****

**פרויקט גמר בהנדסת תוכנה**

**תכנון ותכנות מערכות בטלפונים ניידים באנדרואיד**

שז"ר בת ים

אופיר צבאן

214317315

יורם לס

5.15.24 תשפ"ד

**תוכן עניינים:**

מבוא

תיאום תחום הידע

ארכיטקטורה

מימוש הפרויקט

ממשק משתמש

רפלקציה

ביבליוגרפיה

נספחים

**מבוא:**

**הקדמה –**

שם הפרויקט: "טאקי" / "TAKI".

טאקי הוא משחק קלפים ישראלי נפוץ שפותח ויצא לאור על ידי מפתח המשחקים חיים שפיר.

חוקי המשחק:

הקלפים מורכבים מקלפי מספר- מאחד עד תשע. וקלפים מיוחדים – פלוס, עצור, שנה כיוון, טאקי, 2+, 3+, 3+ שבור, סופר טאקי, מלך ומשנה צבע. כל אחד מהקלפים המיוחדים מבצע פעולה מסוימת.

בתחילת המשחק כל שחקן מקבל שמונה קלפים, שאר הקלפים מונחים בערימה כשפניהם כלפי מטה. וקלף בודד נוסף שנמצא על יד החפיסה, פונה מעלה, ומטרתו לשמש כקלף מנחה (חייב להיות קלף מספר).

המטרה היא להפטר מכל הקלפים ראשון. ניתן להפטר מקלף כאשר מניחים אותו בערימה מעל הקלף המנחה. אך רוב הקלפים יש מגבלות וניתן להניחם אך ורק על קלף בעל אותו צבע או אותה צורה. בחפיסה ישנם גם קלפים מיוחדים כגון "מלך", "משנה צבע", "שלוש+", "שלוש+ שבור" ו- "סופר טאקי" שניתן להניחם ללא קשר לצבע או צורה. אך מעל הקלף "2+" ניתן להניח אך ורק "2+" או "מלך".

[https://upload.wikimedia.org/wikipedia/he/thumb/e/ee/TAKI-Plus2.PNG/30px-TAKI-Plus2.PNG](https://he.wikipedia.org/wiki/%D7%A7%D7%95%D7%91%D7%A5:TAKI-Plus2.PNG)2+ : הקלף מצווה על השחקן הבא לקחת 2 קלפים מהקופה במידה ואין לו 2+ או מלך. אם יש לו קלף של 2+ הוא מניח אותו והשחקן שאחריו צריך להניח 2+ או מלך או לקחת ארבעה קלפים.

[https://upload.wikimedia.org/wikipedia/he/thumb/6/64/TAKI-Plus.PNG/30px-TAKI-Plus.PNG](https://he.wikipedia.org/wiki/%D7%A7%D7%95%D7%91%D7%A5:TAKI-Plus.PNG)

פלוס מאפשר לשחקן להוסיף קלף נוסף באותו הצבע או באותה הצורה

[https://upload.wikimedia.org/wikipedia/he/thumb/d/df/TAKI-Stop.PNG/30px-TAKI-Stop.PNG](https://he.wikipedia.org/wiki/%D7%A7%D7%95%D7%91%D7%A5:TAKI-Stop.PNG)

עצור מדלג על השחקן הבא.

[https://upload.wikimedia.org/wikipedia/he/thumb/2/22/TAKI-Dir.PNG/30px-TAKI-Dir.PNG](https://he.wikipedia.org/wiki/%D7%A7%D7%95%D7%91%D7%A5:TAKI-Dir.PNG)

שנה כיוון משנה את כיוון המשחק.

[https://upload.wikimedia.org/wikipedia/he/thumb/b/bc/TAKI-Color.PNG/30px-TAKI-Color.PNG](https://he.wikipedia.org/wiki/%D7%A7%D7%95%D7%91%D7%A5:TAKI-Color.PNG)

שנה צבע משנה את הצבע

[https://upload.wikimedia.org/wikipedia/he/thumb/6/61/TAKI-Crown.PNG/30px-TAKI-Crown.PNG](https://he.wikipedia.org/wiki/%D7%A7%D7%95%D7%91%D7%A5:TAKI-Crown.PNG)

מבטל פקודות של שאר הקלפים (מלבד 3+ / 3+ שבור) ומאפשר להניח עוד קלף (לא מחייב).

[https://upload.wikimedia.org/wikipedia/he/thumb/0/08/TAKI-TAKI.PNG/30px-TAKI-TAKI.PNG](https://he.wikipedia.org/wiki/%D7%A7%D7%95%D7%91%D7%A5:TAKI-TAKI.PNG)

טאקי מאפשר להניח את כל הקלפים מאותו הצבע (זהים לצבע של הטאקי) בתור אחד.

[https://upload.wikimedia.org/wikipedia/he/thumb/7/7b/TAKI-Super.PNG/30px-TAKI-Super.PNG](https://he.wikipedia.org/wiki/%D7%A7%D7%95%D7%91%D7%A5:TAKI-Super.PNG)

סופר טאקי פועל כמו טאקי רגיל אך הוא מקבל את הצבע שיש בערימה.

[https://upload.wikimedia.org/wikipedia/he/thumb/4/4c/TAKI-Plus3.PNG/30px-TAKI-Plus3.PNG](https://he.wikipedia.org/wiki/%D7%A7%D7%95%D7%91%D7%A5:TAKI-Plus3.PNG)שלוש פלוס מחייב את כל המשתתפים לקחת שלושה קלפים נוספים אלא אם למישהו יש את השלוש פלוס השבור. במידה וכן, הוא מניח אותו ואז מי שהניח את השלוש פלוס חייב לקחת שלושה קלפים.

[https://upload.wikimedia.org/wikipedia/he/thumb/6/60/TAKI-Break3.PNG/30px-TAKI-Break3.PNG](https://he.wikipedia.org/wiki/%D7%A7%D7%95%D7%91%D7%A5:TAKI-Break3.PNG)שלוש פלוס שבור נועד להגן מפני שלוש פלוס. אם מונח ללא קשר לשלוש פלוס, אותו אדם שהניח אותו חייב לקחת שלושה קלפים.

המשחק מסתיים כאשר מישהו סיים את כל הקלפים שלו.

קהל היעד הוא אנשים בכל הגילאים, זהו משחק אסטרטגיה ומזל, כיף, מצחיק, מעביר את הזמן.

**תהליך המחקר וקשיים –**

לא ערכתי מחקר מעמיק לפני תחילת העבודה, אלא ביצעתי אותו תוך כדי. יצרתי לי קובץ עם כל המשימות שעליי לסיים וחלק מרוב המשימות כלל גם מחקר.

כל פעם שנתקלתי בתקלה מסוימת או מבוי סתום חקרתי על הסיבות לתקלה ועל פתרונות.

תחילה עיצבתי את ה- xml של המסכים עד שהגיע הצורך בשימוש בענן ותיכנתי אותם.

על מנת לשמור על המידע של המשתמש הנוכחי חקרתי על ISharedPreferances.

לאחר מכן היה עליי ליצור NetwrokObject למשתמש ולחדר. חקרתי על פעולות בעזרתן ניתן לגשת למידע הזה, לשנות ולקבל אותו.

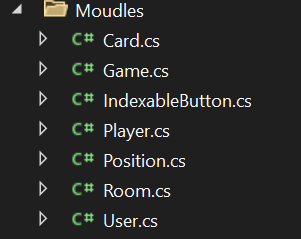
כשהגעתי לחלק של תכנות המשחק, הייתי צריכה ליצור את הגרפיקה, וגם כאן ערכתי מחקר מכיוון שהייתי צריכה ליצור את הכפתורים בצורה דינמית. נעזרתי ב- ChatGpt שלא כל כך עזר, ועל כן התבססתי על פעולה דומה שעשיתי באחד הפרויקטים הישנים ועל אתרים בגוגל.

מכיוון שהפרויקט שלי מכיל המון קלפים, שהם תמונות, הכנסתי את הקבצים הללו לתיקיית drawable, אך משום מה התוכנה לא מצאה את התמונות, ומכיוון שאחת העבודות שעשינו בכיתה גם הייתה מבוססת על קלפים דינמיים הצלחתי למצוא את הבעיה, העברתי את הקלפים ל- Assets.

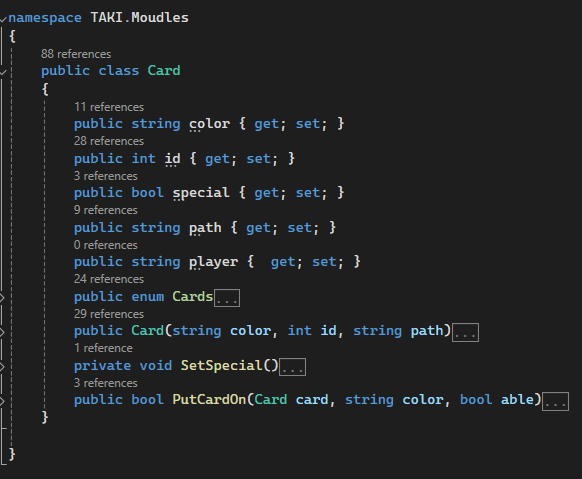
בתכנות המשחק עצמו, נתקלתי בהרבה תקלות של תרדים וכדי לפתור אותן נעזרתי בגוגל, לבסוף יצרתי listeners.

המחקר הכי טוב לדעתי היה העבודה עצמה – כאשר עשיתי debug והייתה לי ההזדמנות לראות מה קורה בפועל. רק אז הבנתי מה באמת התקלה וזה אפשר לי לחשוב על פתרונות לבעיה.

**תיאור תחום הידע:**

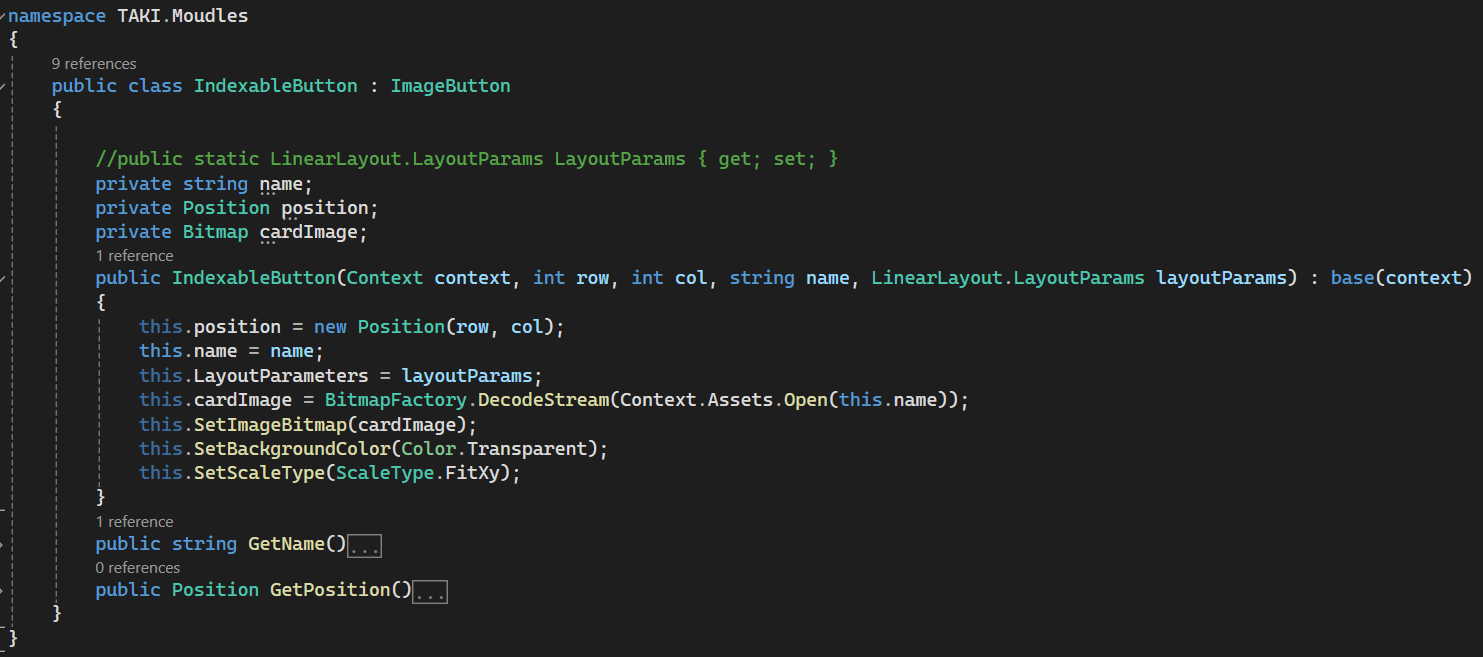
אובייקטים נחוצים:

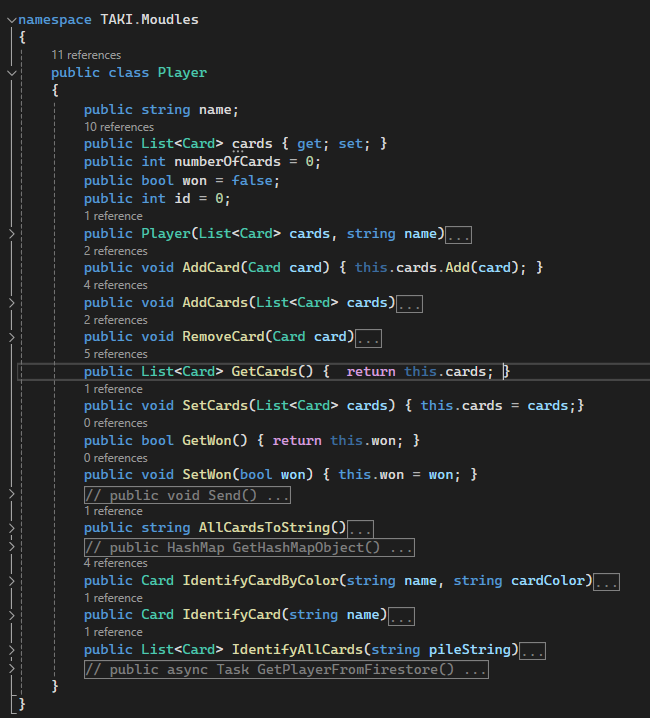
קלף:



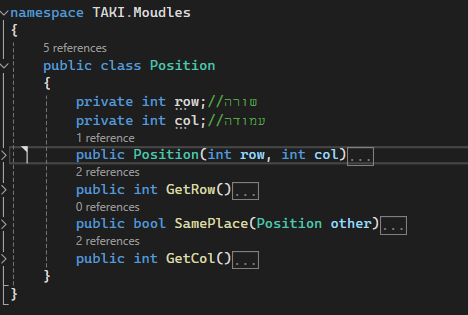
המשחק:

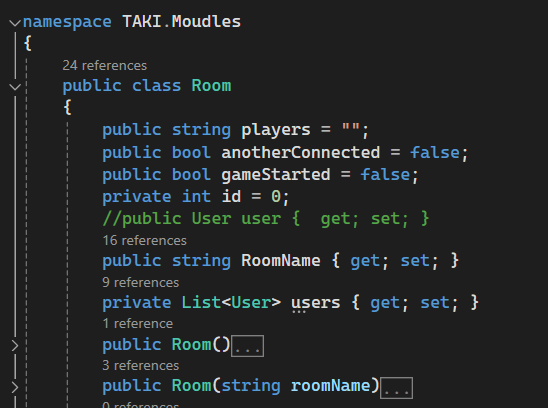


כפתור דינמי:

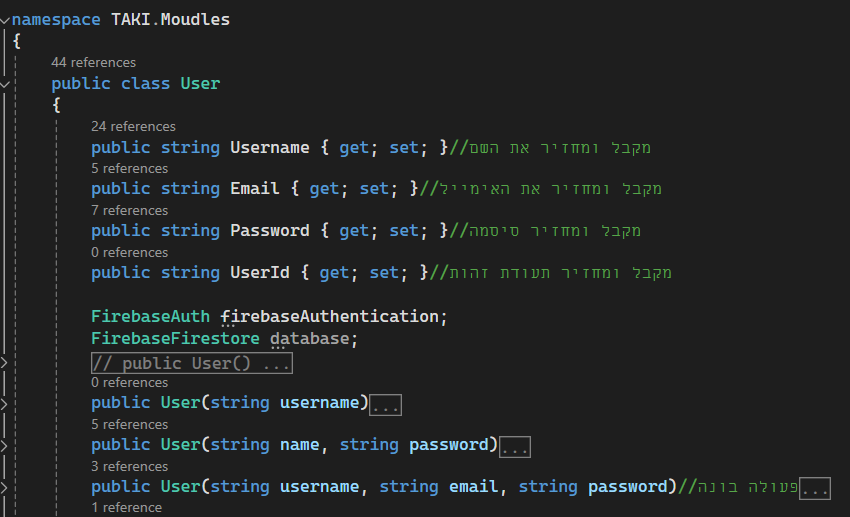
שחקן:

מיקום:



חדר:

משתמש:



הסבר קצר על המחלקות-

קלף: מחלקה שמאפשרת לשמור מידע על כל קלף – נתיב לתמונה, צבע, מספר. בנוסף בודקת איזה קלפים ניתן להניח על אותו הקלף.

משחק: מחלקה שאחראית על כל המשחק עצמו.

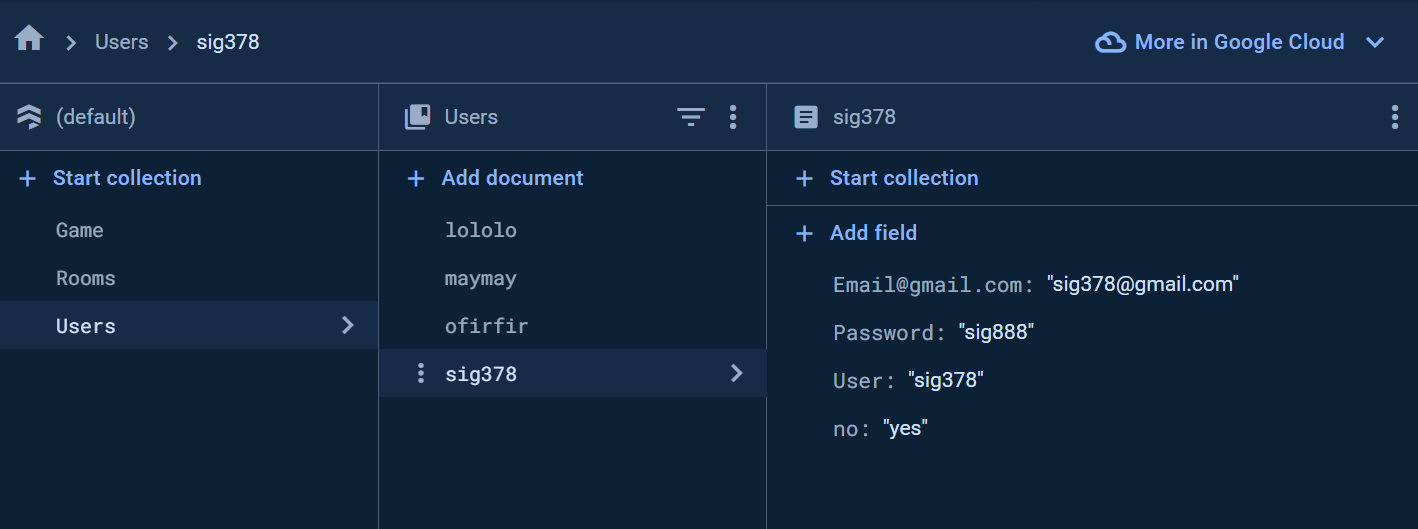
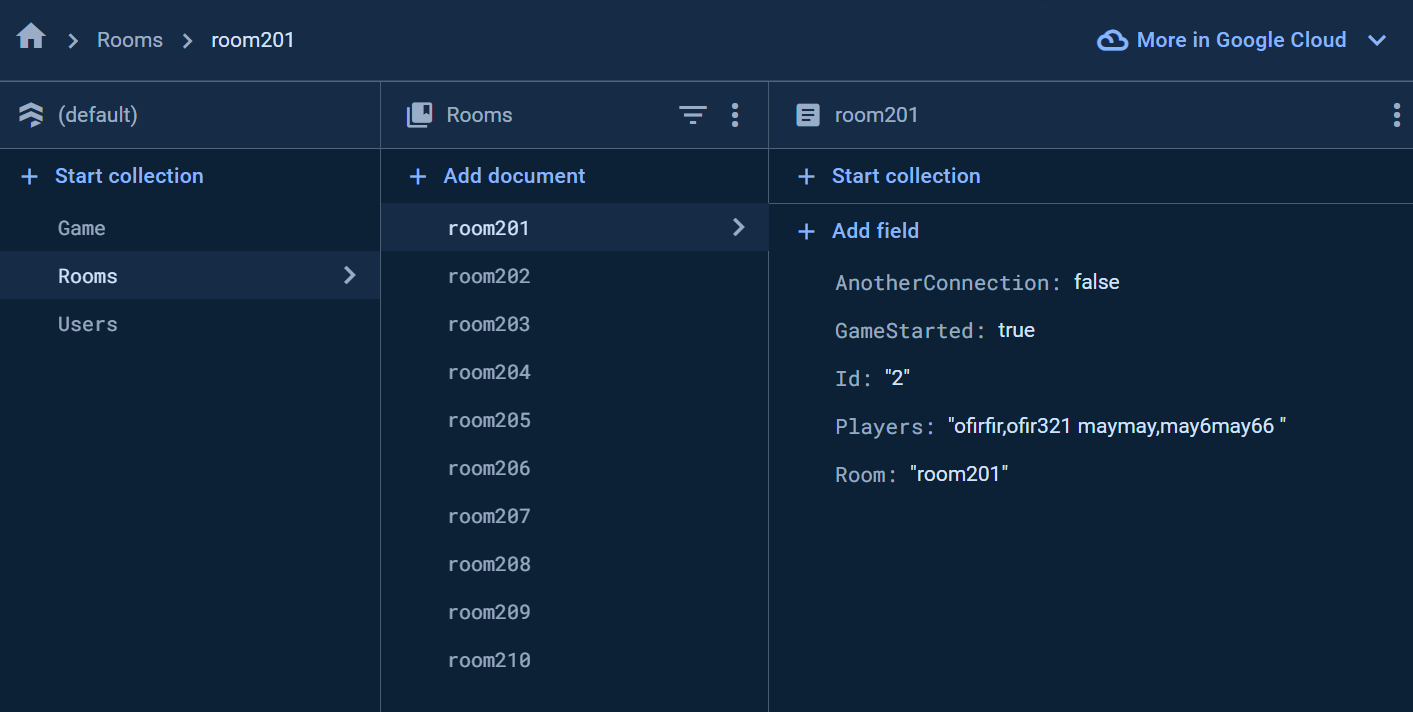
כפתור דינמי: מחלקה שנוצרה בשביל הגרפיקה של המשחק, על מנת ליצור קלפים שניתן ללחוץ עליהם.

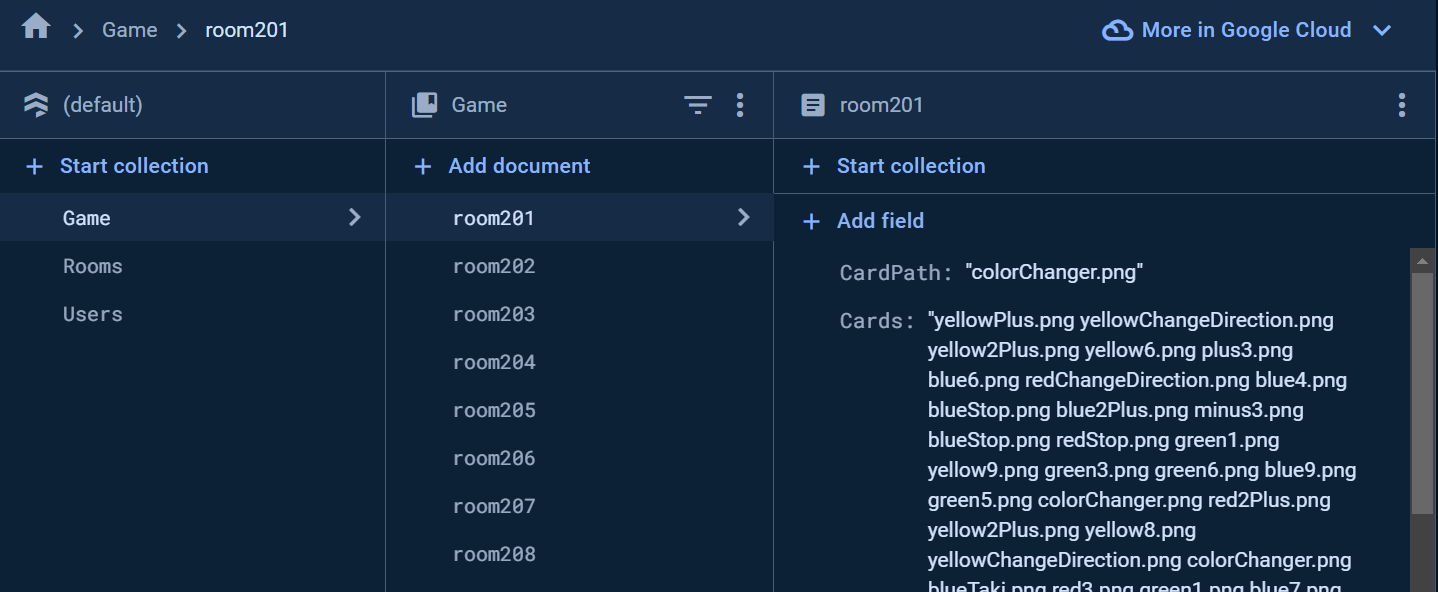
שחקן: לשחקן יש שם זהה למשתמש אך יש לו תכונות אחרות והוא איננו נמצא בענן. השחקן מתייחס ספציפית למשחק הנוכחי ולא לכל האפליקציה.

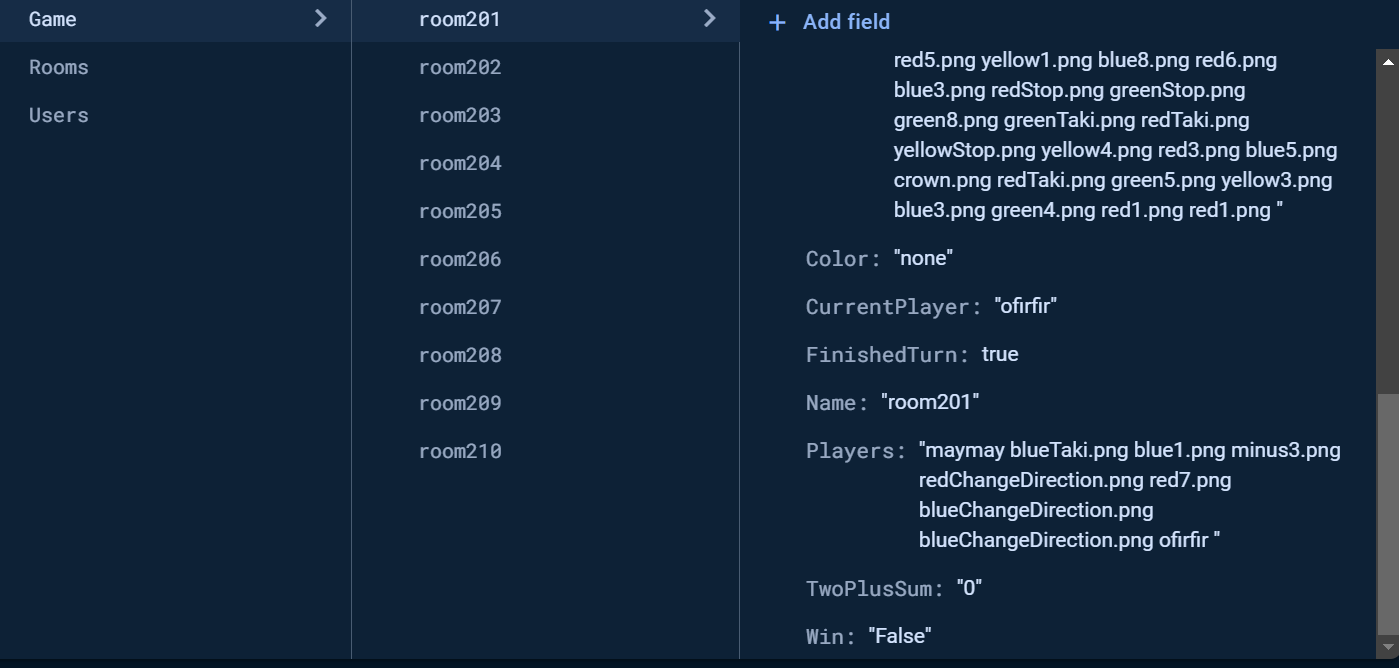
מיקום: מחלקת עזר שנועדה לעזור ליצור את הכפתורים הדינמיים.

חדר: מחלקה שיוצרת משחק. על מנת לשחק חייב להיכנס לחדר, המחלקה הזו שומרת את שמות המשתמשים שהצטרפו למשחק, וכל מיני אלמנטיי עזר כמו "האם המשחק התחיל".

משתמש: מחלקה ששומרת את הסיסמא, את האימייל ואת שם המשתמש של השחקן.

טבלאות מהענן:





תרשים מסכים:

Register

Rules

Main Activity

Game

Start

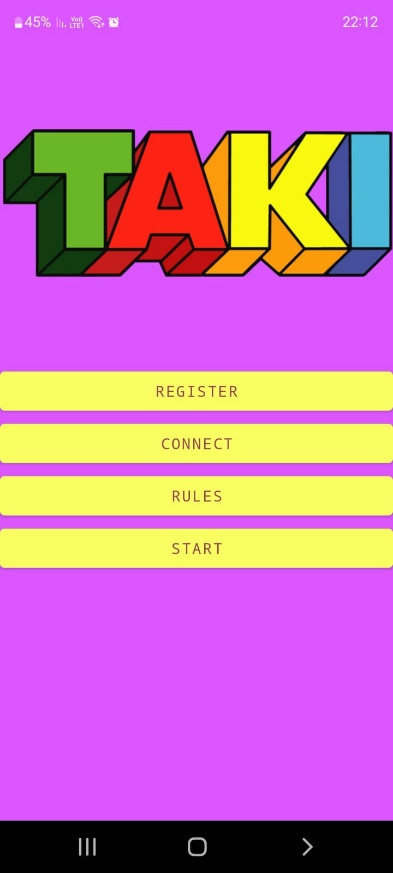
Connection

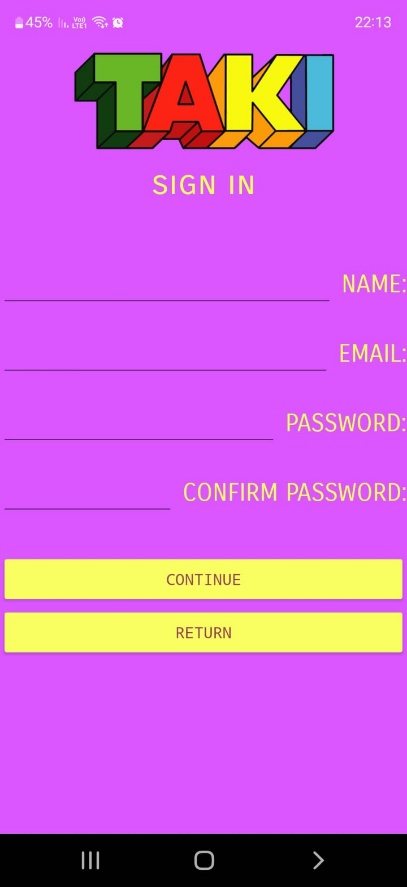
Win / Lose

Wait Room

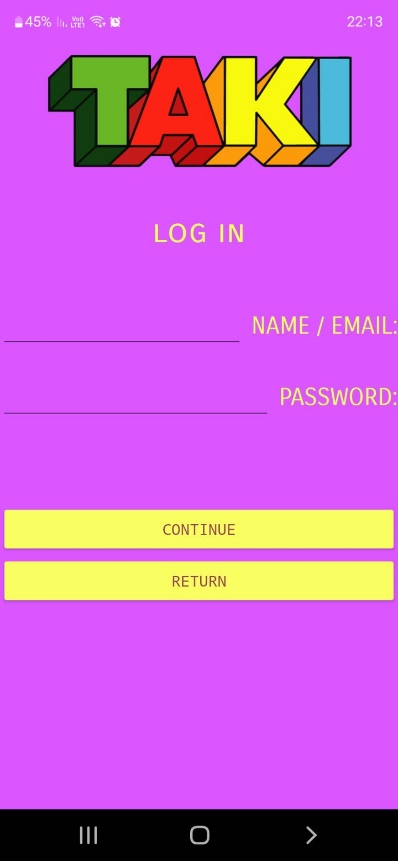
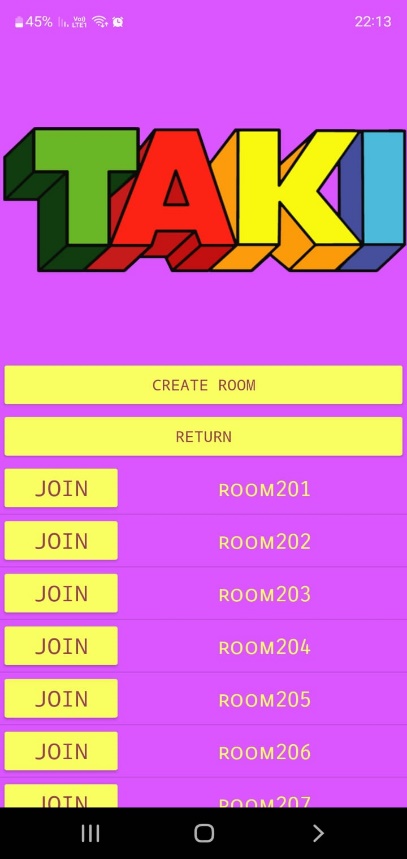
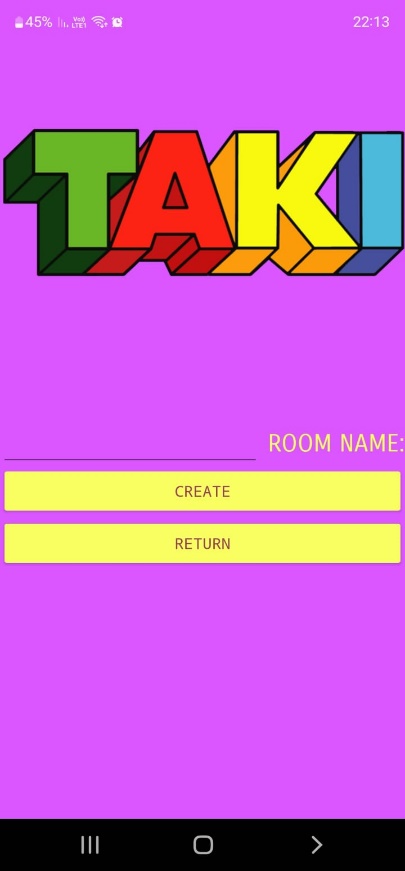
Join Room

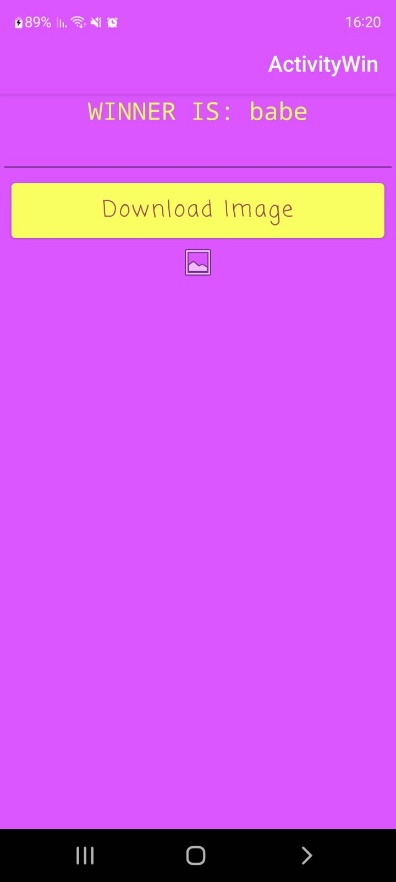
Create Room

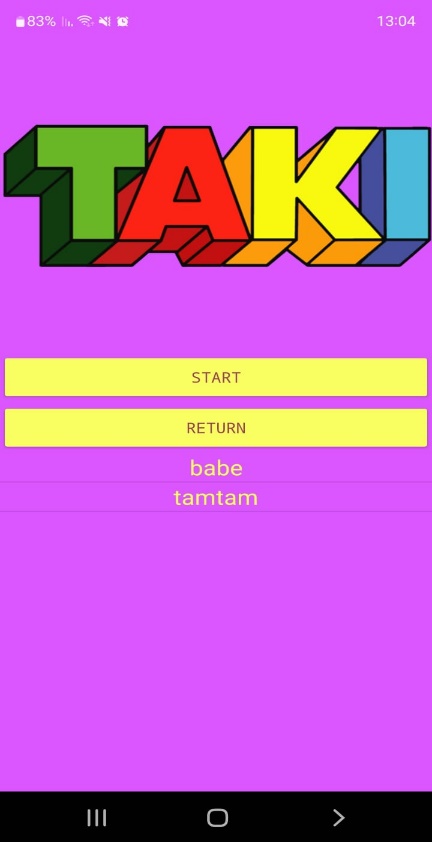
Main Activity: Rules: Register:

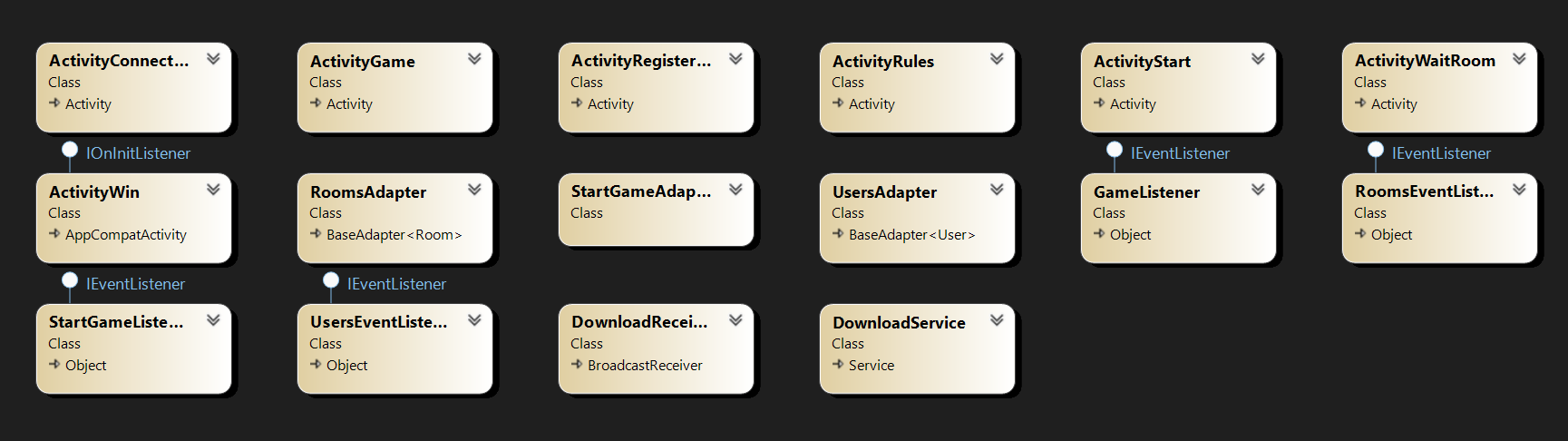


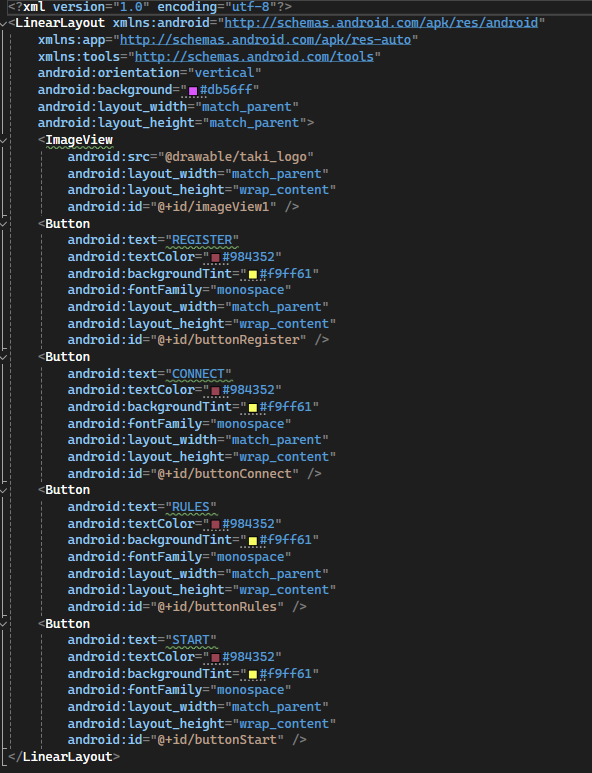
Connection: Start: Create Room:



Wait Room: Game: Win/Lose:





ארכיטקטורה:

**Main Activity:**

המסך מורכב מארבעה כפתורים:

1. הרשם- מעביר למסך הרשמות.
2. התחבר- מעביר למסך התחברות.
3. חוקים- מעביר למסך החוקים.
4. התחל- התחל את המשחק. ניתן ללחוץ עליו רק כשמחוברים.

**Rules:**

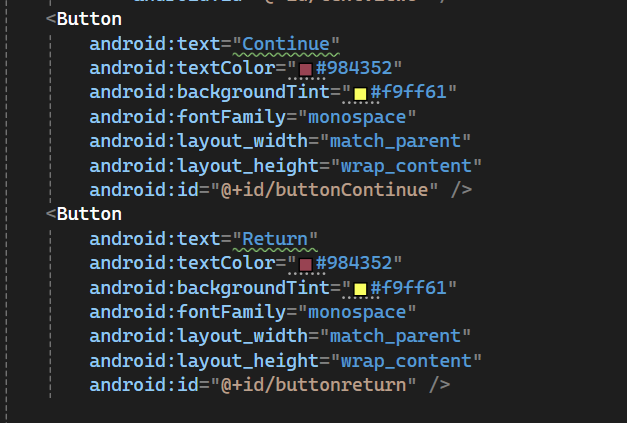
המסך מורכב מתמונה ומכפתור:

התמונה של החוקים.

חזור- מחזיר את המשתמש למסך הקודם.

**Register:**

המסך מורכב משלוש טקסטים שעל המשתמש למלא ומשני כפתורים.

טקסטים:

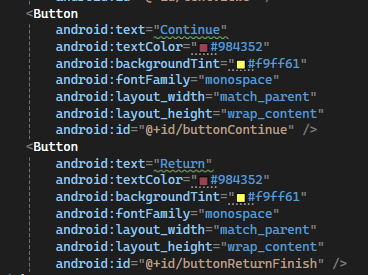
1. שם משתמש.
2. אימייל.
3. סיסמא.

כפתורים:

1. הרשם.
2. חזור.

**Connection:**

המסך מורכב משני טקסטים שעל המשתמש למלא ומשני כפתורים.

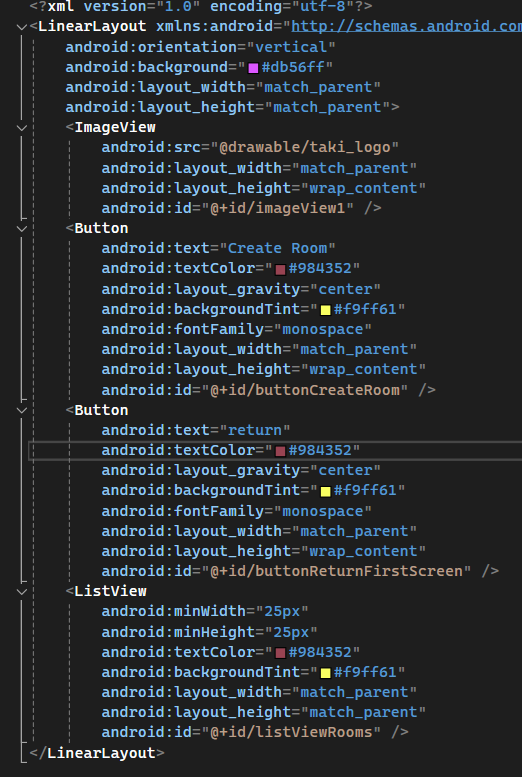
טקסטים:

1. שם משתמש / אימייל.
2. סיסמא.

כפתורים:

1. הירשם.
2. חזור.

**Start:**

המסך מורכב מ- List View שמציג את רשימת החדרים ומשני כפתורים.

כפתורים:

1. צור חדר
2. חזור

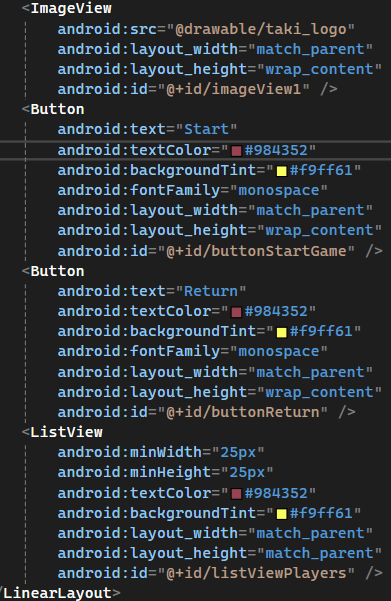
**Create Room:**

המסך מורכב מטקסט שעל המשתמש להקליד, שישמש כשם החדר ומשני כפתורים.

כפתורים:

צור- יוצר חדר.

חזור- חוזר למסך הקודם.

**Wait Room:**

החדר מורכב מכפתור ו- List View שמציג את כל המשתתפים במשחק הנוכחי.

כפתורים:

1. התחל משחק – לחיץ אך ורק לשחקן שיצר את החדר.

**Game:**

המסך מורכב מ- Image Button , מתמונה, ומכפתורים דינמיים, וארבעה כפתורי צבע.

תמונת כפתור:  
הערימה – כשמשתמש לוחץ על הכפתור נוסף לו קלף.

תמונה- כשמשתמש זורק קלף התמונה משתנית.

הכפתורים הדינמיים- הקלפים של המשתמש, בעלי תמונה ולחיצים.

כפתורי צבע- ניתן לראות אותם וללחוץ עליהם רק כשמישהו זרק את הקלף "משנה צבע".

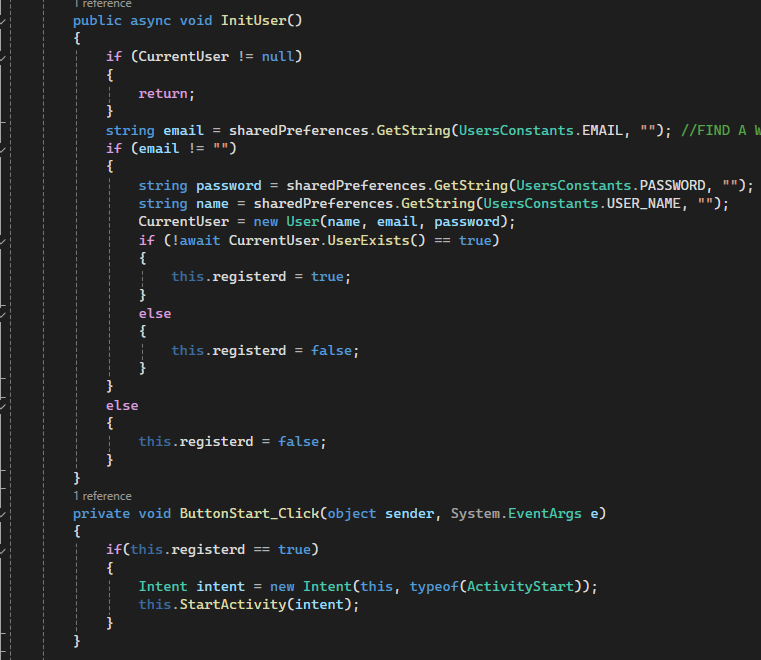
**Win:**

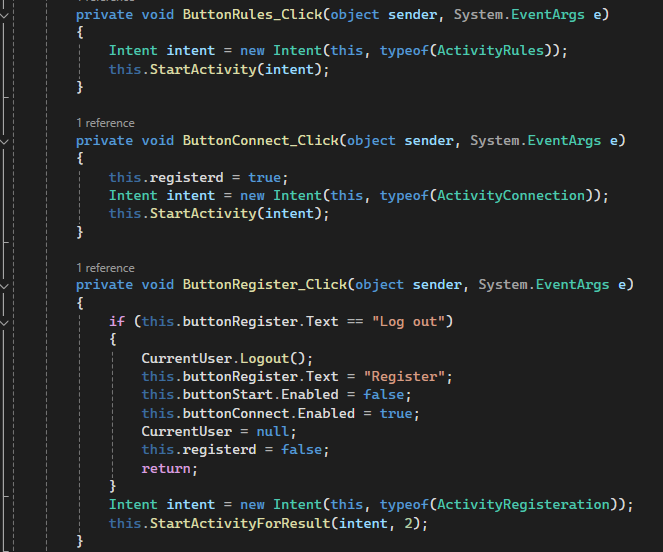
המסך מורכב מטקסט של שם המנצח, מתיבת טקסט, כפתור ותמונה.

המשתמש מכניס קישור לתמונה בתיבת טקסט ולוחץ על הכפתור כדי להוריד אותה. בסוף התמונה מוצגת.

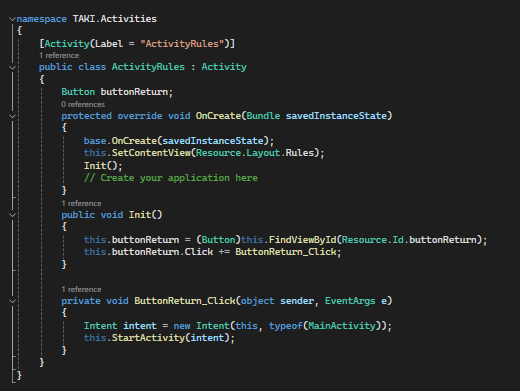
Main Activity:



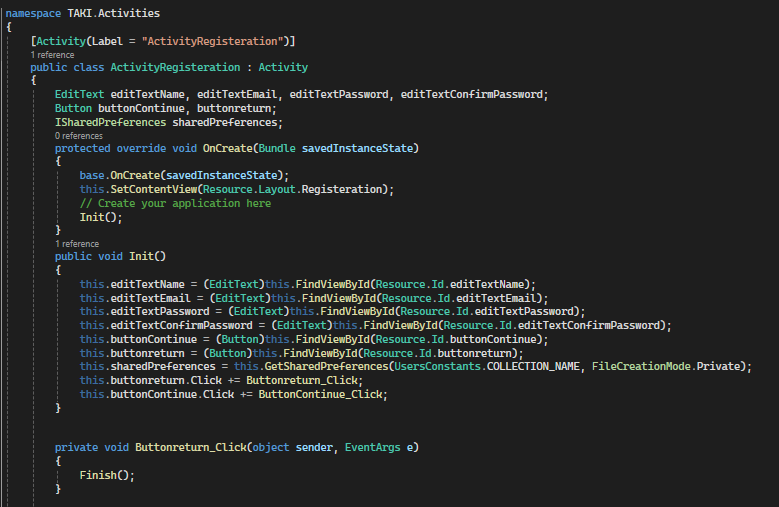
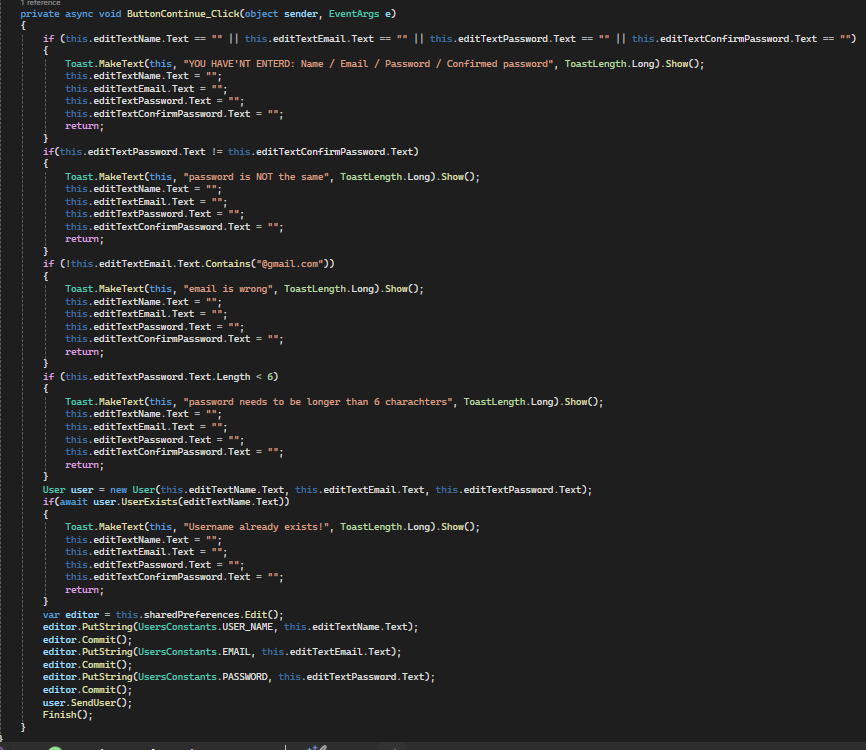


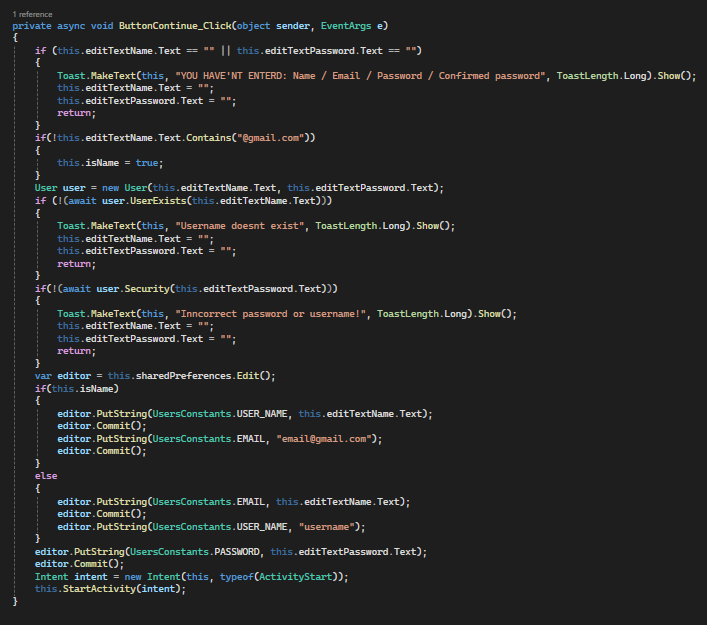
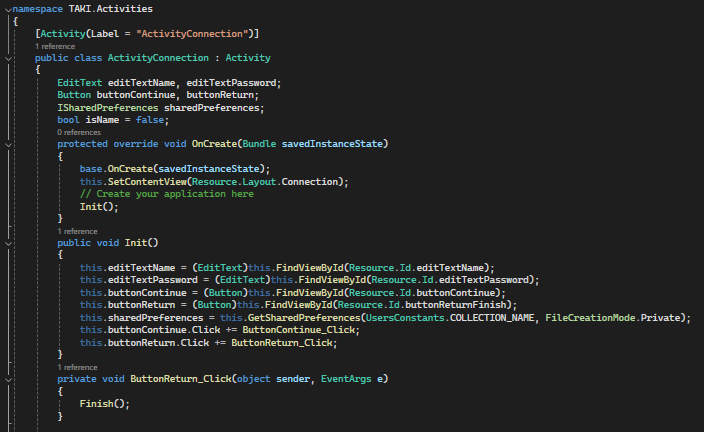


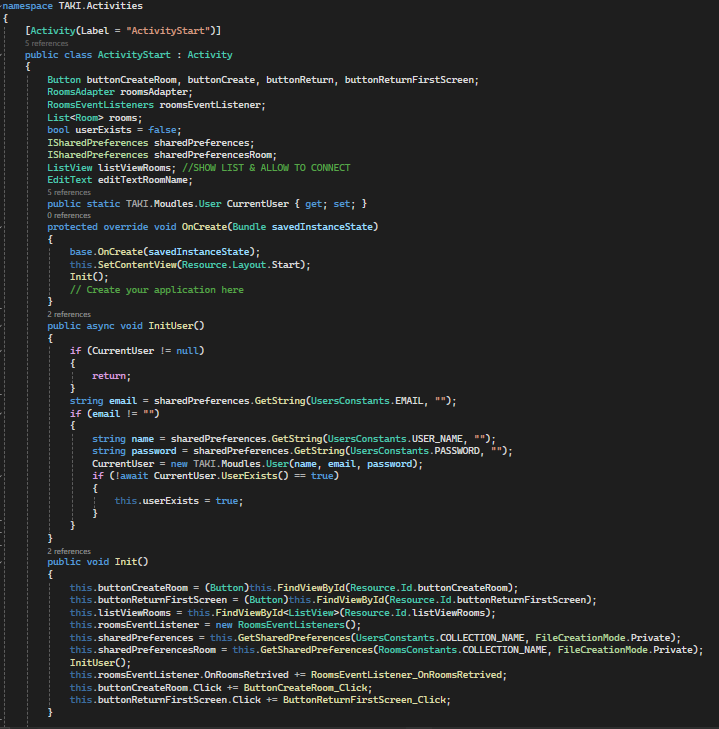
Rules Activity:

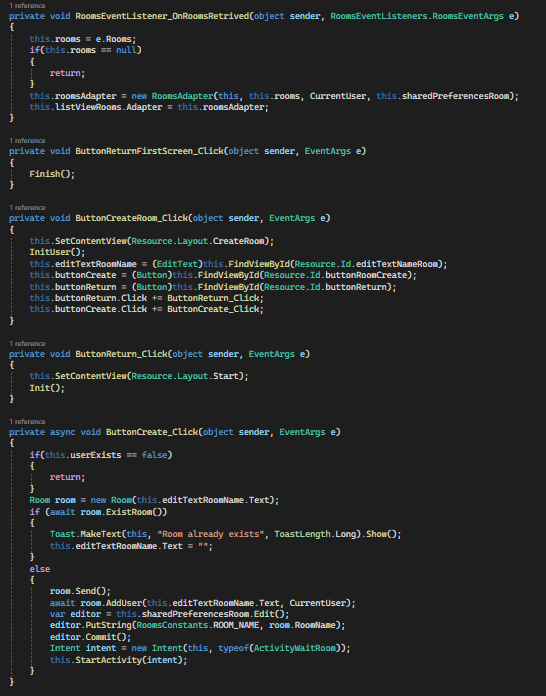


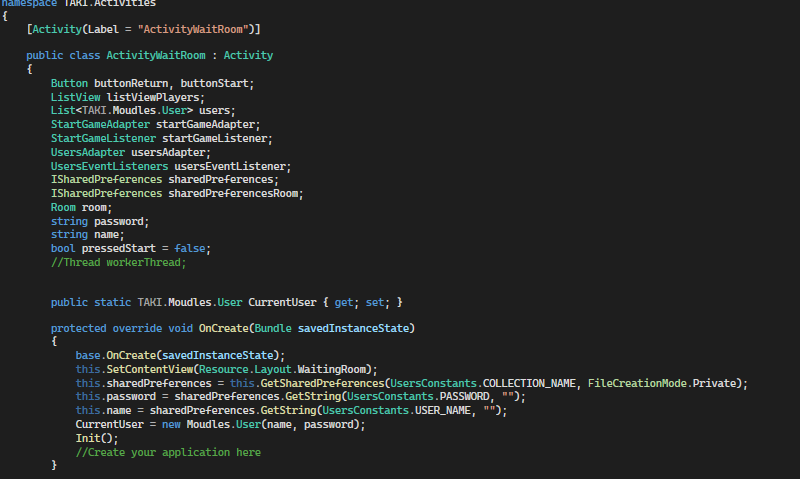
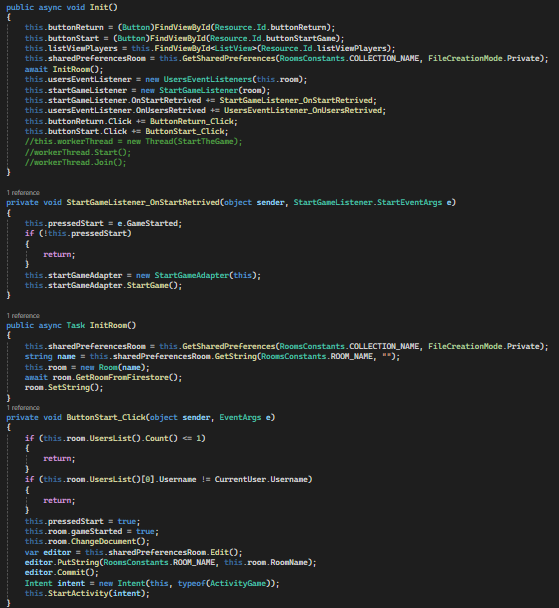
Registration Activity:

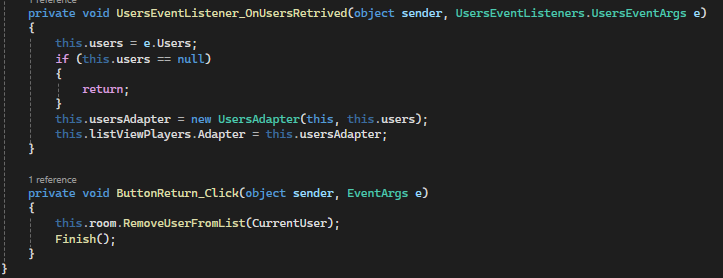


Connection Activity:

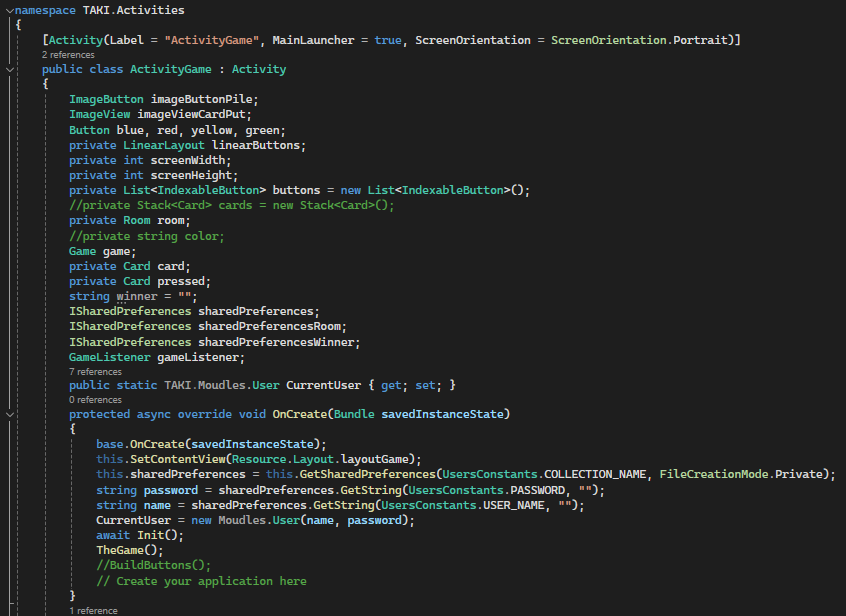
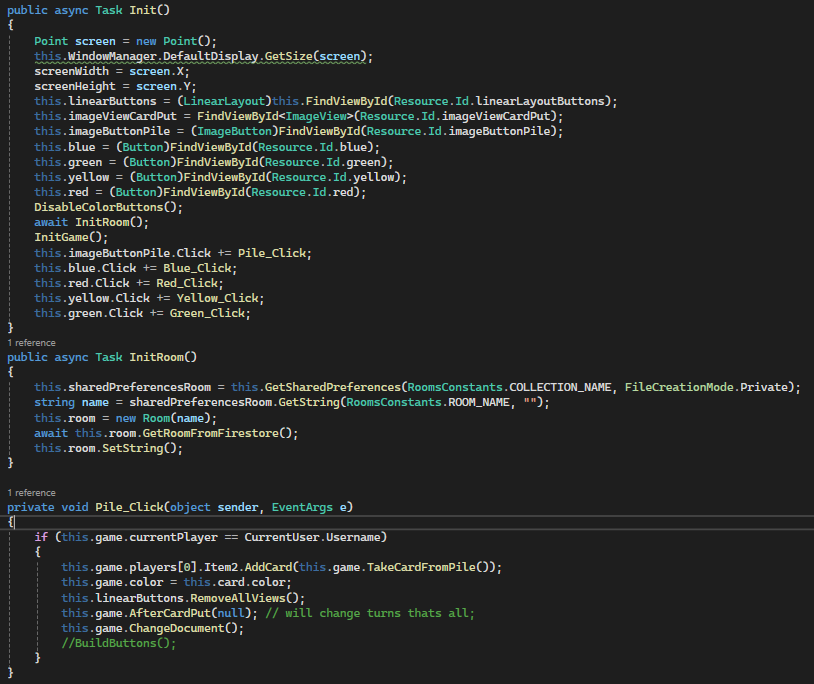
Start Activity:

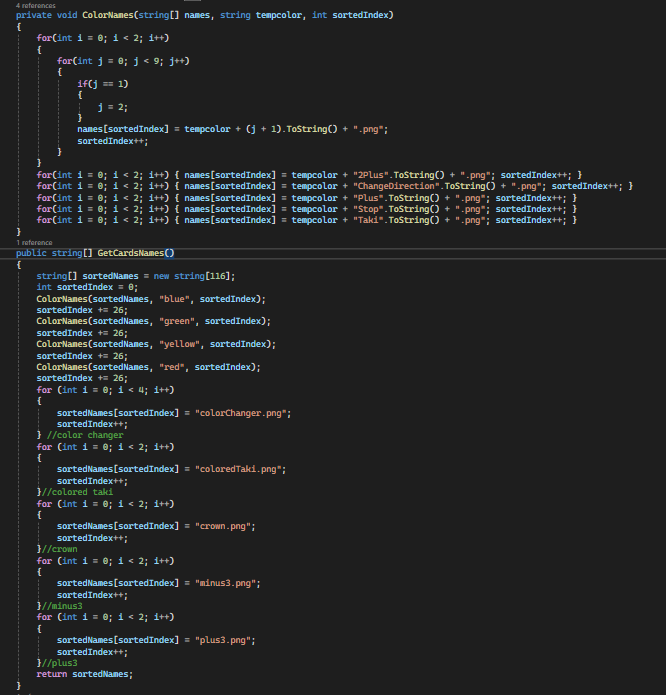
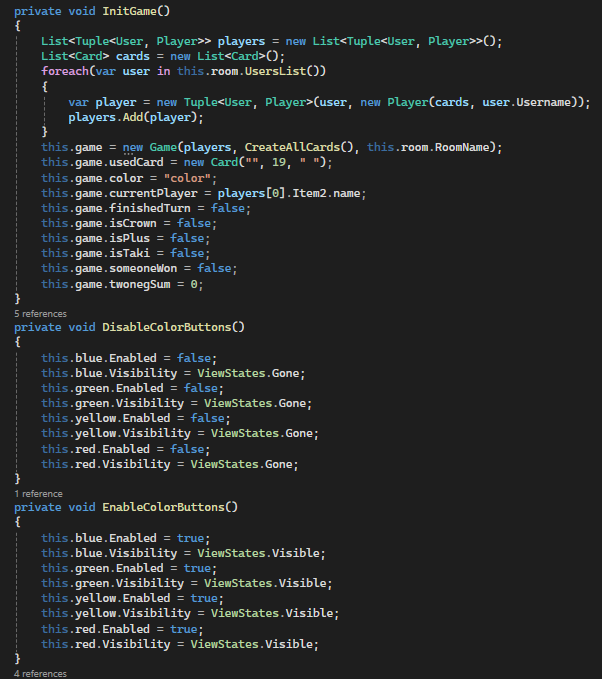


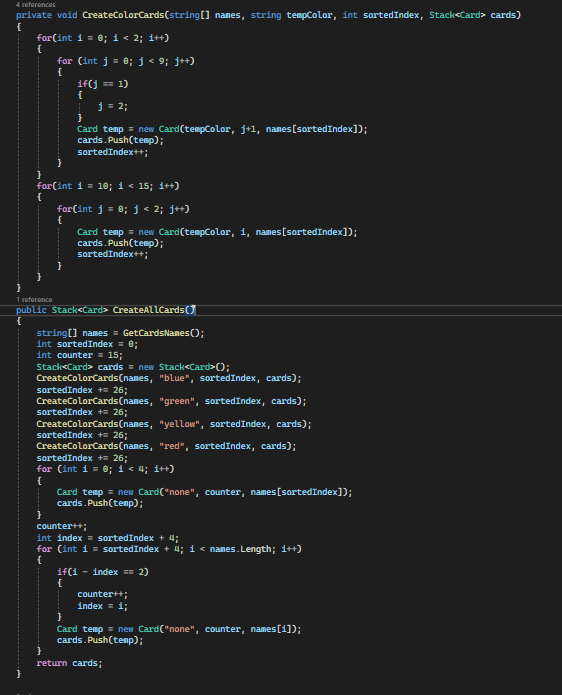
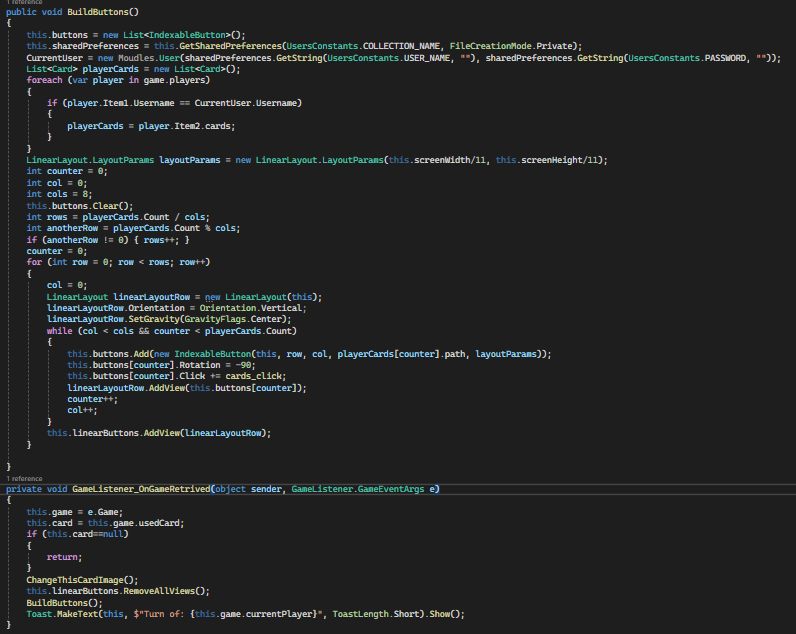
Wait Room Activity:

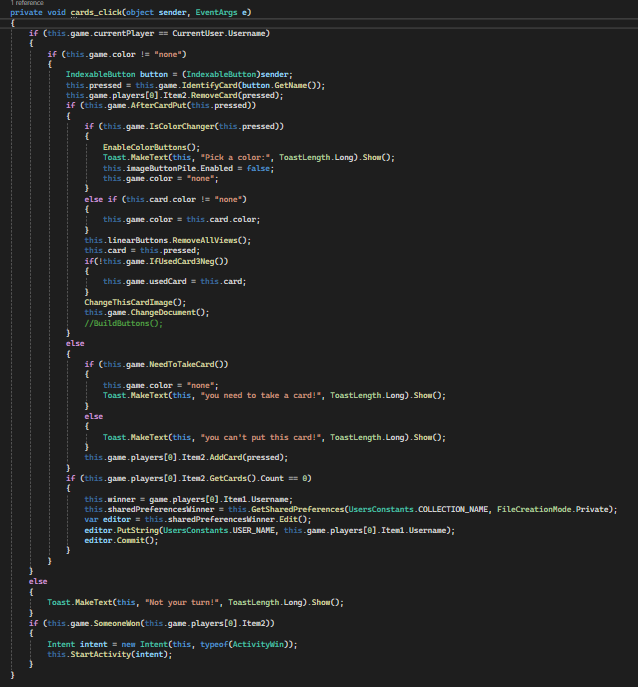
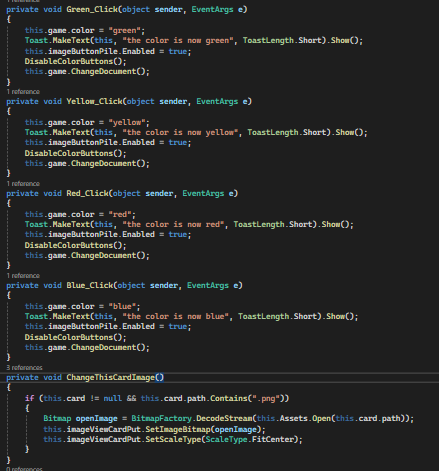


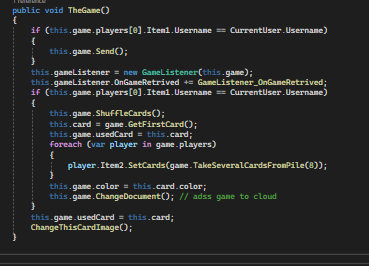
Game Activity:



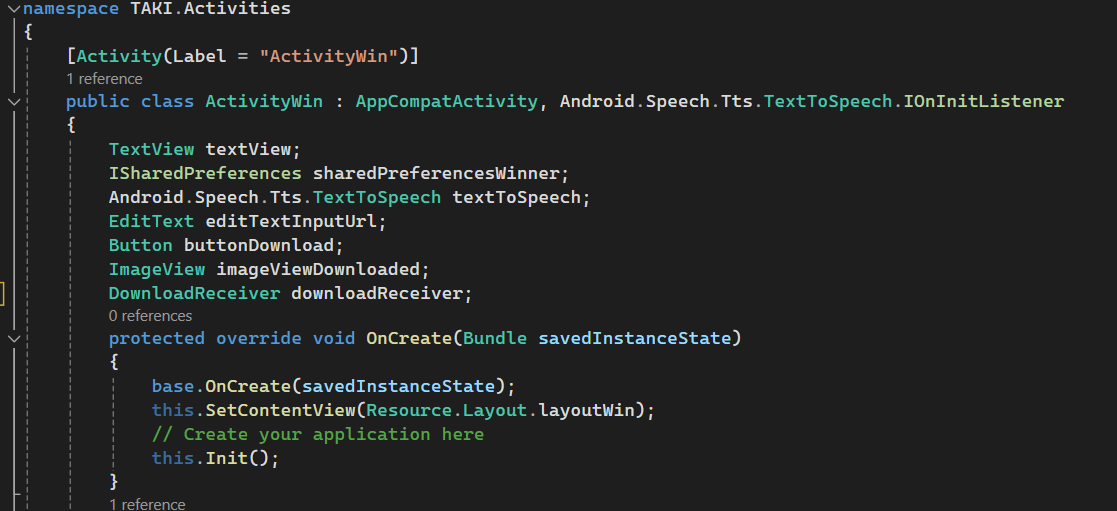


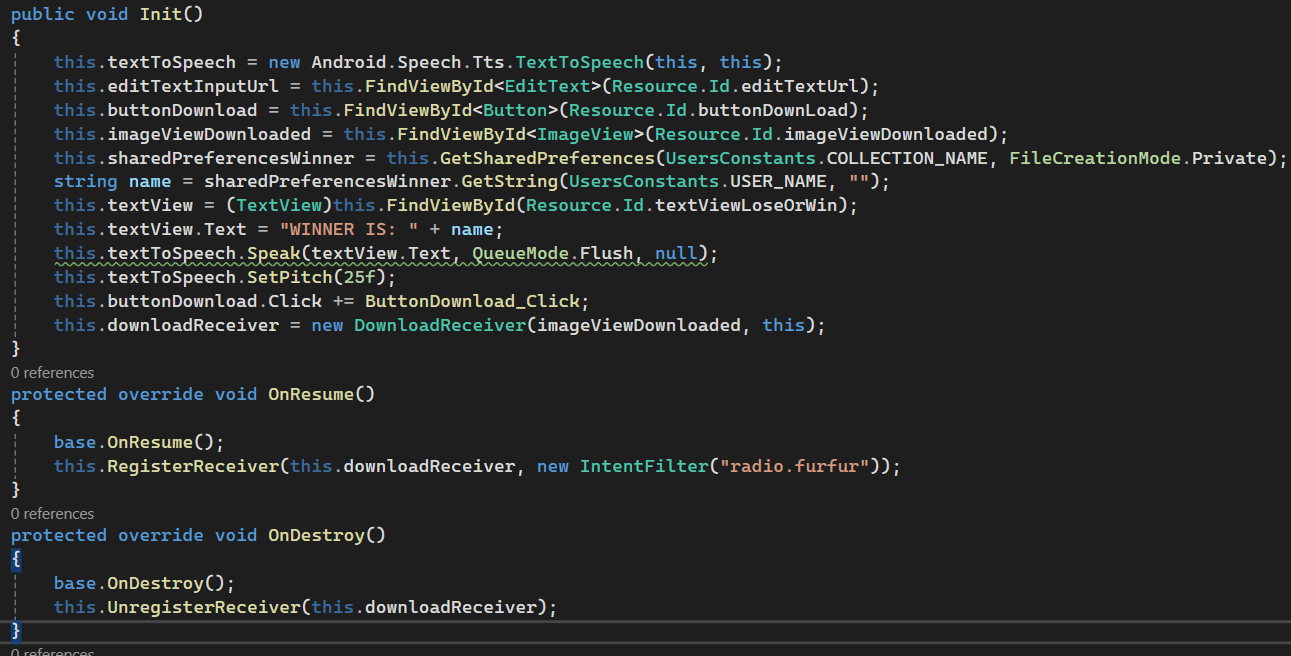
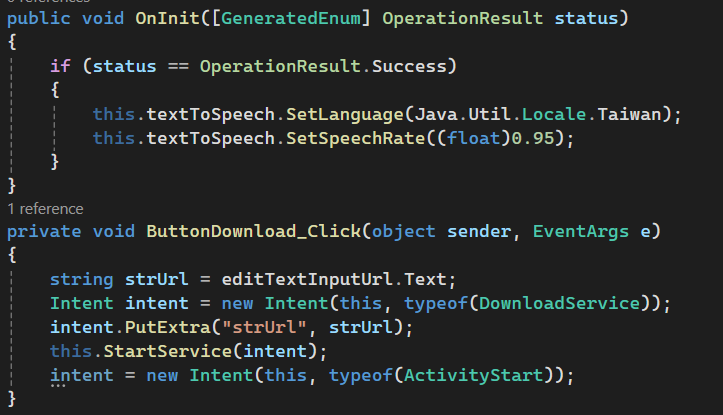






Win Activity:



**תיקיית Activities:**

**Main Activity הסבר:**

Button Register:

מייצג אלמנט כפתור בממשק המשתמש לרישום משתמש. לחיצה על הכפתור מפעילה את תהליך הרישום.

Button Connect:

מייצג אלמנט כפתור בממשק המשתמש להתחברות לפעילות אחרת. לחיצה על הכפתור מתחילה חיבור לפעילות אחרת.

Button Rules:

מייצג אלמנט כפתור בממשק המשתמש לגישה לכללי המשחק. לחיצה על הכפתור פותחת פעילות המציגה את הכללים.

Button Start:

מייצג אלמנט כפתור בממשק המשתמש להתחלת פעילות. לחיצה על הכפתור מתחילה פעילות.

Registered:

משתנה בוליאני המציין האם המשתמש רשום או לא. הוא שולט בהתנהגות של אלמנטים UI מסוימים בהתבסס על מצב ההרשמה.

Shared Preferences:

מופע של ISharedPreferences המשמש לאחסון נתוני המשתמש מקומית במכשיר. הוא מאחסן מידע על המשתמש כגון שם, אימייל וסיסמה.

Current User:

מייצג את המשתמש הנוכחי באפליקציה. זהו מאפיין סטטי של מחלקת המשתמש, המחזיק מידע על המשתמש המחובר.

OnCreate():

מחליף את השיטה שנקראת כאשר הפעילות נוצרת לראשונה. הוא מאתחל את פריטי ה-UI, מסתיר את שורת הפעולות, וקורא לשיטת Init() לאיתור אזכורים ולהקצאתם לתצוגה.

Init():  
מאתחל אלמנטיי ה-UI על ידי חיפוש והקצאת אזכורים לתצוגה המתאימה בפעילות. גם מגדיר מטפלי אירועי לחיצה עבור הכפתורים.

InitUser():

בודק אם המשתמש הנוכחי כבר מחובר על ידי הבאת נתוני המשתמש משתף המחזור. אם משתמש מתגלה, הוא מאתחל את אובייקט ה-CurrentUser.

OnResume():

מחליף את השיטה שנקראת כאשר הפעילות ממשיכה. הוא בודק אם המשתמש הנוכחי הוא null וקורא לשיטת InitUser() כראוי. גם מעדכן אלמנטיי ה-UI בהתאם למצב הרישום.

ButtonStart\_Click():

מטפל באירוע הלחיצה על הכפתור "התחלה". זה מתחיל את הפעילות "ActivityStart" אם המשתמש רשום.

ButtonRules\_Click():

מטפל באירוע הלחיצה על הכפתור "כללי המשחק". מתחיל את הפעילות "ActivityRules" להצגת הכללים.

ButtonConnect\_Click():

מטפל באירוע הלחיצה על הכפתור "התחבר". מגדיר את הדגל registered ל-true ומתחיל את הפעילות "ActivityConnection".

ButtonRegister\_Click():

מטפל באירוע הלחיצה על הכפתור "רישום". אם טקסט הכפתור הוא "התנתקות", הוא מנתק את המשתמש הנוכחי; אחרת, הוא מתחיל את הפעילות "ActivityRegisteration" לרישום משתמש.

**Registration Activity הסבר:**

editTextName:

מקום להזנת שם המשתמש.

editTextEmail:

מקום להזנת כתובת האימייל של המשתמש.

editTextPassword:

מקום להזנת הסיסמה הרצויה.

editTextConfirmPassword:

מקום להזנת אימות הסיסמה.

buttonContinue:

כפתור להמשך הרישום או התהליך הנוכחי.

Buttonreturn:

כפתור לחזרה למסך קודם או ביטול התהליך הנוכחי.

sharedPreferences:

אובייקט המכיל נתונים משותפים בין פעילויות שונות, כגון פרטי משתמש.

OnCreate():

שיטה המקבלת תצורת התצוגה ומאתחלת את האובייקטים והאינסטנסים בפעילות.

Init():

שיטה המאתחלת את כל האלמנטים הנדרשים בממשק המשתמש.

Buttonreturn\_Click():

פעולה שמתרחשת בלחיצה על כפתור החזרה.

ButtonContinue\_Click():

פעולה שמתרחשת בלחיצה על כפתור המשך בתהליך הרישום.

user: אובייקט מסוג User שמייצג את המשתמש שנוצר ושנשלח לשרת.

Editor:

אובייקט המאפשר עריכה ושמירה של נתונים ב-sharedPreferences.

Finish():

שיטה המסיימת את הפעילות הנוכחית ומחזירה את השליטה לפעילות הקודמת.

**Connection Activity הסבר:**

editTextName:

מקום להזנת שם המשתמש או האימייל שלו.

editTextPassword:

מקום להזנת הסיסמה של המשתמש.

buttonContinue:

כפתור הממשיך לפעולה הבאה בתהליך.

buttonReturn:

כפתור המאפשר חזרה או סיום הפעולה הנוכחית.

sharedPreferences:

אובייקט המכיל נתונים משותפים בין פעילויות שונות, כגון פרטי משתמש.

isName:

משתנה בוליאני שמציין אם שם המשתמש נכתב ככתובת אימייל או כשם משתמש.

OnCreate():

שיטה המקבלת תצורת התצוגה ומאתחלת את האובייקטים והאינסטנסים בפעילות.

Init():

שיטה המאתחלת את כל האלמנטים הנדרשים בממשק המשתמש.

ButtonReturn\_Click():

פעולה שמתרחשת בלחיצה על כפתור החזרה או סיום הפעולה הנוכחית.

ButtonContinue\_Click():

פעולה שמתרחשת בלחיצה על כפתור המשך בתהליך.

Rules Activity הסבר:

ButtonReturn:

כפתור המאפשר חזרה או חזרה למסך הראשי של האפליקציה.

OnCreate():   
שיטה המקבלת תצורת התצוגה ומאתחלת את האובייקטים והאינסטנסים בפעילות.

Init():

שיטה המאתחלת את כל האלמנטים הנדרשים בממשק המשתמש.

ButtonReturn\_Click():

פעולה שמתרחשת בלחיצה על כפתור החזרה, המנהלת את המעבר למסך הראשי של האפליקציה.

**Activity Start הסבר:**

ButtonCreateRoom:

כפתור המאפשר למשתמש ליצור חדר חדש.

buttonReturnFirstScreen:

כפתור המאפשר למשתמש לחזור למסך הראשי של האפליקציה.

listViewRooms:

רשימת החדרים הזמינים להצטרפות.

roomsAdapter:

מתאם המאפשר להציג את רשימת החדרים בתצוגה.

roomsEventListener:

מאזין המטפל באירועים הקשורים לחדרים.

Rooms:

רשימת החדרים שהושגה מהשרת.

userExists:

משתנה בוליאני המציין האם המשתמש קיים כבר במערכת או לא.

sharedPreferences:

מאפשר גישה לנתונים שמורים בזיכרון פנימי של המכשיר של המשתמש.

sharedPreferencesRoom:

מאפשר גישה לנתונים שמורים בזיכרון פנימי של המכשיר של המשתמש, קשורים לחדרים.

editTextRoomName:

תיבת טקסט להזנת שם החדר שנוצר.

CurrentUser:

אובייקט מסוג User המייצג את המשתמש הנוכחי.

**Activity Wait Room הסבר:**

Button Return:

כפתור המאפשר למשתמש לחזור למסך הקודם או לסגור את החדר הנוכחי.

Button Start:

כפתור המאפשר למשתמש להתחיל את המשחק כאשר הכל מוכן.

listViewPlayers:

רשימת השחקנים הנוכחיים בחדר הממתינים להתחלת המשחק.

Users:

רשימת השחקנים הנוכחים בחדר.

startGameAdapter:

מתאם המאפשר להציג את תחילת המשחק כאשר מתחילים.

startGameListener:

מאזין המטפל באירועים הקשורים להתחלת המשחק.

usersAdapter:

מתאם המאפשר להציג את רשימת השחקנים בתצוגה.

usersEventListener:

מאזין המטפל באירועים הקשורים לשחקנים בחדר.

sharedPreferences:

מאפשר גישה לנתונים שמורים בזיכרון פנימי של המכשיר של המשתמש.

sharedPreferencesRoom:

מאפשר גישה לנתונים שמורים בזיכרון פנימי של המכשיר של המשתמש, קשורים לחדרים.

Room:

אובייקט מסוג Room המייצג את החדר הנוכחי.

Password:

מחרוזת המכילה את הסיסמה של המשתמש הנוכחי.

Name:

מחרוזת המכילה את שם המשתמש הנוכחי.

pressedStart:

משתנה בוליאני המציין האם המשתמש לחץ על כפתור "התחל" והמשחק יכול להתחיל.

CurrentUser:

אובייקט מסוג User המייצג את המשתמש הנוכחי.

**Activity Game הסבר:**

ImageButtonPile:

כפתור לערימת הקלפים.

ImageViewCardPut:

תמונה להצגת הקלף שהונח.

Button blue, red, yellow, green:

כפתורים לבחירת צבע הקלף.

LinearButtons:

תבנית המכילה כפתורים דינמיים.

ממדי מסך:

screenWidth, screenHeight:

לאחסון ממדי המסך.

נתוני משחק:

buttons:

רשימה של כפתורים מותאמים אישית המייצגים קלפים.

room:

מופע המייצג את חדר המשחק הנוכחי.

game:

מופע המייצג את מצב המשחק הנוכחי.

card, pressed:

מופעים של קלפים, אחד שמשוחק כעת ואחד שנלחץ על ידי המשתמש.

winner:

לאחסון שם המנצח.

sharedPreferences, sharedPreferencesRoom, sharedPreferencesWinner:

לאחסון ושליפת נתוני משתמש ומשחק.

CurrentUser:

המשתמש הנוכחי של המשחק.

GameListener:

להאזנה לעדכוני משחק.

OnCreate(Bundle savedInstanceState):

מאתחלת את הפעילות ומגדירה את תצוגת התוכן. שולפת נתוני משתמש ומפעילה שיטות אתחול.

Init():

מאתחלת רכיבים שונים כמו ממדי מסך, רכיבי ממשק משתמש ומאזיני אירועים.

InitRoom():

מאתחלת את חדר המשחק על ידי שליפת נתוני חדר מהעדפות משותפות ו-Firestore.

InitGame():

מגדירה את המשחק על ידי יצירת שחקנים, קלפים ומצב משחק ראשוני.

שיטות ממשק משתמש

DisableColorButtons():

מבטלת את כפתורי בחירת הצבע.

EnableColorButtons():

מאפשרת את כפתורי בחירת הצבע.

ChangeThisCardImage():

משנה את תמונת הקלף שהונח כעת.

מטפלי אירועים

Pile\_Click(object sender, EventArgs e):

מטפל באירוע כאשר נלחץ כפתור הערימה.

Green\_Click(object sender, EventArgs e):

מטפל באירוע כאשר נלחץ כפתור הצבע הירוק.

Yellow\_Click(object sender, EventArgs e):

מטפל באירוע כאשר נלחץ כפתור הצבע הצהוב.

Red\_Click(object sender, EventArgs e):

מטפל באירוע כאשר נלחץ כפתור הצבע האדום.

Blue\_Click(object sender, EventArgs e):

מטפל באירוע כאשר נלחץ כפתור הצבע הכחול.

cards\_click(object sender, EventArgs e):

מטפל באירוע כאשר נלחץ כפתור קלף.

TheGame():

מתחילה את המשחק, מגדירה את המאזין למשחק, מערבבת קלפים ומחלקת קלפים ראשוניים.

GameListener\_OnGameRetrived(object sender, GameListener.GameEventArgs e):

מטפל בעדכוני מצב משחק שהתקבלו מהמאזין.

GetCardsNames():

מייצרת מערך של שמות קובצי תמונות קלפים.

CreateAllCards():

יוצרת מחסנית של כל הקלפים עם מאפיינים מתאימים.

CreateColorCards(string[] names, string tempColor, int sortedIndex, Stack<Card> cards):

פונקציית עזר ליצירת קלפים צבעוניים.

**Activity Win הסבר:**

TextView textView: אובייקט TextView המייצג תיבת טקסט להצגת המנצח במשחק.

ISharedPreferences sharedPreferencesWinner:

אובייקט ISharedPreferences המאפשר קריאה וכתיבה לקובץ SharedPreferences, המשמש לאחסון פרטים על המנצח במשחק.

Android.Speech.Tts.TextToSpeech textToSpeech:

אובייקט TextToSpeech המאפשר הפקת דיבור בקול סינתטי, כך שהאפליקציה תוכל להגיד את שם המנצח.

EditText editTextInputUrl:

אובייקט EditText המאפשר למשתמש להזין כתובת אינטרנט להורדת התמונה.

Button buttonDownload:

אובייקט Button המאפשר למשתמש לאתחל את תהליך הורדת התמונה.

ImageView imageViewDownloaded:

אובייקט ImageView המייצג את התמונה שהורדה מהרשת.

DownloadReceiver downloadReceiver:

אובייקט מקבל הודעות Broadcast מסוג DownloadReceiver, שמתרחש בעת סיום הורדת התמונה מהשירות.

OnInit([GeneratedEnum] OperationResult status):

משתמשת בפונקציה המוגדרת מראש כדי לאתחל את מנוע הדיבור כאשר הוא מוכן לשימוש.

OnCreate(Bundle savedInstanceState):

מפעילה את פעילות ה־ActivityWin, מאתחלת את הרכיבים ומציגה את תוצאת המנצח במשחק.

Init():

אובייקטי הרכיבים מאתחלים את עצמם ומגדירים את האירועים הנדרשים בפעילות. מתבצעים הגדרות כגון הקראת הטקסט באמצעות תמונת הדיבור והפעלת מאזין להודעות השידור הכללי.

OnResume():

מפעילה את המאזין להודעות Broadcast של DownloadReceiver כאשר הפעילות מופעלת.

OnDestroy():

מפסיקה את המאזין להודעות Broadcast של DownloadReceiver כאשר הפעילות מסתיימת.

ButtonDownload\_Click(object sender, EventArgs e):

פעולה שנקראת כאשר המשתמש לוחץ על כפתור הורדה. הפעולה מקבלת את הכתובת שהוזנה על ידי המשתמש ומפעילה שירות ייחודי להורדת התמונה מהרשת.

**תיקיית Adapters:**

Rooms Adapter הסבר:

האדאפטר אחראי על ניהול והצגת רשימת חדרים (Rooms) באפליקציה.

משתנים פרטיים-

Context:

ההקשר שבו פועלת הפעילות הנוכחית.

Rooms:

רשימה של חדרים המוצגת ברשימת החדרים.

User:

המשתמש הנוכחי.

SharedPreferences:

העדפות משותפות לאחסון נתונים בצורה מתמשכת.

הקונסטרקטור של המחלקה מקבל את ההקשר, רשימת החדרים, המשתמש וההעדפות המשותפות, ומאתחל את המשתנים הפרטיים.

GetItem:

מחזירה את האובייקט במיקום המסוים. במקרה הזה, מחזירה את המיקום עצמו.

GetItemId:

מחזירה את המזהה הייחודי של האובייקט במיקום המסוים. במקרה הזה, מחזירה את המיקום עצמו.

GetView:

אחראית על יצירת התצוגה עבור כל פריט ברשימה. שיטה זו מייצרת את התצוגה של החדר בהתאם לעיצוב המוגדר.

ButtonJoin\_Click:

מטפלת באירוע הלחיצה על כפתור ההצטרפות לחדר. השיטה מאתחלת את תהליך ההצטרפות לחדר, ומתחילה פעילות חדשה במידת הצורך.

מאפיינים-

Count:

מחזירה את מספר החדרים ברשימה.

this[int position]:

מחזירה את החדר במיקום המסוים ברשימה.

**User Adapter הסבר:**

משתנים פרטיים-

Context:

ההקשר שבו פועלת הפעילות הנוכחית.

Users:

רשימה של משתמשים המוצגת ברשימה.

הקונסטרקטור מקבל את ההקשר ואת רשימת המשתמשים ומאתחל את המשתנים הפרטיים.

GetItem:

מחזירה את האובייקט במיקום המסוים. במקרה הזה, מחזירה את המיקום עצמו.

GetItemId:

מחזירה את המזהה הייחודי של האובייקט במיקום המסוים. במקרה הזה, מחזירה את המיקום עצמו.

GetView:

אחראית על יצירת התצוגה עבור כל פריט ברשימה. שיטה זו מייצרת את התצוגה של המשתמש בהתאם לעיצוב המוגדר.

מאפיינים-

Count:

מחזירה את מספר המשתמשים ברשימה.

this[int position]:

מחזירה את המשתמש במיקום המסוים ברשימה.

Start Game Adapter הסבר:

מחלקה זו אינה אדאפטר קלאסי שמנהל תצוגות ברשימה, אלא אחראית להתחיל פעילותActivity) ) חדשה.

משתנים פרטיים-

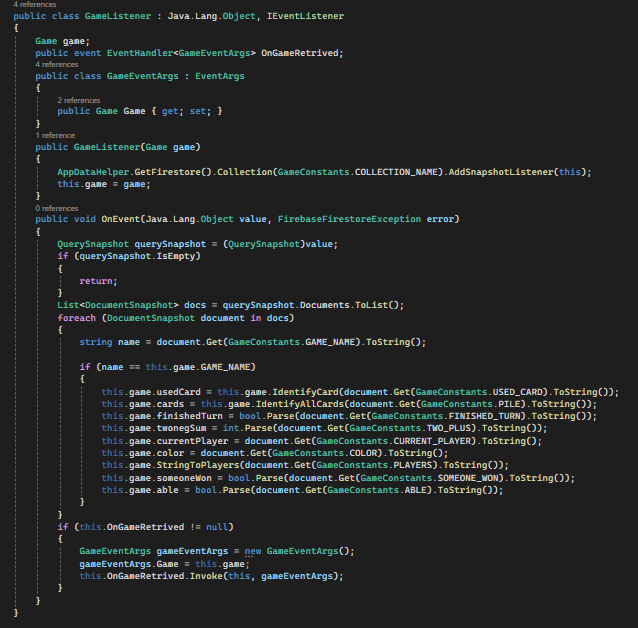
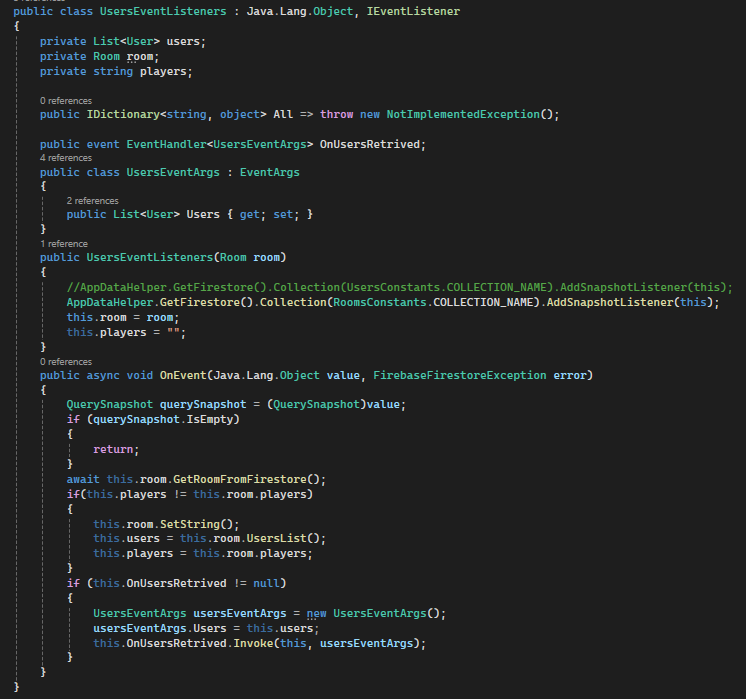
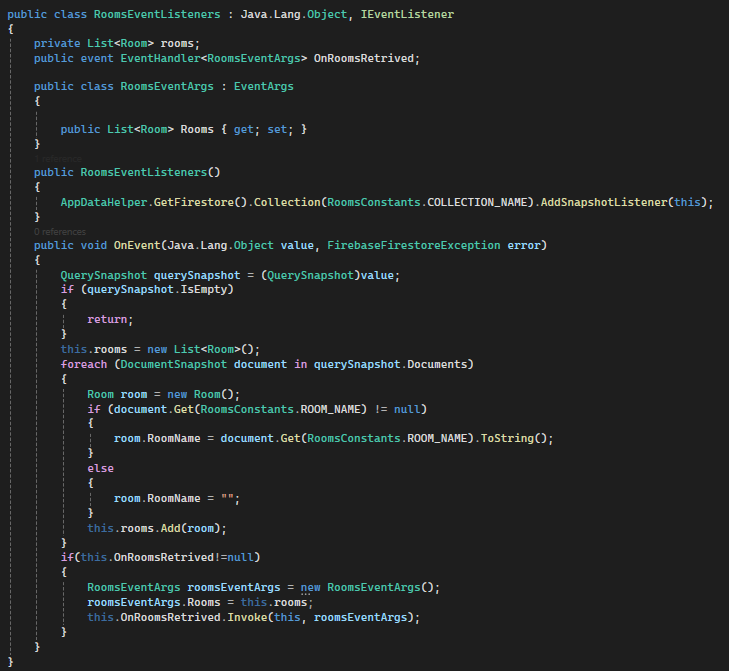
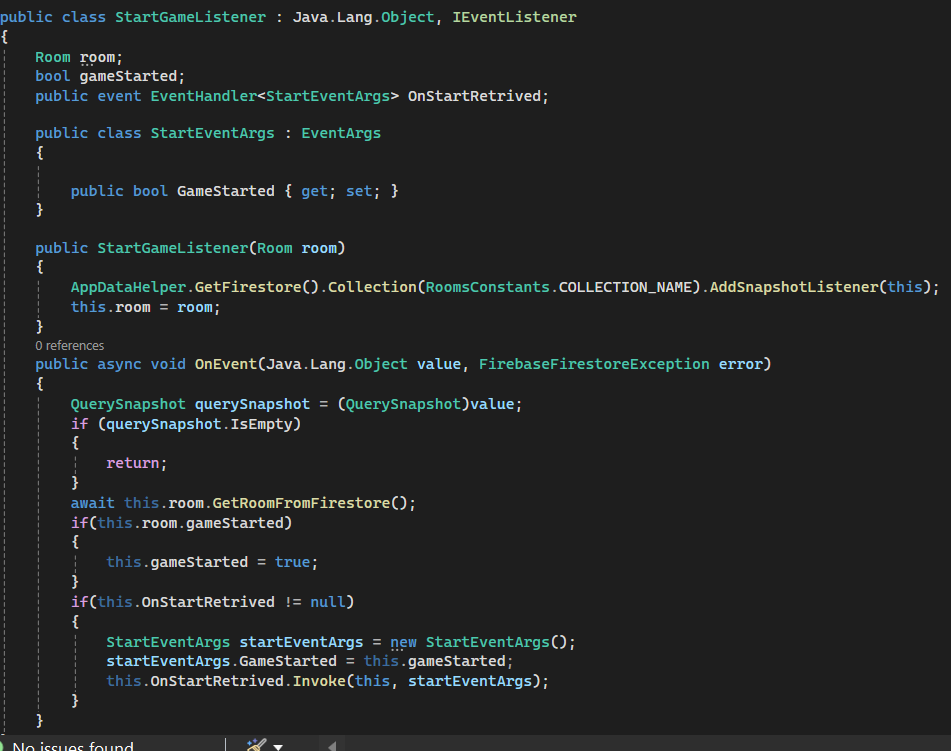
Context:

ההקשר שבו פועלת הפעילות הנוכחית. ההקשר משמש להפעלת פעילויות חדשות או גישה למשאבים של האפליקציה.

הקונסטרקטור מקבל את ההקשר ומאתחל את המשתנה הפרטי המתאים.

StartGame:

שיטה זו יוצרת Intent חדשה שמתחילה את הפעילות ActivityGame. הפעילות הנוכחית היא ActivityWaitRoom.

**תיקיית Listeners:**

Rooms Event Listener הסבר:

המחלקה RoomsEventListeners משמשת להאזנה לאירועים הקשורים לשינויים במאגר הנתונים של חדרים באמצעות Firebase Firestore. המחלקה מממשת את הממשק IEventListener של Firebase Firestore ומנהלת רשימה של חדרים כאשר מתרחשים שינויים בנתונים, המחלקה מעבדת את השינויים ומפעילה אירוע מותאם אישית כדי להודיע עליהם.

משתנים פרטיים-

Rooms:

רשימה של אובייקטי Room שמכילה את כל החדרים המתקבלים מהמאגר.

OnRoomsRetrived:

אירוע שמופעל כאשר נתוני החדרים מתקבלים מהמאגר. האירוע משתמש באירוע מותאם אישית (RoomsEventArgs) שמעביר את רשימת החדרים.

RoomsEventArgs:

מחלקה פנימית שמרחיבה את EventArgs. משמשת להעברת רשימת החדרים באירוע OnRoomsRetrived.

הקונסטרקטור מאתחל את המאזין לשינויים באוסף הנתונים של החדרים במאגר Firestore באמצעות קריאה ל- AddSnapshotListener.

OnEvent:

שיטה שממומשת מהממשק IEventListener. השיטה מופעלת כאשר יש שינוי במאגר הנתונים של .Firestore

Value:

האובייקט שמכיל את הנתונים שהתקבלו מהמאגר.

Error:

אובייקט של FirebaseFirestoreException שמכיל את פרטי השגיאה אם ישנה.

השיטה בודקת אם יש מסמכים באוסף הנתונים שהתקבל. אם כן, היא יוצרת רשימת חדרים חדשה וממלאת אותה בנתונים מהמסמכים. לאחר מכן, אם האירוע OnRoomsRetrived אינו ריק, היא מפעילה אותו ומעבירה את רשימת החדרים באמצעות האובייקט RoomsEventArgs.

**User Event Listener הסבר:**

המחלקה UsersEventListeners משמשת להאזנה לאירועים הקשורים לשינויים במאגר הנתונים של משתמשים (או שחקנים) באמצעות Firebase Firestore. המחלקה מממשת את הממשק IEventListener של Firebase Firestore ומנהלת רשימה של משתמשים (User) עבור חדר נתוןRoom) .) כאשר מתרחשים שינויים בנתונים, המחלקה מעבדת את השינויים ומפעילה אירוע מותאם אישית כדי להודיע עליהם.

Users:

רשימה של אובייקטי User שמכילה את כל המשתמשים הקשורים לחדר.

Room:

אובייקט Room שמייצג את החדר שאליו מתייחסים המשתמשים.

Players:

מחרוזת שמייצגת את רשימת השחקנים הנוכחית של החדר.

אירועים-

OnUsersRetrived:

אירוע שמופעל כאשר נתוני המשתמשים מתקבלים מהמאגר. האירוע משתמש באירוע מותאם אישית (UsersEventArgs) שמעביר את רשימת המשתמשים.

מחלקות פנימיות-

UsersEventArgs:

מחלקה פנימית שמרחיבה את EventArgs. משמשת להעברת רשימת המשתמשים באירוע OnUsersRetrived.

הקונסטרקטור מקבל אובייקט Room ומאזין לשינויים באוסף הנתונים של החדרים במאגר Firestore באמצעות קריאה ל- AddSnapshotListener. הוא מאתחל את המשתנה room למשתנה שהתקבל ומאפס את המשתנה players למחרוזת ריקה.

OnEvent:

שיטה שממומשת מהממשק IEventListener. השיטה מופעלת כאשר יש שינוי במאגר הנתונים של Firestore. השיטה מקבלת שני פרמטרים-

Value:

האובייקט שמכיל את הנתונים שהתקבלו מהמאגר.

error:

אובייקט של FirebaseFirestoreException שמכיל את פרטי השגיאה אם ישנה.

השיטה בודקת אם המסמכים שהתקבלו ריקים. אם לא, היא מאחזרת את החדר המעודכן מהמאגר ובודקת אם רשימת השחקנים השתנתה. אם כן, היא מעדכנת את רשימת המשתמשים ואת המשתנה players. לאחר מכן, אם האירוע OnUsersRetrived אינו ריק, היא מפעילה אותו ומעבירה את רשימת המשתמשים באמצעות האובייקט UsersEventArgs.

**Start Game Listener הסבר:**

המחלקה StartGameListener משמשת להאזנה לאירועים הקשורים לתחילת משחק בחדר מסוים באמצעות Firebase Firestore. המחלקה מממשת את הממשק IEventListener של Firebase Firestore ומנהלת אובייקט Room כדי לבדוק האם המשחק התחיל. כאשר מתרחשים שינויים בנתונים, המחלקה מעבדת את השינויים ומפעילה אירוע מותאם אישית כדי להודיע עליהם.

משתנים פרטיים-

Room:

אובייקט Room שמייצג את החדר שבו נבדק האם המשחק התחיל.

gameStarted:

משתנה בוליאני שמסמן האם המשחק התחיל.

אירועים-

OnStartRetrived:

אירוע שמופעל כאשר מתקבלים נתונים שמסמנים האם המשחק התחיל. האירוע משתמש באירוע מותאם אישית שמעביר את המצב של תחילת המשחק.

מחלקות פנימיות-

StartEventArgs:

מחלקה פנימית שמרחיבה את EventArg משמשת להעברת המצב של תחילת המשחק באירוע OnStartRetrived.

הקונסטרקטור מקבל אובייקט Room ומאזין לשינויים באוסף הנתונים של החדרים במאגר Firestore באמצעות קריאה ל- AddSnapshotListener. הוא מאתחל את המשתנה room למשתנה שהתקבל.

OnEvent:

שיטה שממומשת מהממשק IEventListener. השיטה מופעלת כאשר יש שינוי במאגר הנתונים של Firestore. השיטה מקבלת שני פרמטרים:

Value:

האובייקט שמכיל את הנתונים שהתקבלו מהמאגר.

Error:

אובייקט של FirebaseFirestoreException שמכיל את פרטי השגיאה אם ישנה.

השיטה בודקת אם המסמכים שהתקבלו ריקים. אם לא, היא מאחזרת את החדר המעודכן מהמאגר ובודקת אם המשחק התחיל. אם המשחק התחיל, היא מעדכנת את המשתנה gameStarted לערך true. לאחר מכן, אם האירוע OnStartRetrived אינו ריק, היא מפעילה אותו ומעבירה את המצב של תחילת המשחק באמצעות האובייקט StartEventArgs.

Game Listener הסבר:

מחלקת GameListener משמשת כמאזין לשינויים במשחק באפליקציה, באמצעות Firebase Firestore. המחלקה מממשת את הממשק IEventListener ומאזינה לשינויים בנתוני המשחק השמורים במסד הנתונים. כאשר מתבצעים שינויים, המחלקה מפעילה אירוע מותאם אישית על מנת להעביר את הנתונים המעודכנים למחלקות אחרות באפליקציה.

משתנים פרטיים-

Game:

אובייקט משחק המייצג את המשחק שבו נבדקים השינויים.

אירועים-

OnGameRetrived:

אירוע מותאם אישית המפעיל אירוע כאשר מתקבלים נתונים עם שינויים במשחק. האירוע משתמש במחלקת אירוע מותאמת אישית להעברת המשחק המעודכן.

מחלקות פנימיות-

GameEventArgs:

מחלקה פנימית שמרחיבה את EventArgs. משמשת להעברת נתוני המשחק באירוע OnGameRetrived.

הקונסטרקטור מקבל אובייקט משחק (Game) ומאזין לשינויים בנתוני המשחק במסד הנתונים באמצעות קריאה ל- AddSnapshotListener. הוא מאתחל את המשתנה game לאובייקט שהתקבל.

שיטות עיקריות

OnEvent:

שיטה שממומשת מהממשק IEventListener. השיטה מופעלת כאשר יש שינוי במאגר הנתונים של Firestore שקשור למשחק. השיטה מקבלת שני פרמטרים:

Value:

האובייקט שמכיל את הנתונים שהתקבלו מהמאגר.

Error:

אובייקט של FirebaseFirestoreException שמכיל את פרטי השגיאה אם ישנה.

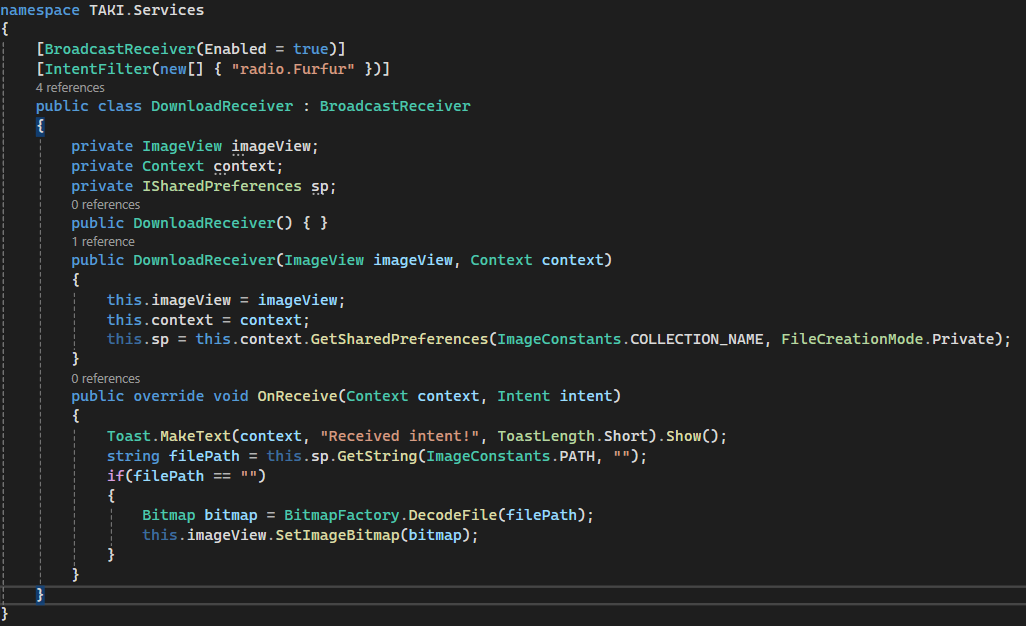
השיטה מבצעת את הפעולות הבאות:

בודקת אם המסמך המתקבל אינו ריק.

עבור כל מסמך ברשימת המסמכים, מתרגמת את הנתונים המצויים בו לנתוני המשחק המתאימים.

אם המסמך מכיל מידע על המשחק הנוכחי, היא מעדכנת את נתוני המשחק באובייקט game.

בודקת אם האירוע OnGameRetrived אינו ריק ומפעילה אותו, מעבירה את נתוני המשחק המעודכנים באמצעות אובייקט GameEventArgs.



**תיקיית Services:**

**DownloadReceiver הסבר:**

המחלקה מוגדרת כ-BroadcastReceiver ומיועדת להתעדכן בעת מקבלת הודעה מסוימת.

המחלקה מכילה משתנים פרטיים כגון imageView, context ו־sp (SharedPreferences) המיועדים לאחסון מידע על התמונה המורדת, וכן לתצוגתה. ניתן להעביר את ה־ImageView וה־Context בזמן יצירת האובייקט של המחלקה דרך הבנאי השני.

פעולת OnReceive מופעלת כאשר מתקבלת הודעה מסוג "radio.Furfur". בתוך הפעולה, הופקדה משימת הצגת הודעת חלון (Toast) המציגה הודעה קצרה למשתמש.

לאחר מכן, הפעולה בודקת אם קיים נתיב לתמונה בקבצי ה־SharedPreferences. אם קיים, המחלקה מפעילה את BitmapFactory כדי להמיר את התמונה מקובץ לסוג Bitmap , ומציבה את התמונה בתוך ה- ImageView המועברת ביצירת האובייקט.

בסך הכל, מחלקת DownloadReceiver מתפקדת כגורם המתפקד ברקע, מקבל הודעות בצורה חיצונית, ומטפלת בהצגת התמונה באפליקציה כאשר היא מורדת בהצלחה מהרשת.

**DownloadService הסבר:**

מכיל משתנה פרטי strUrl המיועד לאחסון כתובת האינטרנט של הקובץ שיש להוריד, וכן משתנה sp שמיועד לאחסון מידע על התמונה המורדת ב־SharedPreferences.

המחלקה מממשת את שיטות OnCreate, OnStartCommand ו־ OnDestroy המהוות חלק ממחזור חיי השירות באנדרואיד. בתוך שיטת OnStartCommand, השירות מקבל את כתובת האינטרנט של הקובץ להורדה דרך ה־Intent המועבר לו ומפעיל תהליך הורדה ברקע באמצעות תהליך נפרד.

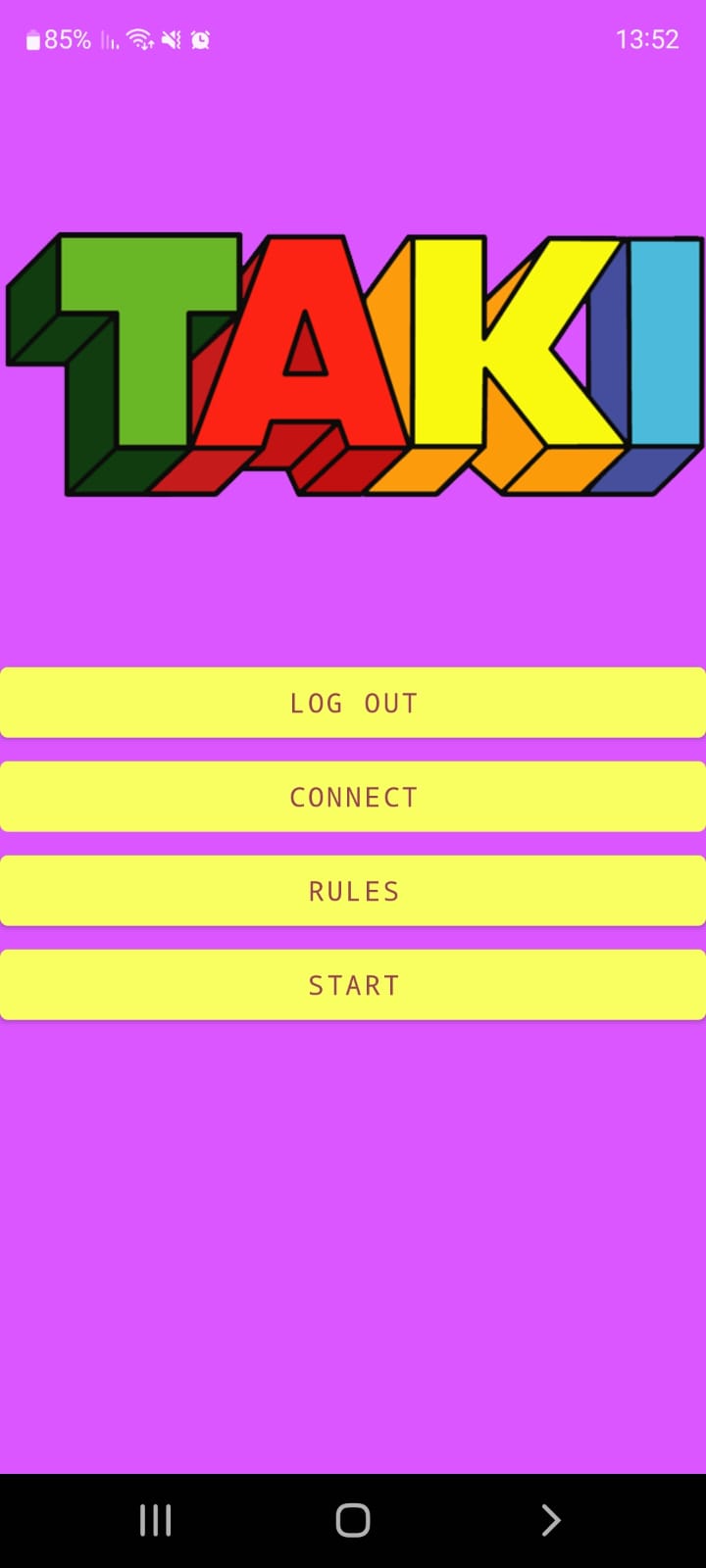
במתודה Download, השירות משתמש במחלקת WebClient כדי להוריד את הנתונים מהאינטרנט כתוב בכתובת שמקבל. הנתונים מורדים כתוך מערך של .Bytes לאחר הורדת הנתונים, הם מורכבים לתמונה בפורמט png ומורכבים לקובץ בכתובת מקומית על ידי פתיחת FileStream וכתיבת הנתונים לתוך הקובץ.

לאחר הורדת הקובץ ושמירתו בצורה מקומית, השירות מעדכן את ערך ה־SharedPreferences עם המיקום המקומי של התמונה, ולאחר מכן מפעיל הודעת Broadcast כדי להודיע לרכיבים אחרים שהורדת התמונה הושלמה.

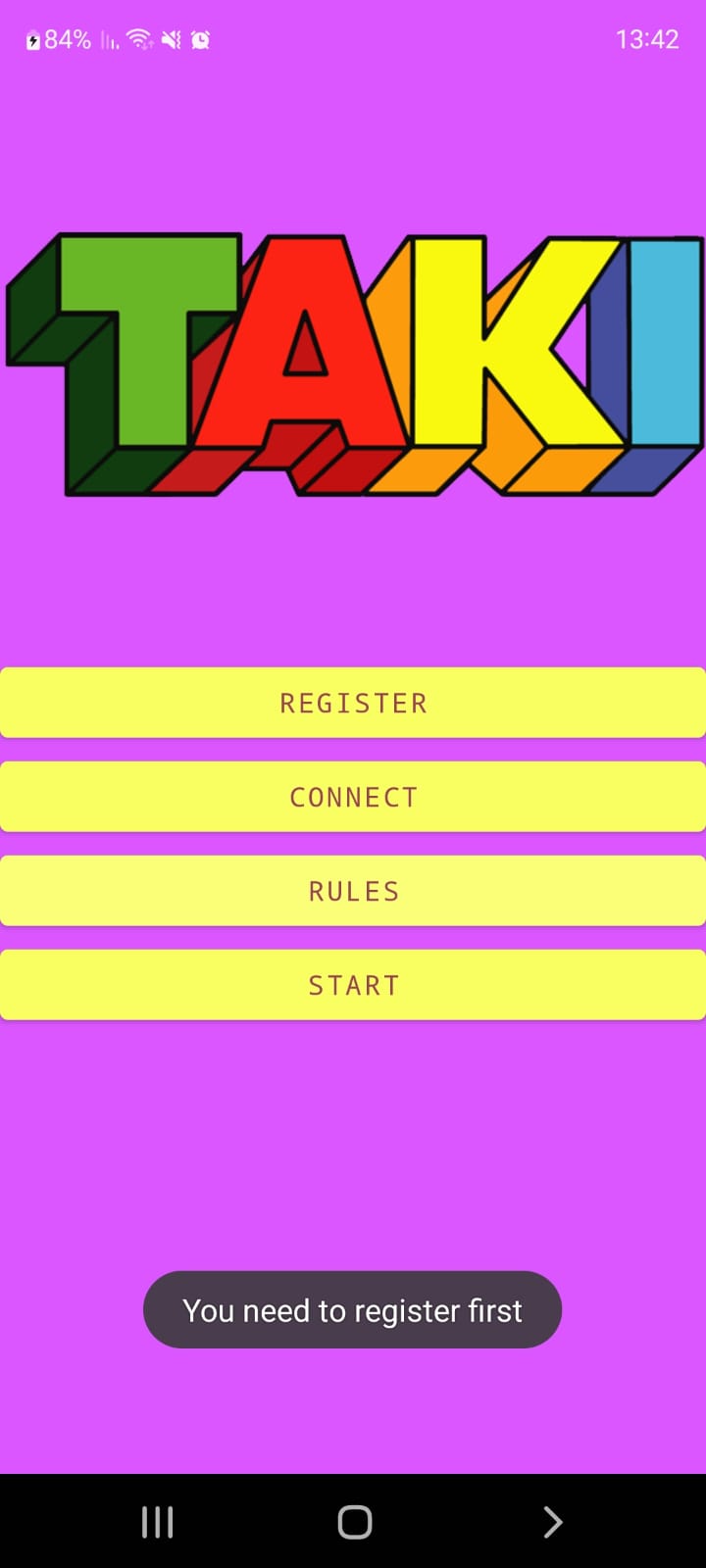
בסופו של דבר, השירות נכבה על ידי קריאה לשיטת StopSelf() המפסיקה את פעולתו.

**ממשק למשתמש:**

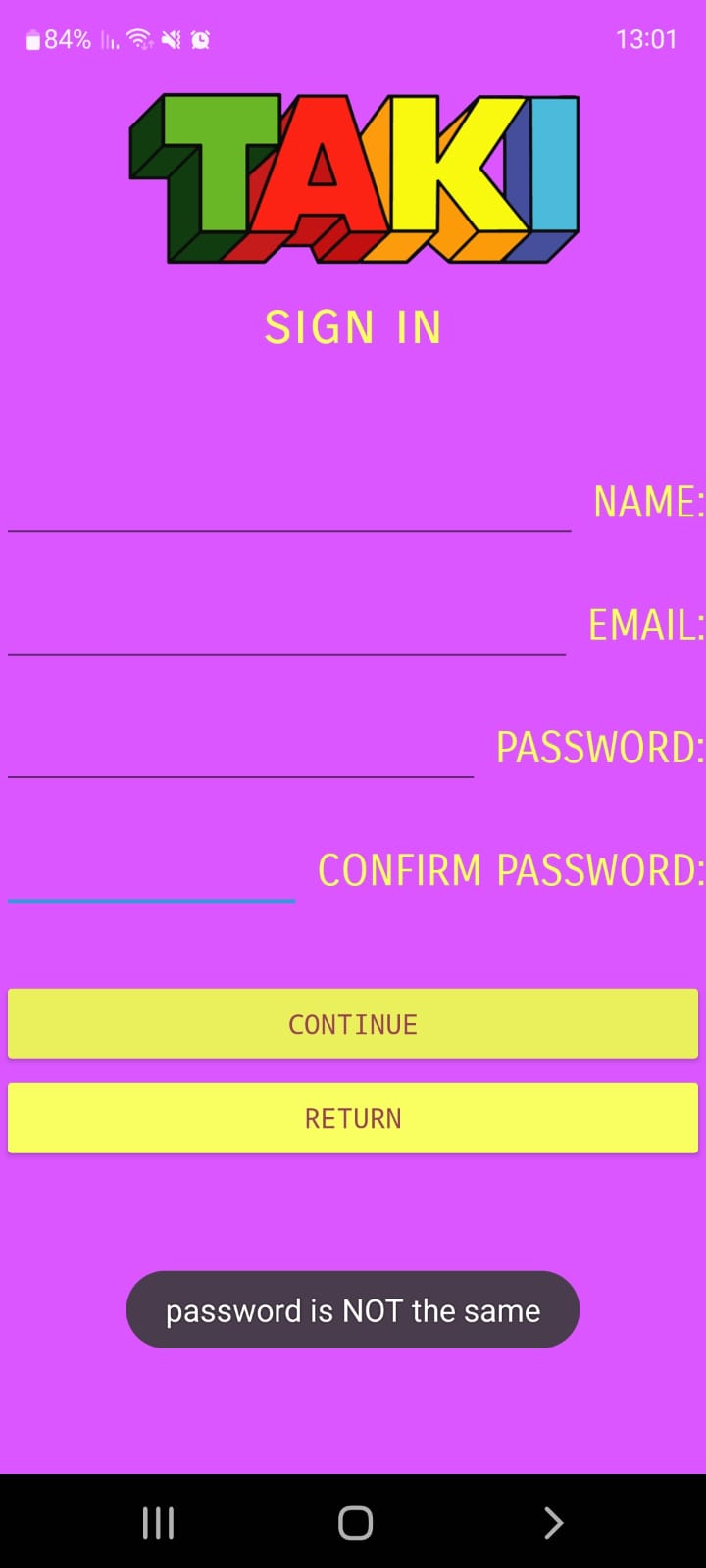
מסך ראשון: ארבעה כפתורים שמובילים למסכים אחרים.

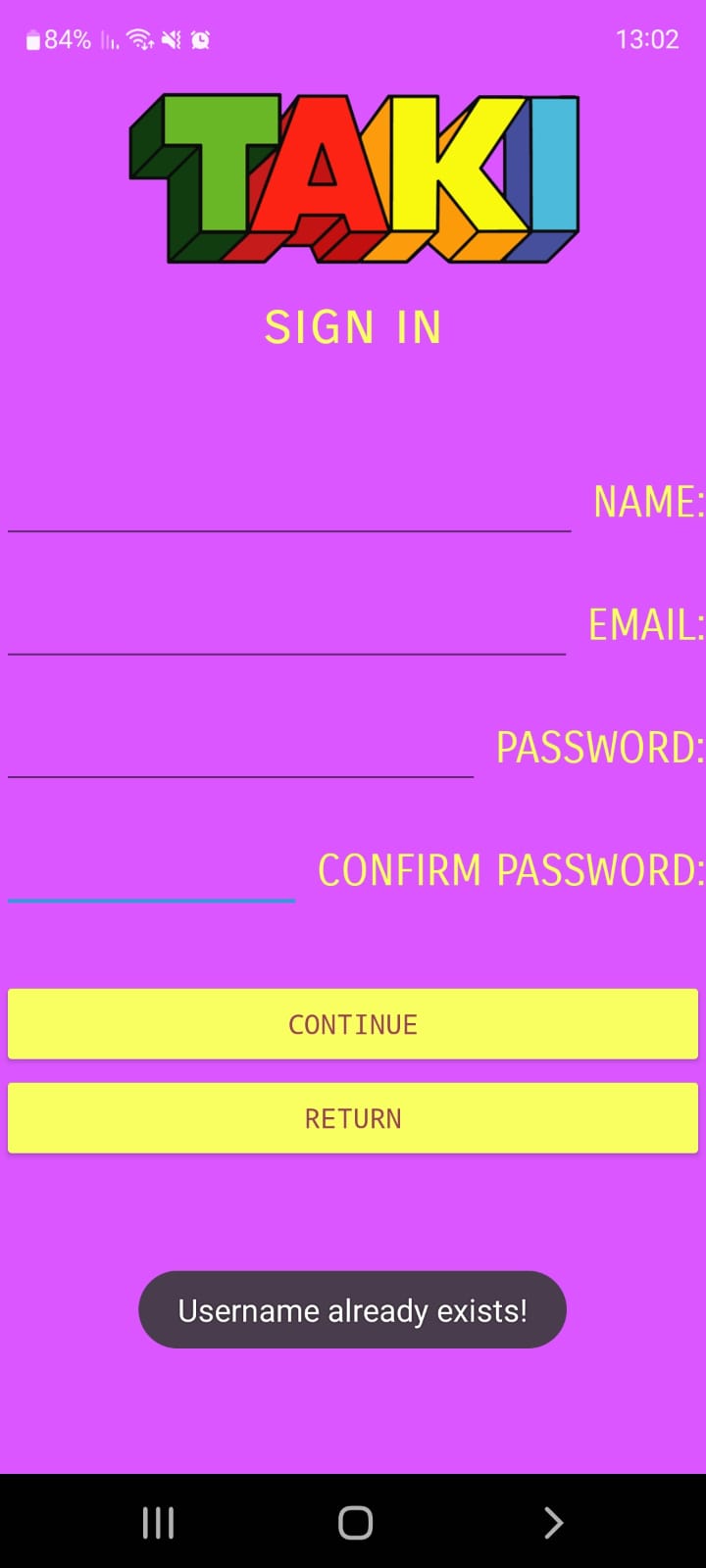
אם המשתמש נרשם – הכפתור העליון הופך ל- LOG OUT.

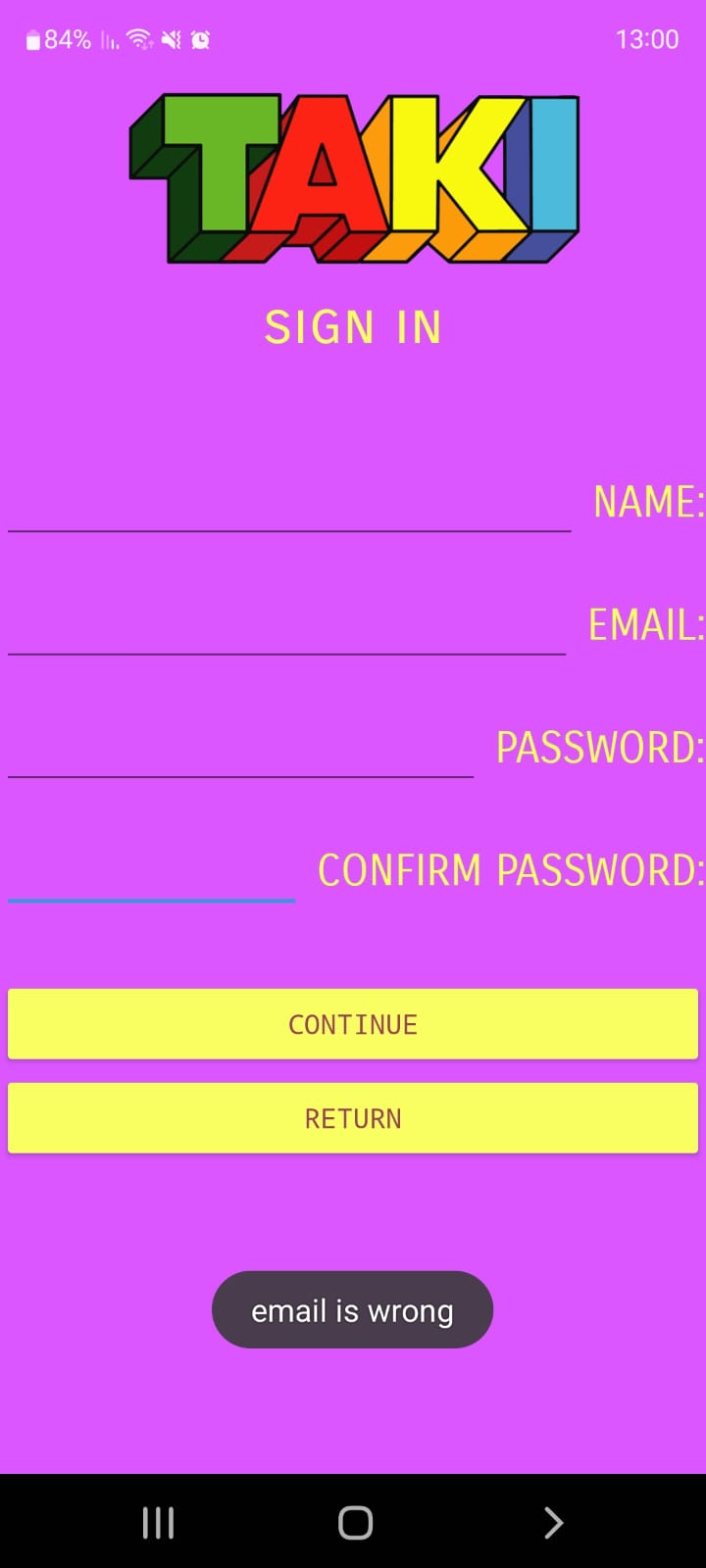
הכפתור START לא לחיץ אלא אם המשתמש רשום – אם אינו רשום מופיעה התראה:

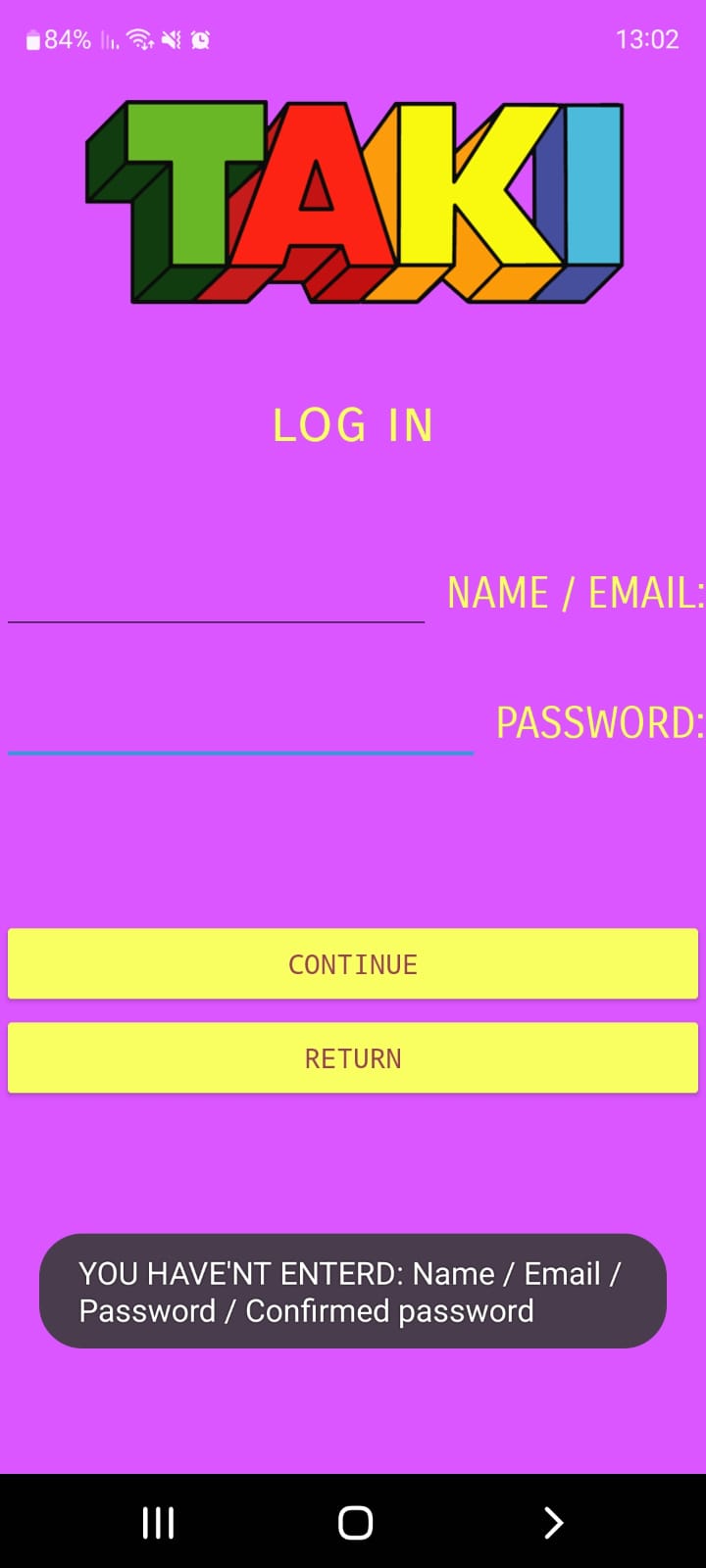


מסך הרשמה: ארבע תיבות טקסט, המשתמש צריך להקליד- אם אחת מתיבות הטקסט ריקה מופיעה התראה.

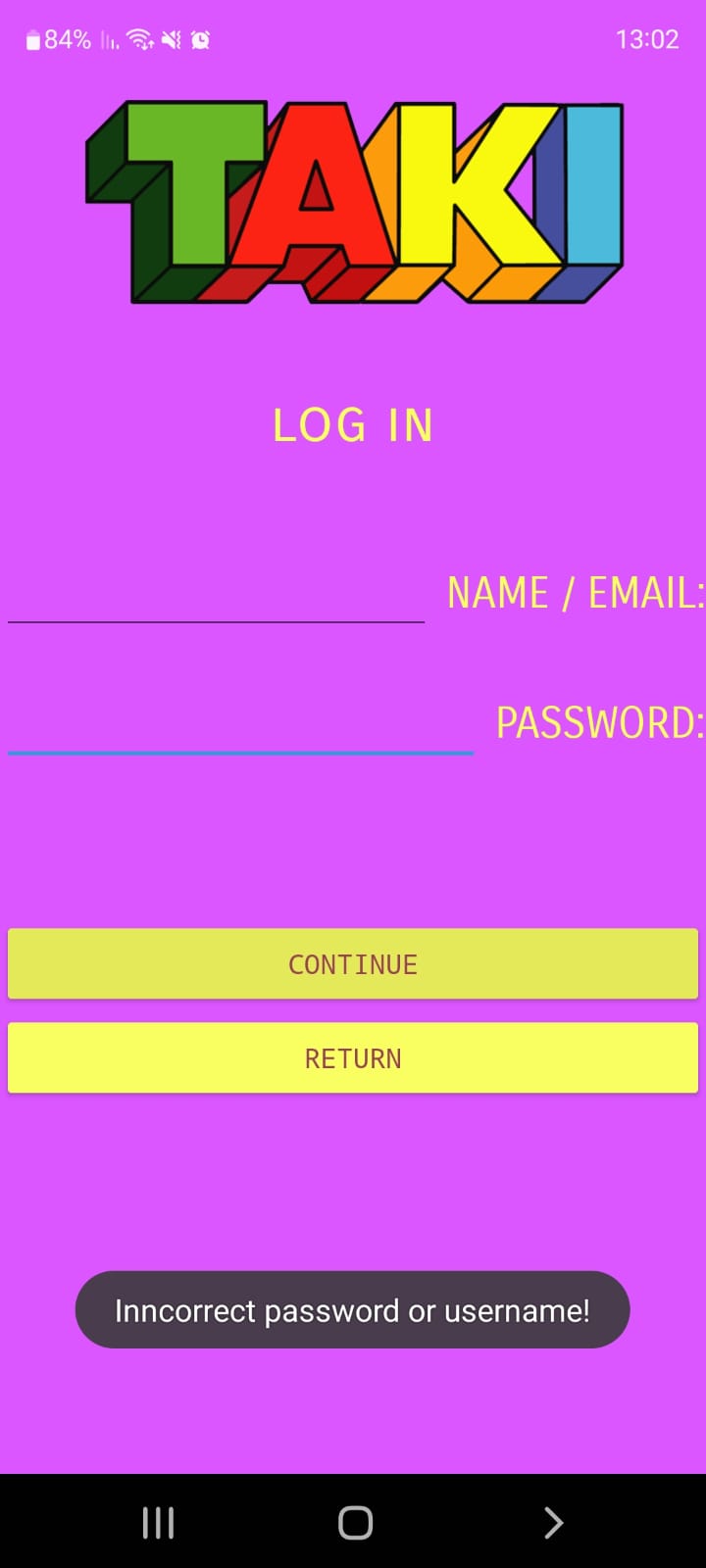
אם הסיסמא ואימות הסיסמא לא זהים מופיעה התראה.

אם משתמש עם שם זהה כבר קיים מופיעה התראה.

אם האימייל לא תקין מופיעה התראה.

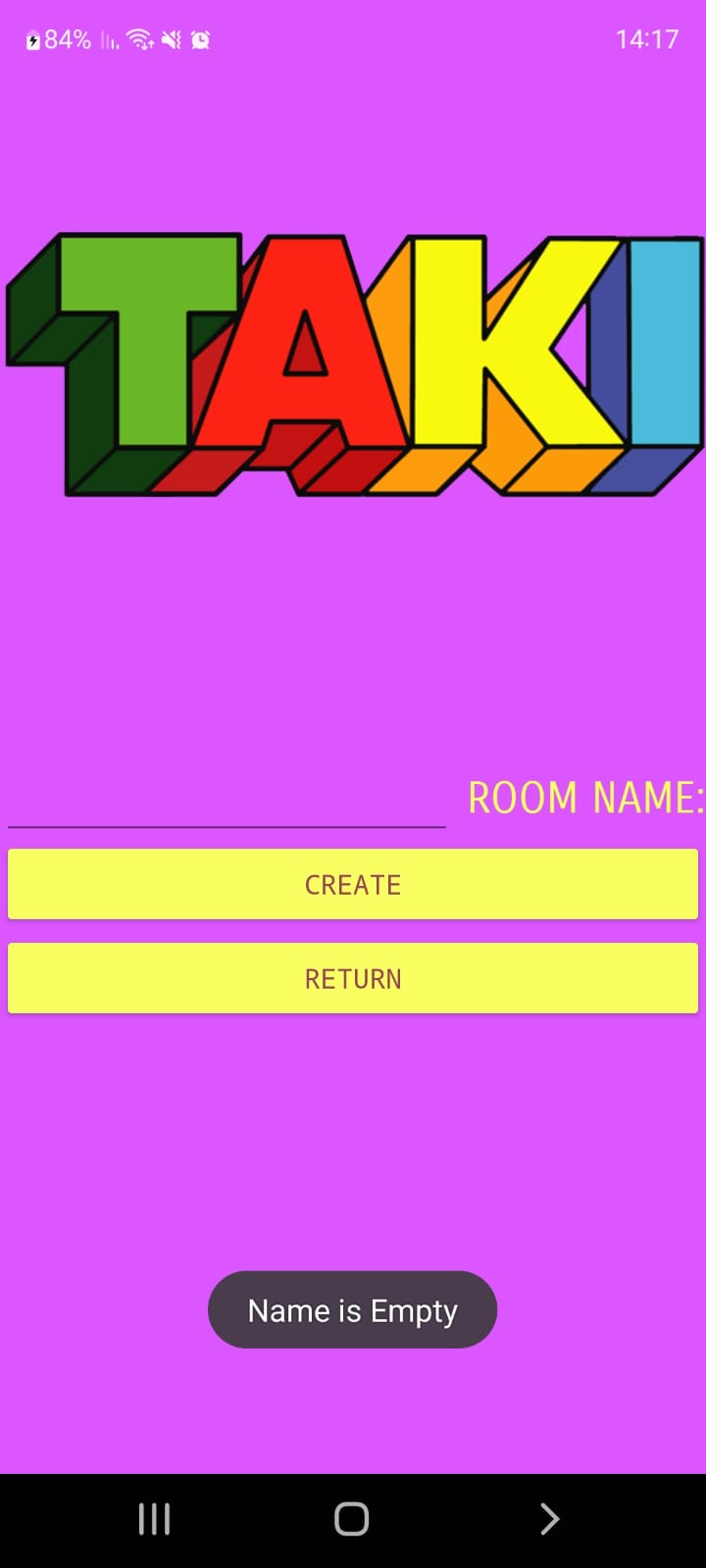
מסך התחברות: שתי תיבות טקסט, המשתמש מקליד – אם אחת מהתיבות ריקה מופיעה התראה.

אם אחד השדות שגוי מופיעה התראה.

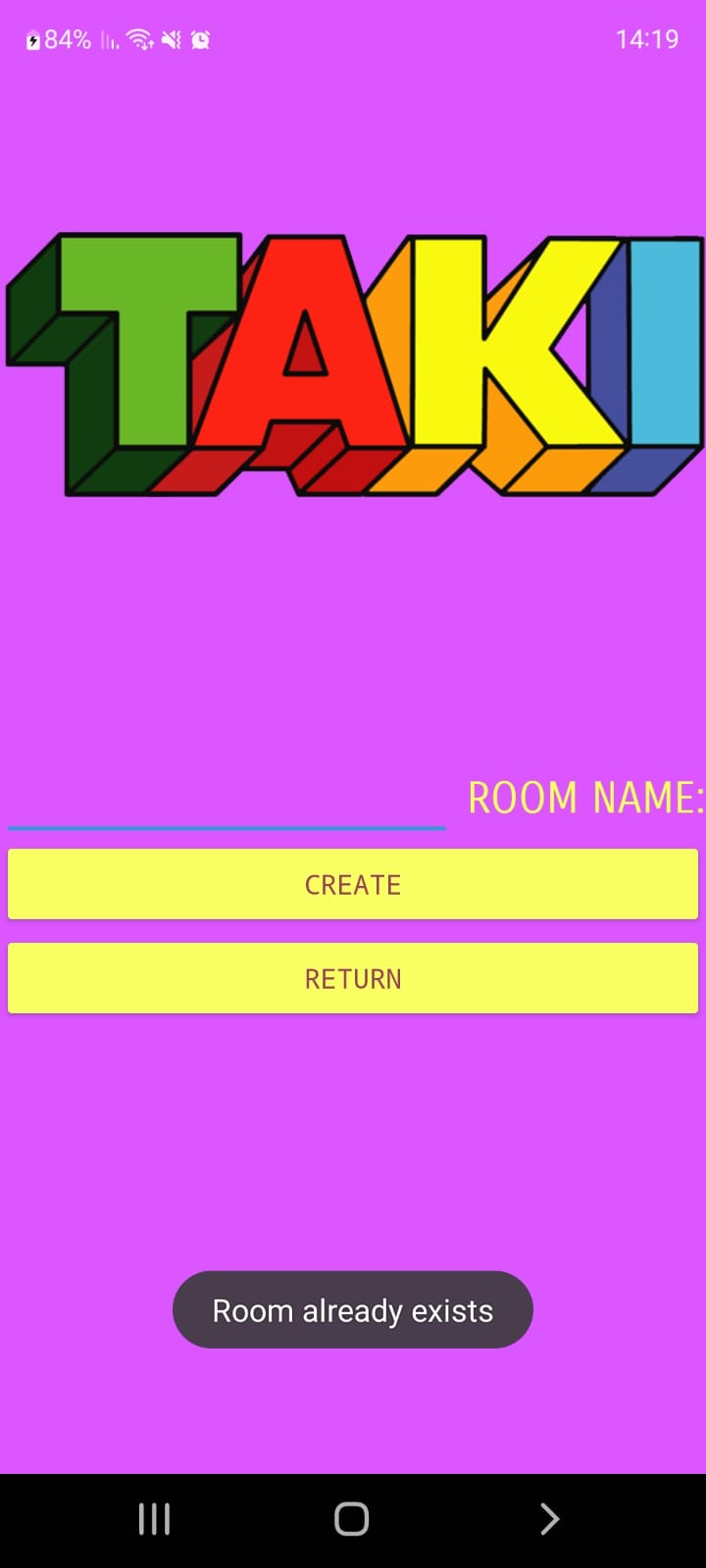


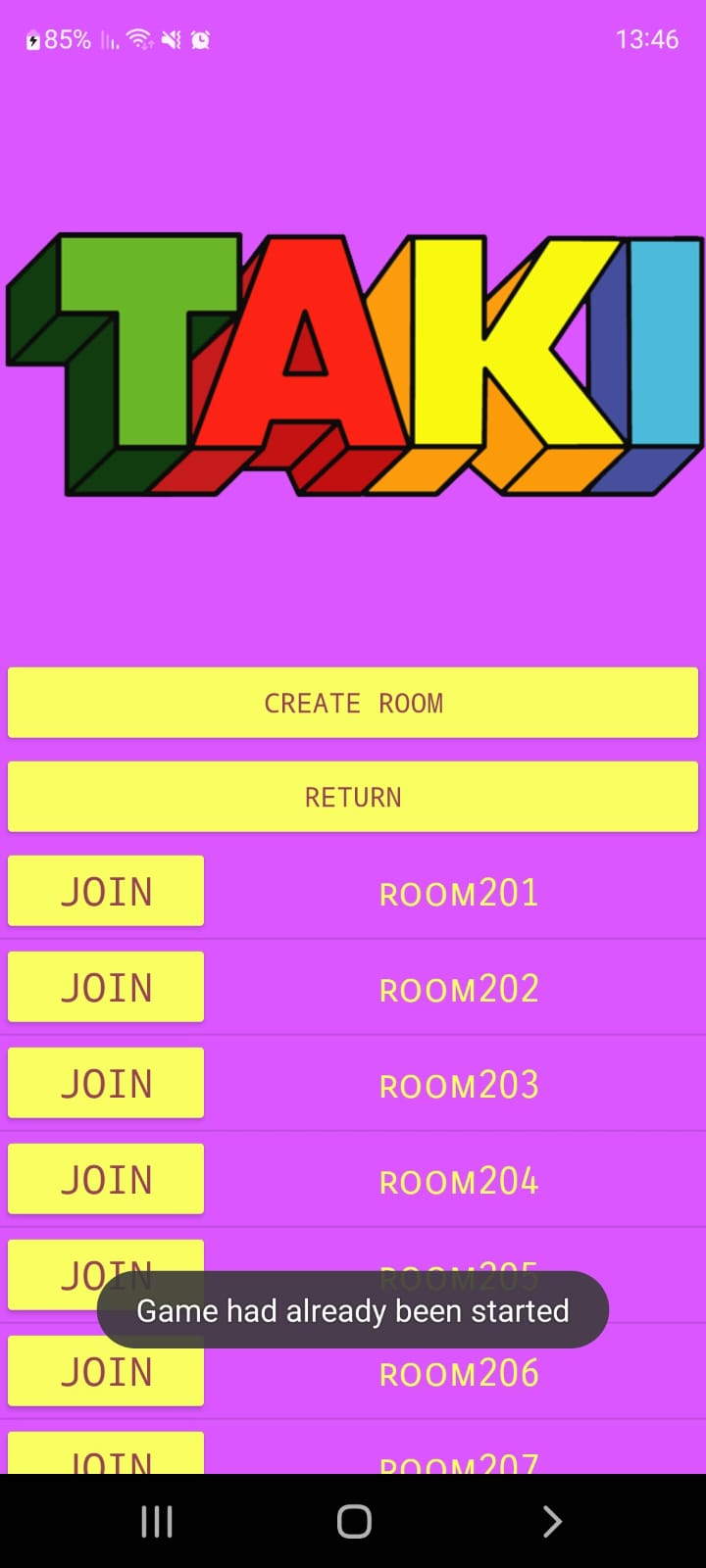
מסך יצירת חדר: כפתור "צור" כפתור "חזור" ותיבת טקסט,

המשתמש מקליד – אם התיבה ריקה מופיעה התראה.

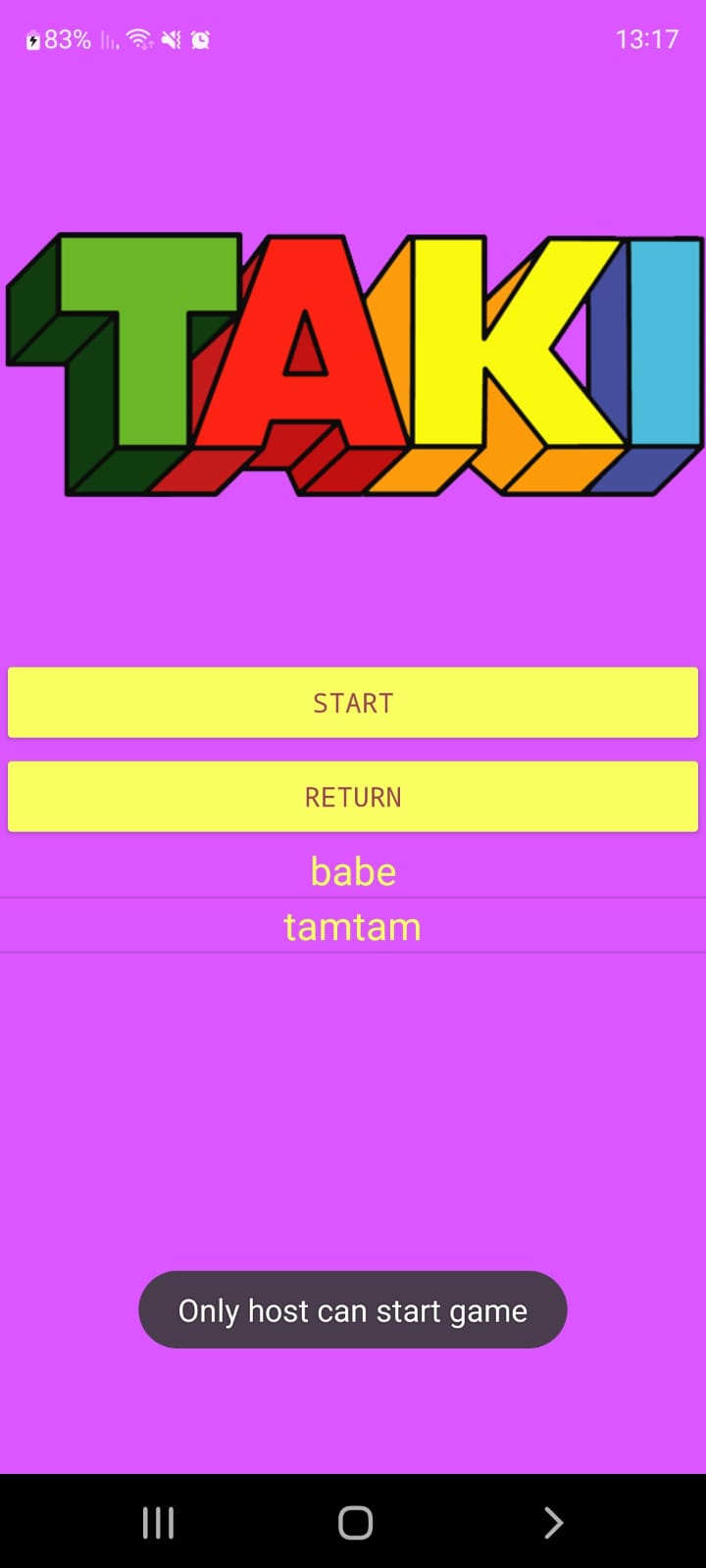


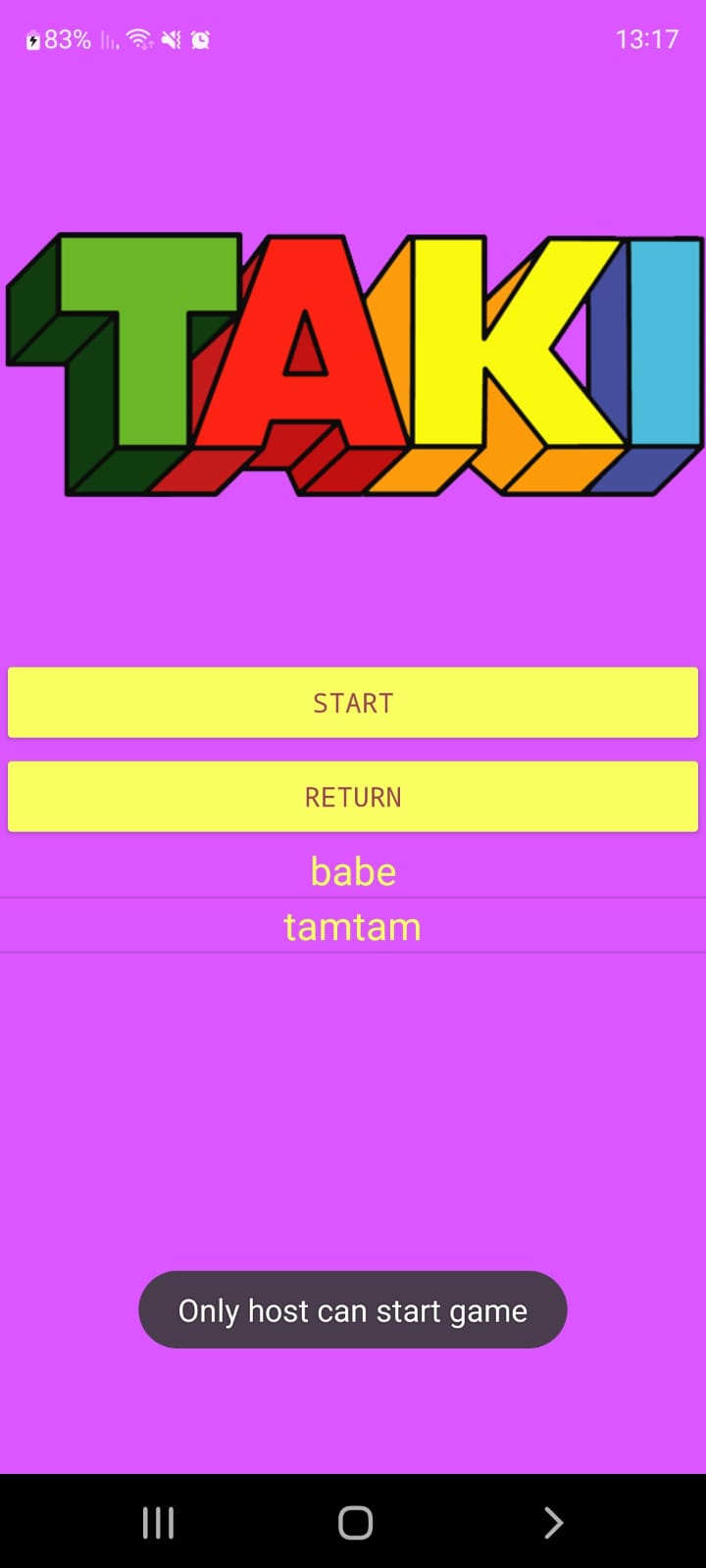
אם החדר כבר קיים מופיעה התראה.



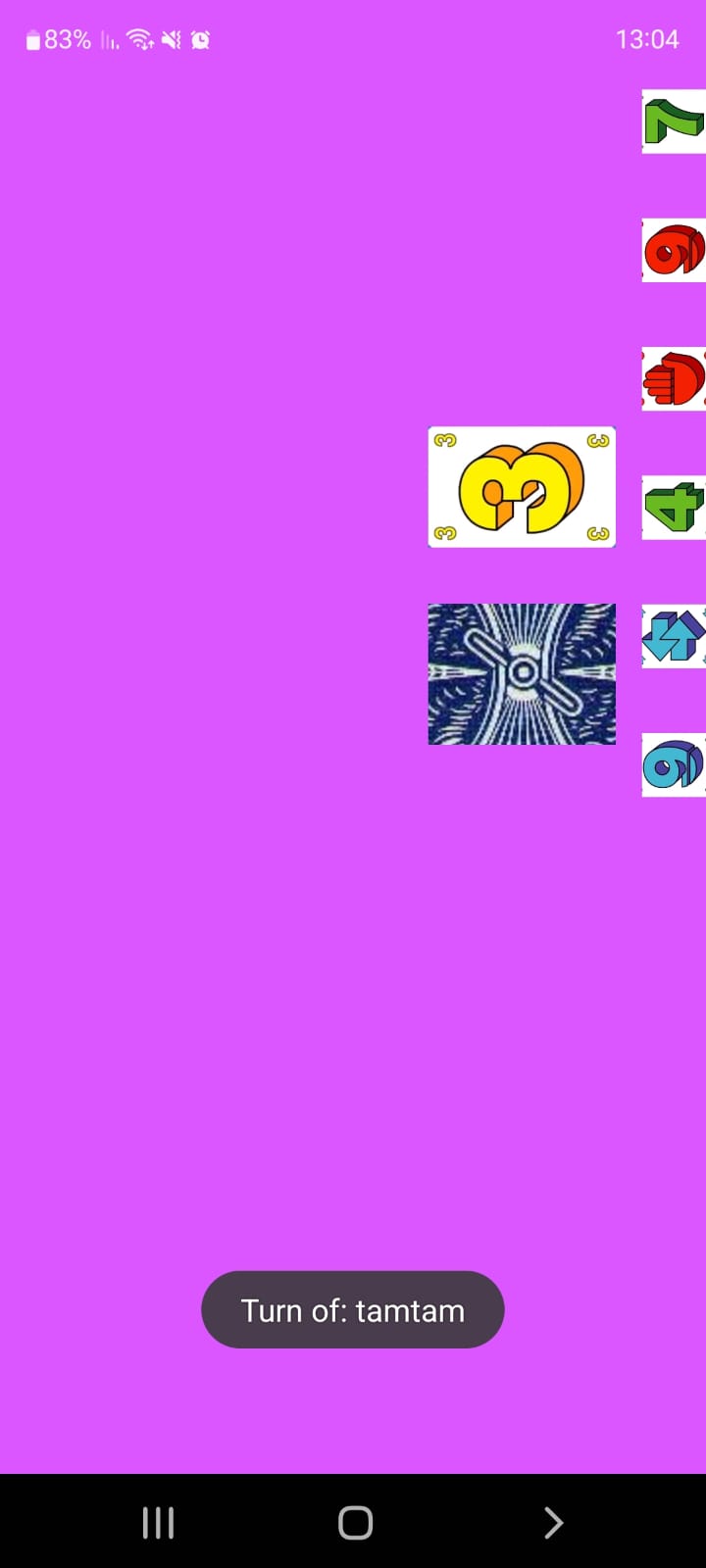
כפתור "JOIN" לחיץ רק אם המשחק עוד לא התחיל:

מסך לחכות שהמשחק יתחיל: רשימה של כל המשתתפים במשחק.

כפתור התחלה – לחיץ רק ל- HOST .

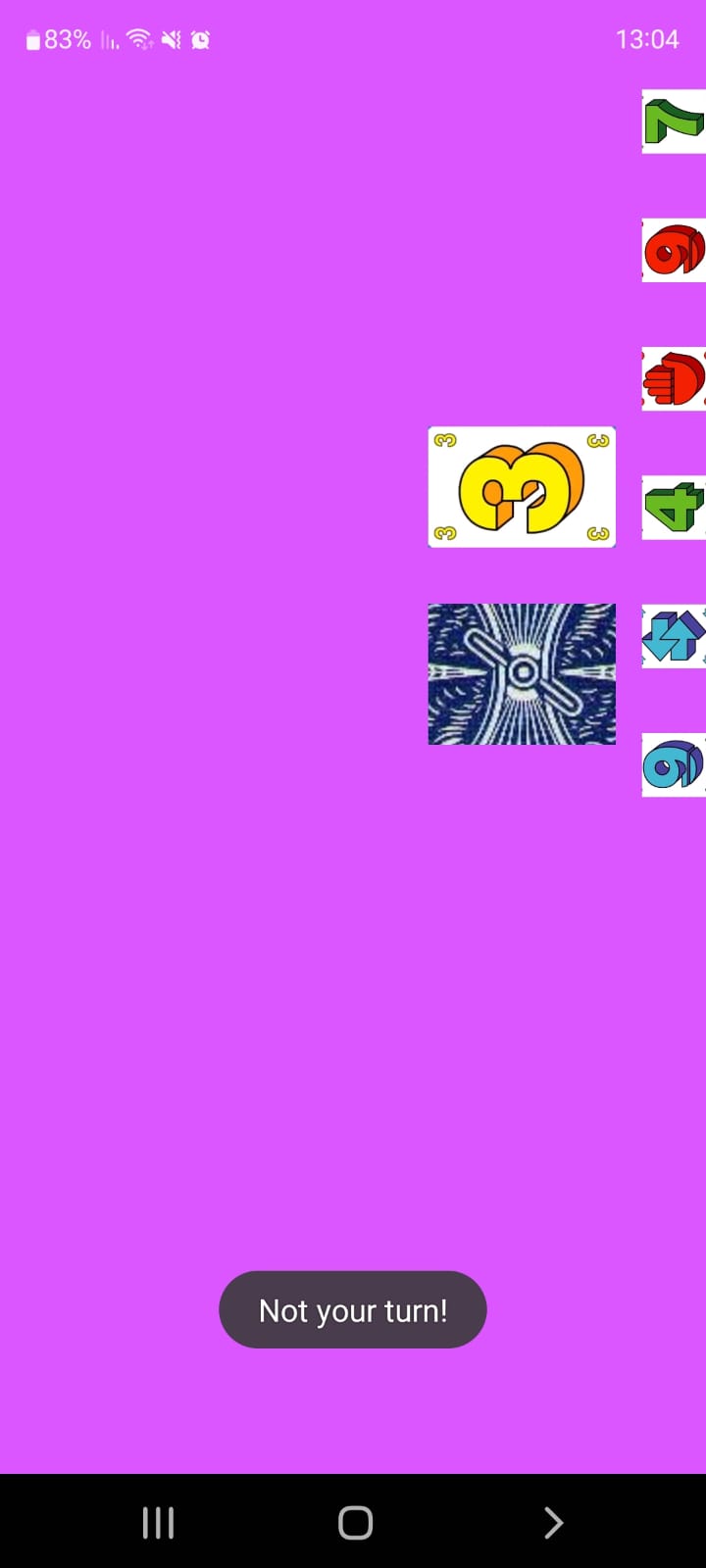
אם משתתף רגיל מנסה ללחוץ על הכפתור אז מופיעה התראה.

מסך המשחק: המסך מורכב מכפתור עם תמונה (ערימה), מתמונה (פח), ומשורה של שמונה כפתורי תמונה (קלפים). אם המשתמש לוחץ על הערימה, כפתור נוסף לשורת הכפתורים שלו.

 בתחילת כל תור מופיעה התראה של מי התור.

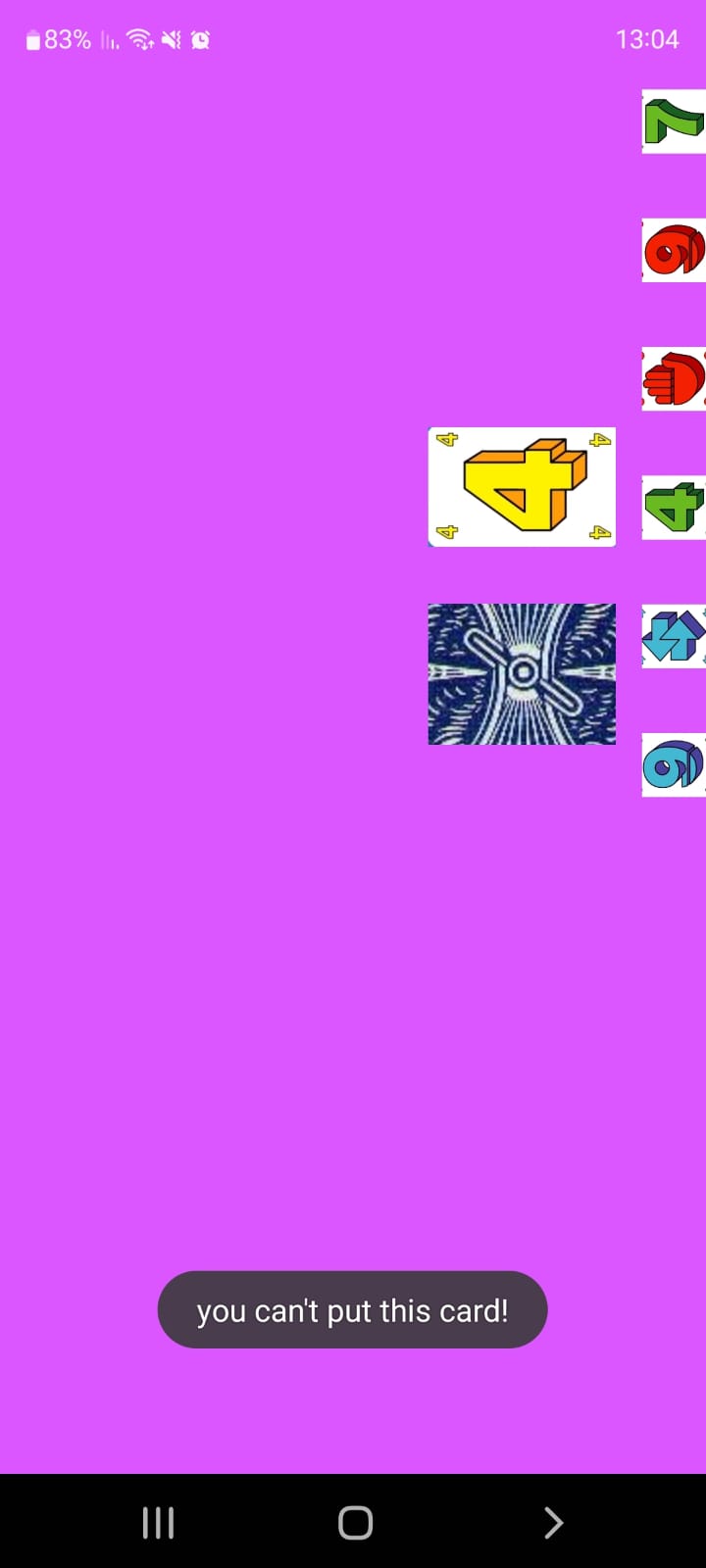


אם המשתמש מנסה ללחוץ על איזשהו כפתור כשזה לא תורו מופיעה התראה.

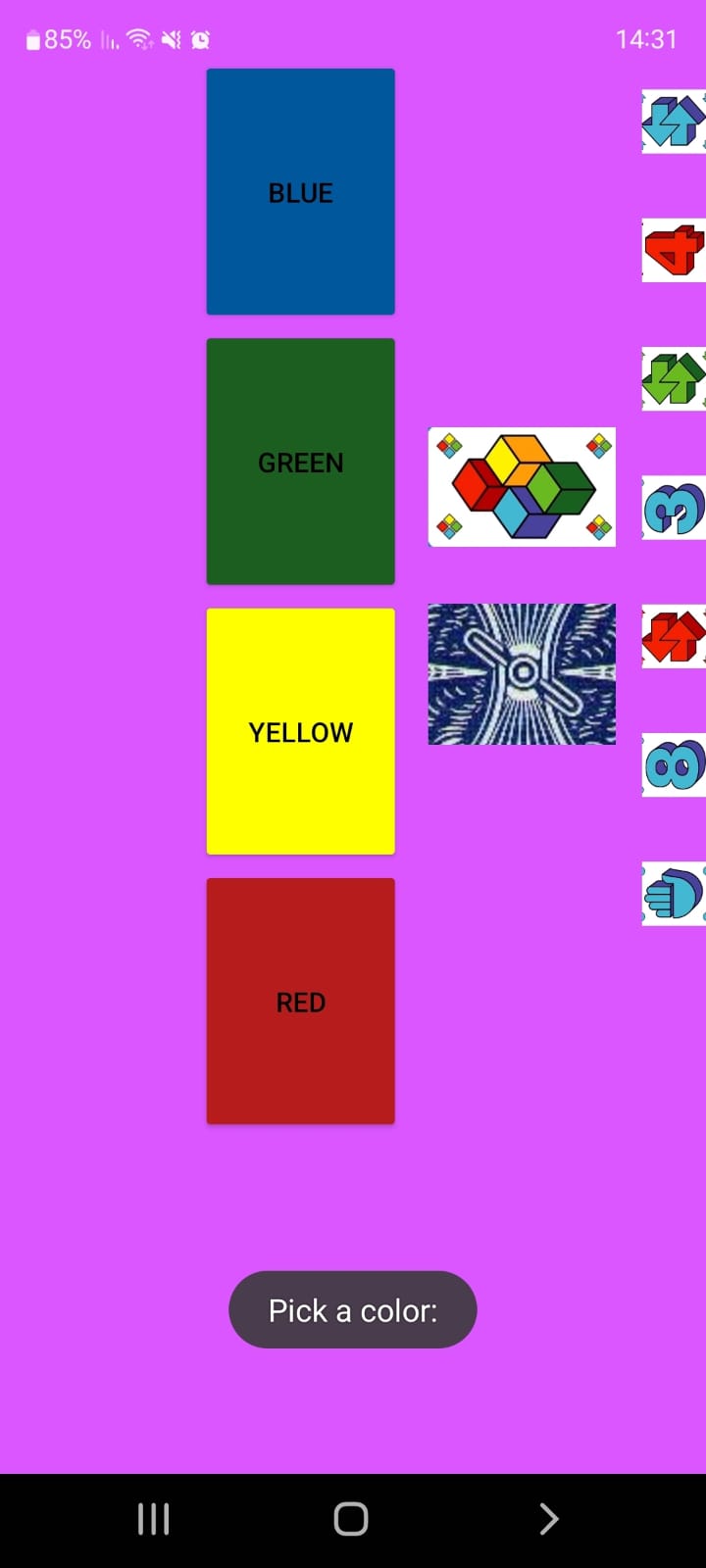


אם המשתמש צריך לקחת קלף מופיעה התראה.

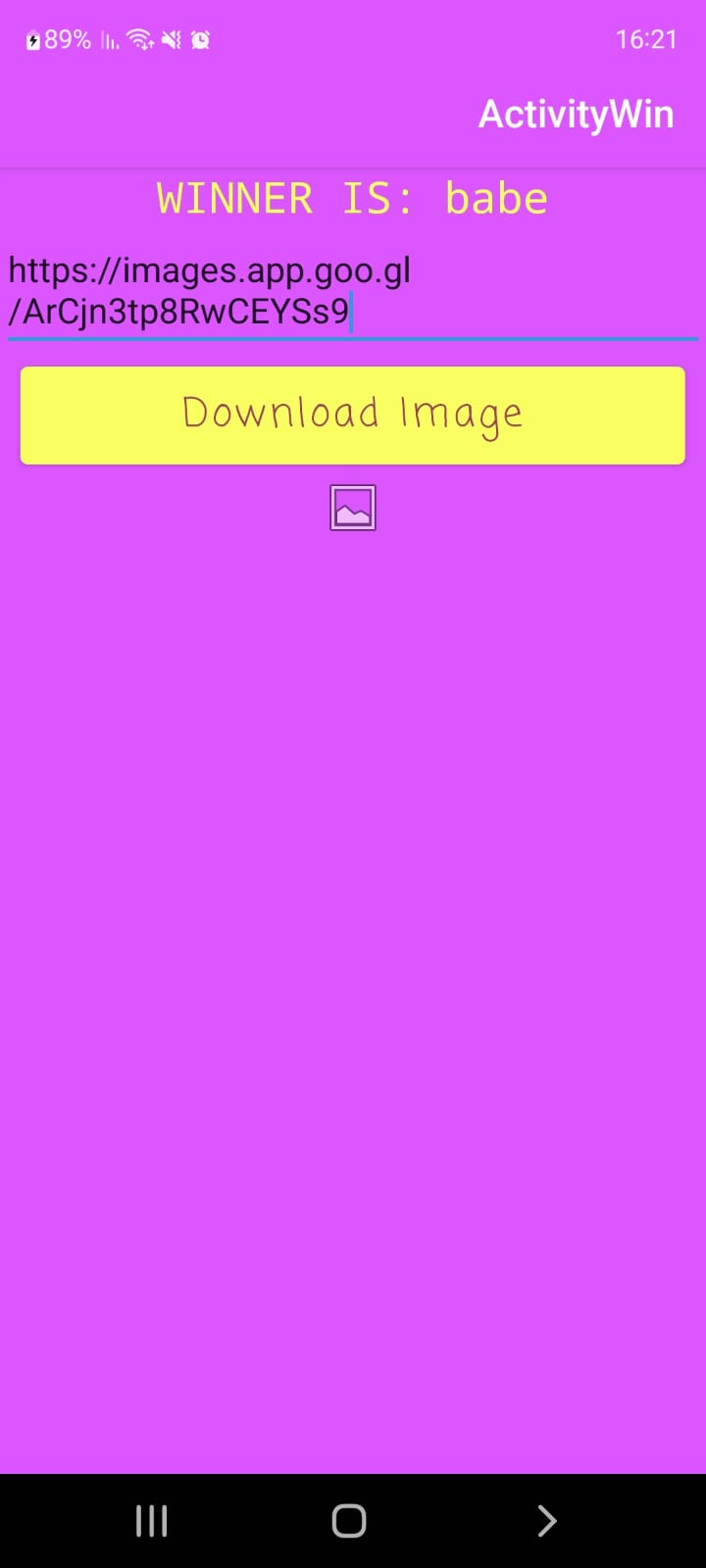
אם המשתמש מנסה להניח קלף לא מתאים מופיעה התראה.



אם המשתמש מניח קלף "משנה צבע" מופיעים לו ארבעה כפתורים והתראה, עד שהוא לוחץ על אחד מהם הוא לא יכול ללחוץ על שום כפתור אחר.



אחרי שהמשתמש לחץ על כפתור הכפתורים נעלמים ומופיעה התראה.

במסך הניצחון: מופיע שם המנצח ותיבת טקסט על מנת להקליד כתובת תמונה.

רפלקציה:

במהלך הפיתוח, נתקלתי באתגרים רבים כגון ניהול המעברים בין המסכים, טיפול בתגובות מהמשתמשים והתמודדות עם בעיות ביצועים. הפרויקט היה מרתק אך מאתגר, כך שלעיתים הרגשתי שאני מתקשה לפתור בעיה מסוימת או שאני עוברת תקופות של ניסיונות וטעויות.

אולם, עם הזמן וההתמודדות עם האתגרים, הגיעה הרגעה מרגיעה ותחושת ההצלחה כאשר התוכנה התחילה לקום על רגלייה והתחילה לתת תוצאות. כל תיקון, כל שדרוג, כל שיפור היה מספר נקודות באישור של ההתקדמות שלי ושל הפרויקט עצמו.

הפרויקט הזה למד אותי להעריך את התהליכים הארוכים והקשים ואת ההתמודדות עם האתגרים. גם אם בעיתי לפתור בעיה מסוימת, עם כל ניסיון וכל טעות, רק הרגשתי שהצלחתי להתקדם קדימה.

בסופו של דבר, כאשר האפליקציה יצאה לאוויר והתחילה לשמש את המשתמשים בצורה יומיומית, הרגשתי גאווה ושביעות רצון עצמית. עם כל האתגרים והמאמצים, הפרויקט נמשך חצי שנה, אך בסופו התוצר היה משמעותי וחוויתי מאוד.

ביבליוגרפיה:

חומרים עיקריים בשימוש-

ויקיפדיה

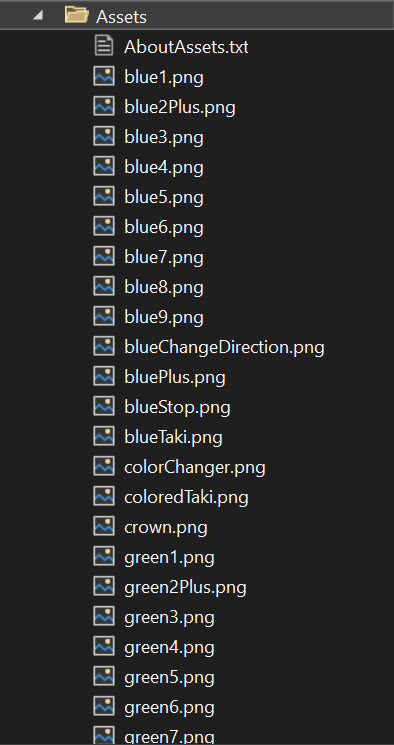
GEEKS FOR GEEKS

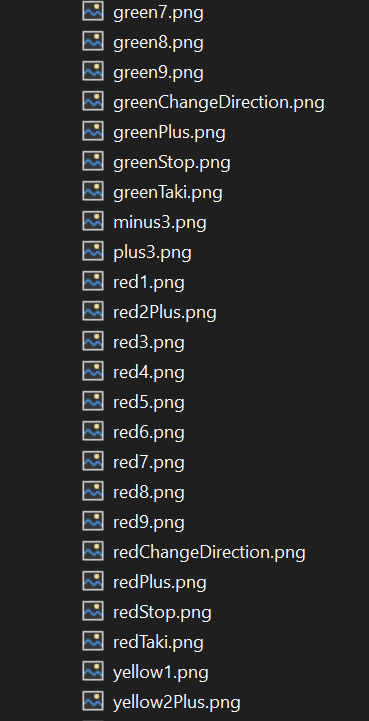
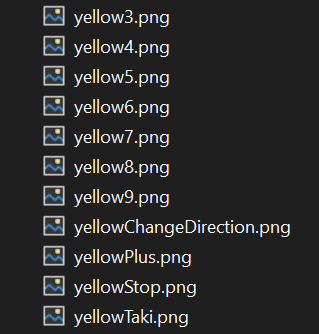
MICROSOFT.COM

CHAT GPT

פרויקטים קודמים שעשינו בכיתה.

נספחים:

 **תיקיית Assets:** מכילה את כל תמונות הקלפים.



זהו