android:layout_marginLeft="25dp"
    android:orientation="vertical">
```

```
    <ImageButton
```

```
        android:id="@+id/st2"
        android:layout_width="75dp"
        android:layout_height="75dp"
        android:background="@drawable/empty2"
        android:scaleType="fitXY"></ImageButton>
```

```
    <TextView
```

```
        android:id="@+id/tv_str2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:fontFamily="serif-monospace"
        android:text="namepl"
        android:autoSizeTextType="uniform"
```

```
        android:textSize="20sp"></TextView>

    </LinearLayout>

    <ImageButton
        android:layout_width="75dp"
        android:layout_height="75dp"
        android:layout_marginLeft="15dp"
        android:background="@drawable/done2"
        android:id="@+id/ib_done_build"
        android:scaleType="centerCrop"
        android:padding="0dp">
    </ImageButton>

</LinearLayout>

</LinearLayout>
```

Activity_full_details_creation

```

<?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=".Full_details_creation"
    android:orientation="vertical">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:fontFamily="serif-monospace"
        android:textSize="30dp"
        android:text="Please fill in the details and submit your team
and/or League"
        android:layout_marginTop="15dp"
        android:layout_marginLeft="15dp">

    </TextView>

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter team's name"
        android:layout_marginLeft="15dp"
        android:layout_marginTop="10dp"
        android:id="@+id/et_team_name"
        android:layout_marginRight="15dp"/>

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter league's name"
        android:layout_marginLeft="15dp"
        android:layout_marginTop="10dp"
        android:id="@+id/et_league_name"
        android:layout_marginRight="15dp"/>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="25sp"
        android:text="Add friends to your league"
        android:fontFamily="serif-monospace"
        android:layout_marginTop="10dp"
        android:layout_marginLeft="15dp"
        android:id="@+id/tv_add_friend"
        >
    </TextView>

    <ImageButton
        android:layout_width="150dp"

```

```
        android:layout_height="150dp"
        android:layout_gravity="center"
        android:layout_marginTop="10dp"
        android:background="@drawable/add_people"
        android:id="@+id/add_people"/>

        <ImageButton
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:layout_marginTop="10dp"
            android:background="@drawable/submit_team"
            android:id="@+id/btn_submit_league"/>

    </LinearLayout>
```

Activity_leagues_list_view

```
<?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=".Leagues_listView"
    android:orientation="vertical">

    <ListView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/lv_leagues">
    </ListView>

</LinearLayout>
```

Activity_lv_teams_in_league

```

<?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=".Lv_Teams_In_League"
    android:orientation="vertical">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="leagues's name"
        android:gravity="center"
        android:textSize="30sp"
        android:layout_marginTop="10dp"
        android:fontFamily="serif-monospace"
        android:textStyle="bold"
        android:id="@+id/tv_league_name_teamsLST">
    </TextView>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:layout_marginTop="10dp"
        android:layout_marginLeft="10dp">

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:fontFamily="serif-monospace"
            android:text="Rank"
            android:textSize="20dp">
        </TextView>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:fontFamily="serif-monospace"
            android:text="Team and Manager"
            android:textSize="20dp"
            android:layout_marginLeft="10dp">
        </TextView>

        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="points"
            android:fontFamily="serif-monospace"
            android:layout_marginLeft="0dp"

```

```
        android:textSize="20dp"
        android:gravity="center">
    </TextView>

</LinearLayout>

<ListView
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/lv_teams_inLeague"
    android:layout_marginTop="10dp">
</ListView>

</LinearLayout>
```

Activity_main

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:orientation="vertical">

    <ImageView
        android:layout_marginTop="50dp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="@drawable/title"
        android:layout_gravity="center">
    </ImageView>

    <TextView
        android:layout_marginTop="40dp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:textSize="30sp"
        android:text="Welcome"
        android:fontFamily="serif-monospace"
        android:textStyle="bold">
    </TextView>

    <TextView
        android:layout_marginTop="80dp"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:textSize="30sp"
        android:text="Enter a username:"
        android:fontFamily="serif-monospace"
        android:textStyle="bold">
    </TextView>

    <EditText
        android:layout_marginTop="10dp"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Username"
        android:id="@+id/etNewUser"
    />

    <ImageButton
        android:layout_width="225dp"
        android:layout_height="75dp"
        android:layout_gravity="center"

```



```
android:layout_marginTop="70dp"  
android:background="@drawable/submit_blue"  
android:id="@+id/btn_submit_user"/>
```

```
</LinearLayout>
```

Activity_main_menu

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"

    android:orientation="vertical">

    <TextView
        android:id="@+id/hello_tv"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginLeft="30dp"
        android:layout_marginTop="40dp"
        android:fontFamily="serif-monospace"
        android:text="Hello"
        android:textSize="30sp"
        android:textStyle="bold" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginLeft="30dp"
        android:layout_marginTop="40dp"
        android:fontFamily="serif-monospace"
        android:text="Please select one of these options"
        android:textSize="20sp"
        android:textStyle="bold" />

    <ImageButton
        android:id="@+id/btn_show_leagues"
        android:layout_width="150dp"
        android:layout_height="150dp"
        android:layout_gravity="center"
        android:layout_marginTop="40dp"
        android:background="@drawable/show_league_btn" />

    <ImageButton
        android:id="@+id/btn_create_league"
        android:layout_width="150dp"
        android:layout_height="150dp"
        android:layout_gravity="center"
        android:layout_marginTop="25dp"
        android:background="@drawable/create_btn" />

</LinearLayout>
```

Activity_pl_players_lv

```
<?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=".Pl_players_LV"
  android:orientation="vertical">

  <SearchView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/sv_pls">
  </SearchView>

  <ListView
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/lv_pls">
  </ListView>

</LinearLayout>
```

Activity_splash_screen

```
<?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=".Splash_screen"
    android:orientation="vertical">

    <ImageView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="@drawable/trent_image">
    </ImageView>

</LinearLayout>
```

League_row

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <LinearLayout
        android:layout_width="280dp"
        android:layout_height="wrap_content"
        android:orientation="vertical">
        <TextView
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:textStyle="bold"
            android:text="league name"
            android:textSize="30sp"
            android:layout_marginLeft="15dp"
            android:layout_marginTop="5dp"
            android:id="@+id/tv_id_league_row">
        </TextView>

        <TextView
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:text="created by: "
            android:textSize="25sp"
            android:layout_marginLeft="15dp"
            android:layout_marginTop="5dp"
            android:id="@+id/tv_league_created_by_row">
        </TextView>

        <TextView
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:text="x current teams"
            android:textSize="25sp"
            android:layout_marginLeft="15dp"
            android:layout_marginTop="5dp"
            android:id="@+id/tv_amount_teams_row">
        </TextView>

    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">

        <TextView
            android:layout_width="match_parent"
            android:layout_height="90dp"

```

```
        android:textSize="25sp"
        android:text="80 pts"
        android:textStyle="bold"
        android:layout_gravity="center"
        android:gravity="center"
        android:id="@+id/tv_your_points_in_League_row">
    </TextView>

    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:textSize="20sp"
        android:text="y place"
        android:id="@+id/tv_current_place_in_league_row"
        android:layout_marginLeft="5dp">
    </TextView>

</LinearLayout>

</LinearLayout>
```

One_item_list

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="20dp"
    android:layout_marginTop="20dp">

    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:fontFamily="serif-monospace"
      android:textColor="@color/blue_menu"
      android:textSize="30sp"
      android:text="league's name">
    </TextView>

    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textSize="30sp"
      android:fontFamily="serif-monospace"
      android:text="1"
      android:layout_marginLeft="95dp">
    </TextView>

</LinearLayout>
```

Pl_player_row

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="5dp"

    android:layout_marginBottom="5dp"
    android:layout_marginRight="5dp">

    <ImageView
        android:layout_width="65dp"
        android:layout_height="65dp"
        android:id="@+id/pl_pic_pl_row"
        android:layout_marginTop="10dp">
    </ImageView>

    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:layout_marginLeft="10dp"
        android:layout_gravity="center">
        <TextView
            android:layout_width="200dp"
            android:layout_height="wrap_content"
            android:text="pl player full name"
            android:textSize="20sp"
            android:id="@+id/pl_fullname_pl_row">
        </TextView>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="teams name"
            android:layout_marginTop="10dp"
            android:id="@+id/teams_name_pl_row">
        </TextView>

    </LinearLayout>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="$"
        android:layout_gravity="center"
        android:id="@+id/pl_cost_pl_row"
        android:textSize="15sp"
        android:layout_marginLeft="5dp">
    </TextView>

```



```
<ImageView
    android:id="@+id/pl_team_pic_pl_row"
    android:layout_width="55dp"
    android:layout_height="55dp"
    android:layout_marginLeft="15dp"
    android:layout_marginTop="10dp"
    android:layout_marginRight="20dp">
</ImageView>

</LinearLayout>
```

Team_row

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="10dp"
    android:layout_marginLeft="10dp">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="1"
        android:textSize="25sp"
        android:id="@+id/place_in_league_team_row"
        android:layout_marginLeft="25dp"
        android:layout_marginTop="5dp">
    </TextView>

    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:layout_marginLeft="30dp"
        >
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:text="team's name"
            android:textStyle="bold"
            android:textSize="25sp"
            android:id="@+id/teams_name_team_row">
        </TextView>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="team's manager"
            android:layout_marginTop="10dp"
            android:textSize="20sp"
            android:gravity="center"
            android:id="@+id/teams_manager_team_row">
        </TextView>

    </LinearLayout>

    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:text="80"
        android:textSize="25sp"
        android:gravity="right"

```

```
        android:layout_marginRight="35dp"
        android:id="@+id/teams_points_team_row">
    </TextView>
</LinearLayout>
```

צד שרת**apiPremierLeague.py**

```
import json
import requests
import unicode
from datetime import date
import datetime

class PremierLeague:

    def __init__(self):
        self.lastUpdated = None
        self.current_user_data = self.bringForUser()
        self.all_data = json.loads(self.bring_players())

    # RETURNS THE FULL JSON DATA OF ALL THE PLAYERS FROM PREMIER LEAGUE API
    def bring_players(self):

        r = "https://fantasy.premierleague.com/api/bootstrap-static/"
        req = requests.get(r)
        self.lastUpdated = date.today()
        return req.text

    # RETURNS THE SUM OF POINTS OF ALL THE PLAYERS CODES
    def calc11(self, list11):

        teamspoints = 0
```

```

for i in list11:
    teamspoints += self.dict_points.get(str(i))

return teamspoints

# RETURNS THE JSON OF ALL THE PLAYERS OVER 10 TOTAL POINTS
# WITH THE NECESSARY PROPERTIES
def bringForUser(self):

    if date.today() == self.lastUpdated:
        return self.current_user_data

    s = self.bring_players()
    s = json.loads(s)
    self.all_data = s
    self.update_dict()

    init_json = '{"plPlayers":[]}'
    data = json.loads(init_json)

    for i in s["elements"]:

        if i["total_points"] > 10:
            f = i["first_name"] + " " + i["second_name"]
            ffixed = unicode.decode(f)
            data["plPlayers"].append({'fullname': ffixed,
                                      'position': i["element_type"],
                                      'cost': i["now_cost"],
                                      'code': i["code"],
                                      'team': i["team_code"]})

    return data

```

```
# RETURNS THE USL OF PLAYER PHOTO FROM THE API

def bringPlayerPhoto(self):

    r = "https://platform-static-
files.s3.amazonaws.com/premierleague/photos/players/110x140/p152760.png"

    return r


# RETURNS THE USL OF TEAM'S PHOTO FROM THE API

def bringTeamPhoto(self):

    return "https://fantasy.premierleague.com/dist/img/badges/badge_14_40.png"


# UPDATED THE DICTIONARY WITH WHE PLAYER'S POINTS

def update_dict(self):

    self.dict_points = {}

    for i in self.all_data["elements"]:

        self.dict_points.update({str(i["code"]): int(i["total_points"])})
```

cloud_messaging.py

```

# Copyright 2018 Google Inc. All Rights Reserved.
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
#   http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

from __future__ import print_function

import datetime
from firebase_admin import messaging
from firebase_admin import credentials
import firebase_admin

class NotifyClient:

    def send_multicast(self, registration_tokens, league_owner, league_name):
        # [START send_multicast]
        # Create a list containing up to 500 registration tokens.
        # These registration tokens come from the client FCM SDKs.

        message = messaging.MulticastMessage(
            data={'league_owner': league_owner, 'league_name': league_name},
            tokens=registration_tokens,
        )

        message = self.android_message(registration_tokens, league_owner, league_name)
        response = messaging.send_multicast(message)
        # See the BatchResponse reference documentation
        # for the contents of response.
        print('{0} messages were sent successfully'.format(response.success_count))
        # [END send_multicast]

    def android_message(self, registration_tokens, league_owner, league_name):
        # [START android_message]
        message = messaging.MulticastMessage(
            android=messaging.AndroidConfig(
                ttl=datetime.timedelta(seconds=3600),
                priority='normal',
            )
        )

```

```
notification=messaging.AndroidNotification(
    title=str(league_owner) + " invited you!",
    body="to: " + str(league_name),
    # icon='stock_ticker_update',
    # color='#f45342'
),
),
tokens=registration_tokens,
# topic='industry-tech',
)
# [END android_message]
return message
```


getFromClient.py

```

from flask import Flask
import users
import mysql.connector
import league
import json
import apiPremierLeague
import teams
from firebase_admin import credentials
import firebase_admin
import cloud_messaging

app = Flask(__name__)

# INITIALIZING VARIABLES

# INITIALIZE MYSQL CONNECTION TO THE SCHEMA
mydb = mysql.connector.connect(host="localhost",
                               user="NoamK",
                               passwd="NoamKDB",
                               database="my_fantasy_pl")

# FIREBASE CONNECTION TO THE APP
if not len(firebase_admin._apps):
    cred = credentials.Certificate("my-fantasy-2020-firebase-adminsdk-n2h7g-aecca56929.json")
    default_app = firebase_admin.initialize_app(cred)

# LOAD PLAYERS FROM PREMIER LEAGUE API
c = apiPremierLeague.PremierLeague()

@app.route("/noam/ok")
def getOk():

    print("i got it")

    return "hello noam"

# FUNCTION GETS NEW TOKEN AND USERNAME TO EDIT TOKEN
# RETURNS ANSWER THAT THE TOKEN IS EDITED
@app.route("/editToken/<username>/<new_token>")
def editToken(username, new_token):
    u = users.User()
    u.edit_token(mydb, username, new_token)

```

```

return "updated"

# GETS A USERNAME
# RETURN "valid name" THE USER EXISTS IN USERS TABLE
@app.route("/isUsernameExists/<username>")
def isUsernameExists(username):
    u = users.User()

    if not u.is_exist(mydb, username):
        return "valid name"
    return "name already exists"

# GETS USERNAME TOKEN AND ADDS NEW USER TO USERS TABLE
@app.route("/addUser/<username>/<token>")
def addUser(username, token):
    u = users.User()

    if u.add_user(username, mydb, token):
        return "added"
    return "exists"

# GETS A USERNAME
# RETURNS JSON OF ALL THE LEAGUES THAT THE USERNAME INVITED FOR OR CREATED
@app.route("/getMyLeagues/<username>")
def getUserLeagues(username):
    l = league.League()
    listLeagues = l.get_my_leagues(mydb, username)
    return json.dumps(listLeagues)

# GETS LEAGUE'S CREATOR, LEAGUE NAME, TEAM'S NAME, LIST OF 11 CODES OF SELECTED
PLAYERS AND LIST OF USERS
# RETURNS MESSAGE THAT THE LEAGUE ADDED TO THE DATABASE
@app.route("/addLeague/<username>/<leagueName>/<teamName>/<teamList>/<usersList
>")
def addLeague(username, leagueName, teamName, teamList, usersList):

    l = league.League()

    usersList = json.loads(usersList)
    teamList = json.loads(teamList)

    l.add_league(mydb, username, leagueName, teamName, teamList, usersList)

    u = users.User()

```

```

tokens = u.getTokenByUser(mydb, usersList)

n = cloud_messaging.NotifyClient()

n.send_multicast(tokens, username, leagueName)
return "added"

# GETS USERNAME, LEAGUE ID, TEAM'S NAME, CODES
# RETURNS MESSAGE THAT THE LEAGUE ADDED TO THE DATABASE
@app.route("/addTeamToLeague/<username>/<leagueId>/<teamName>/<teamList>")
def addTeamToLeague(username, leagueId, teamName, teamList):

    u = users.User()
    id_u = u.nameToId(mydb, username)
    teamList = json.loads(teamList)

    t = teams.Team()
    t.add_team(mydb, id_u, teamName, leagueId, teamList)

    return "added"

# RETURNS THE LIST THE PREMIER LEAGUE PLAYERS
@app.route("/getPLPlayers")
def getPLPlayers():
    return json.dumps(c.bringForUser())

# RETURNS THE URL OF PLAYER'S PHOTO
@app.route("/getPlayerPhoto")
def getPlayerPhoto():
    return c.bringPlayerPhoto()

# RETURNS THE URL OF TEAMS'S PHOTO
@app.route("/getTeamPhoto")
def getTeamPhoto():
    return c.bringTeamPhoto()

"""
# GETS TEAM'S ID
# RETURNS THE CODES OF THE TEAM'S PLAYERS
@app.route("/getPLplayersById/<teamId>")
def getPLplayersById(teamId):
    t = teams.Team()

```

```
        return t.getPlayers(mydb,teamId)
    """

# GETS A USERNAME
# RETURNS THE LIST OF OTHER USERNAME'S IN THE DATABASE
@app.route("/getOtherUsers/<you>")
def getOtherUsers(you):
    u = users.User()
    return json.dumps(u.getOthers(mydb, you))

# RETURNS TEAM'S CODES BY TEAM ID
@app.route("/getSomeoneTeam/<idTeam>")
def getSomeoneTeam(idTeam):
    t = teams.Team()
    return json.dumps(t.getPlayers(mydb,idTeam))

if __name__ == '__main__':
    app.run(host="0.0.0.0")
```

league.py

```

import mysql.connector
import teams
import users
import json

class League:

    # ADD NEW LEAGUE TO LEAGUE TABLE
    # ADD THE USERS WHO INVITED TO THE LEAGUE
    def add_league(self, dbConnection, username, league_name, team_name,
team_list, users_list):

        u = users.User()

        id_user = u.nameTold(dbConnection, username)

        my_cursor = dbConnection.cursor()
        q = "INSERT INTO `my_fantasy_pl`.`league` (`name`, `idCreator`) VALUES
('"+league_name+"', '"+str(id_user)+"');"
        my_cursor.execute(q)
        dbConnection.commit()

        id_league = my_cursor.lastrowid

        t1 = teams.Team()

        t1.add_team(dbConnection, id_user, team_name, id_league, team_list)

        q1 = "insert into my_fantasy_pl.mnleagueuser(idLeague,idUser) values(%s, %s);"
        vals = (id_league, id_user)
        my_cursor.execute(q1, vals)
        dbConnection.commit()

        for i in users_list:
            id_guest = u.nameTold(dbConnection,i)
            vals = (id_league, id_guest)
            my_cursor.execute(q1, vals)
            dbConnection.commit()
        return True

    # RETURNS THE LEAGUES THAT THE USER SHOULD PARTICIPATE IN
    # GETS A USERNAME

```

```

# RETURNS LEAGUE DETAILS THAT THE USER SHOULD PARTICIPATE IN
def get_my_leagues(self, dbConnection, name):
    u1 = users.User()
    my_cursor = dbConnection.cursor()
    idu = u1.nameTold(dbConnection, name)

    s = "select * from (select tbl.idLeague,tbl.idUser,league.name,league.idCreator "
    \
        "from (select * from my_fantasy_pl.mnleagueuser where idUser = "
    "+str(idu)+") as tbl " \
        "inner join league on tbl.idLeague = league.idLeague) as sub_q " \
        "inner join user on sub_q.idCreator = user.idUser;"

    init_json = '{"leagues":[]}'
    data = json.loads(init_json)
    my_cursor.execute(s)
    ltemp = my_cursor.fetchall()

    for i in ltemp:
        q3 = "select * from team inner join user on team.idOwner = user.idUser where "
        idLeague = " + str(i[0]) + ";"
        my_cursor.execute(q3)
        teams_in_league = my_cursor.fetchall()
        jsonTeams = '{"teamslst":[]}'
        jTeams = json.loads(jsonTeams)
        for j in teams_in_league:
            jTeams['teamslst'].append(
                {'idTeam': j[0], 'idOwner': j[1], 'teamName': j[3], 'usernameOwner': j[6],
                "points": j[4]})

        league_name = i[2]
        league_owner = i[3]
        league_owner_name = i[5]
        data['leagues'].append(
            {'name': league_name, 'idLeague': i[0], 'idOwner': league_owner,
            'creator_username': league_owner_name,
            'teams': jTeams['teamslst']})

    return data

```

teams.py

```

import apiPrenierLeauge
import json

import getFromClient
import users

class Team:

    # ADDS TEAM TO THE DATABASE AND THEIR PLAYERS
    def add_team(self, dbConnection, id_user, team_name, id_leauge,
starting11_codes):

        points = self.calc_points(starting11_codes)
        my_cursor = dbConnection.cursor()
        q = "INSERT INTO `my_fantasy_pl`.`team` (`idOwner`, `idLeague`, `teamName`,
`points`)" \
        " VALUES (%s, %s, %s,%s);"
        vals = (str(id_user), str(id_leauge), team_name, str(points))
        my_cursor.execute(q, vals)
        dbConnection.commit()

        team_id = my_cursor.lastrowid

        for i in starting11_codes:
            q1 = "insert into mnplayersteams(idPlayer,idTeam) values(%s,%s);"
            vals1 = (str(i),str(team_id))
            my_cursor.execute(q1, vals1)
            dbConnection.commit()

    # CALCULATES POINTS FROM THE LIST OF CODES
    def calc_points(self,list11):
        a = getFromClient.c
        return a.calc11(list11)

    # RETURNS THE TEAM'S PLAYERS BY ID
    def getPlayers(self, dbConnection, idTeam):
        my_cursor = dbConnection.cursor()

        my_cursor.execute("select idPlayer from my_fantasy_pl.mnplayersteams where
mnplayersteams.idTeam =" + str(idTeam) + ";" )

        res = my_cursor.fetchall()

```

```
#init_json = '{"plPlayers":[]}'  
#data = json.loads(init_json)
```

```
lst = []
```

```
for i in res:  
    lst.append(int(i[0]))
```

```
if lst == [] or lst == None:  
    lst = "Empty"
```

```
return lst
```


users.py

```

import mysql.connector
import json
class User:

    def __init__(self):
        x = 3

    # ADD USER TO THE DATABASE WITH THE TOKEN
    def add_user(self, name, dbConnection, token):

        if self.is_exist(dbConnection, name):
            return False

        my_cursor = dbConnection.cursor()
        my_cursor.execute("insert into my_fantasy_pl.user (name, firebase_token)
VALUES ('"+name+"', '"+token+"')")

        dbConnection.commit()

        return True
    """
    # RETURNS THE USERS IN THE USERS TABLE
    def get_users(self, dbConnection):
        my_cursor = dbConnection.cursor()

        my_cursor.execute("select * from my_fantasy_pl.user")

        res = my_cursor.fetchall()

        return res
    """
    # RETURN IF A USERNAME EXISTS IN THE DATABASE
    def is_exist(self, dbConnection, name):

        my_cursor = dbConnection.cursor()

        my_cursor.execute("select * from my_fantasy_pl.user where name =
 '"+name+"'")

        res = my_cursor.fetchall()

        return res != []

```

```

# RETURNS A USERNAME BY ID
def nameTold(self, dbConnection, name):
    my_cursor = dbConnection.cursor()

    my_cursor.execute("select my_fantasy_pl.user.idUser from my_fantasy_pl.user
where name = '"+name+"'")

    res = my_cursor.fetchall()

    return res[0][0]

# RETURNS ID BY USERNAME
def idToName(self, dbConnection, idUser):
    my_cursor = dbConnection.cursor()

    my_cursor.execute("select my_fantasy_pl.user.name "
                      "from my_fantasy_pl.user "
                      "where my_fantasy_pl.user.idUser = '"+str(idUser)+"'")

    res = my_cursor.fetchall()

    return res[0][0]

# RETURNS THE USERNAMES WHO AREN'T EQUAL TO THE USERNAME
def getOthers(self, dbConnection, user):
    my_cursor = dbConnection.cursor()
    q = "select name from user where name != '"+user+"';"
    my_cursor.execute(q)

    res = my_cursor.fetchall()

    init_json = '{"other_users":[]}'
    data = []

    for i in res:
        x = i[0]
        data.append(x)

    return data

# UPDATES THE TOKEN BY THE USERNAME
def edit_token(self, dbConnection, username, newToken):
    vals = (newToken, username)

```

```

q = "UPDATE user SET user.firebase_token = %s WHERE user.name = %s ;"
my_cursor = dbConnection.cursor()
my_cursor.execute(q, vals)
dbConnection.commit()
return

# RETURNS A LIST OF TOKENS BY THE LISTS OF USERNAMES
def getTokenByUser(self, dbConnection, lstUsers):
    lstTokens = []
    my_cursor = dbConnection.cursor()

    for i in lstUsers:
        q = 'select firebase_token from user where name = "' + str(i) + '";'
        my_cursor.execute(q)
        token = my_cursor.fetchall()
        val = token[0][0]
        lstTokens.append(val)
    return lstTokens

def main():
    mydb = mysql.connector.connect(host="localhost",
                                   user="NoamK",
                                   passwd="NoamKDB",
                                   database="my_fantasy_pl")

    u = User()

    u.add_user("ggGoodx", mydb)
    lst = u.get_users(mydb)

    print(lst)

    print(u.nameTold(mydb, "ggGood"))
    print(u.idToName(mydb, 16))

if __name__ == "__main__":
    main()

```