 תאריך: 25.08.19



Final project book

,u

305600934 – דניאל

308192608 - אוראל

311383038 - איציק

208209056 - ליטל

תוכן עניינים:

תקציר פרויקט..............................................................................................................**3**

הגדרת עולם הבעיה......................................................................................................**4**

הגדרת הבעיה והנעשה בתחום.......................................................................................**5**

פתרון..........................................................................................................................**7**

הצדקה אקדמית...........................................................................................................**10**

מתודולוגיות וכלי פיתוח, חומרה ותוכנה...........................................................................**12**

תיאור המערכת- כולל תמונות מסך.................................................................................**13**

תוצרי הפרויקט............................................................................................................**32**

ביבליוגרפיה................................................................................................................**33**

תקציר פרויקט:

מטרת הפרויקט הייתה ליצור תשתית לאפליקציות שונות תוך כדי שימוש בשרת חיצוני לבניית פונקציות שמחזירות למשתמש את הנקודות והתגים אותם הרוויח עבור פעולות מסוימות. בנוסף נוצר שרת web שבו אנו מבקשים מכל מפתחי האפליקציות להכניס את הנתונים הנדרשים כגון חוקים לנתינת נקודות ותגים ופעולות אפשריות באפליקציה אשר נשמרות בFirebase-. הפרויקט נבנה כרגע כאב טיפוס על אפליקציה אותה פיתחה קבוצתנו- אפליקציה רשת חברתית במסגרת פרויקט התמחות. לפרויקט זה הוספנו את כל הפונקציונאליות הנדרשת לצורך הפעלת התשתית.

התשתית נוצרה לניהול משימות/אתגרים עבור משתמשי האפליקציה של אותו המפתח. הוא שקובע את האתגרים כאשר לכל רמה מוגדר ניקוד במהלכה במידה והמשתמש הגיע לניקוד זה, המשתמש עובר לרמה הבאה. בתוך השרת שנוצר מוגדרים כל סט החוקים והתגים שנקבעו על ידי מפתח בשביל להעניק למשתמש בעת הצורך.

כל משתמש מכיל בפרופיל האפליקציה האישי שלו את המזהה של המשתמש, ניקוד (שהצטבר מהשלמת האתגרים והפעולות השונות המוגדרות באפליקציה) וכמות התגים שאותו המשתמש צבר. בלחיצה על התגים ייחשף בפני המשתמש עמוד עם שמות כל התגים אותם הרוויח.

תוך כדי בניית לוגיקת הגיימיפיקציה, ניהלנו ממשק web: שבו קיים עמוד המנהל את הרמות, כאשר המפתח ידאג להוספת הרמות. כל רמה מכילה הגדרות הכוללות: ניקוד עבור כל רמה. כל רמה מעבירה אותנו לעמוד המנהל את האתגרים: לכל אתגר יש הגדרות הכוללות: עריכה, מחיקה וסטטיסטיקה אישית. לכל האתגרים באותה הרמה נקבע ניקוד מראש. המפתח ידאג להוספת האתגרים. בסטטיסטיקה האישית של כל אתגר תופיע הרשימה של כל המשתמשים שקיבלו את האתגר וזמני ההתחלה והסיום. כמו כן יוצג זמן ממוצע לביצוע המשימה.

את לוגיקת הגיימיפיקציה בנינו על גבי שרת שמכיל את ספריית SDK שמתקשרת מצד אחד עם האפליקציה שבנינו (או לחילופין עם אפליקציה שנוצרה על ידי מפתח) ומצד שני עם האתר ה-web.

הגדרת עולם הבעיה:

גיימיפיקציה לוקחת את הרצון הטבעי של המשתמש לחברות, למידה, תחרות, הישגיות, מעמד וביטוי עצמי ומנתח אותו באמצעות תרחישים/סיטואציות ספציפיות.

בין האפליקציות המפורסמות אשר מממשות את נושא הגיימיפיקציה נוכל למצוא את:

* Waze
* Facebook
* Instagram
* Stackoverflow

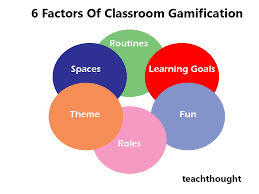
גיימיפיקציה לוקחת את פעולות המשתמש שהוא מבצע באפליקציה ולפי סט חוקים שנקבע מראש הוא מעניק למשתמש נקודות ותגים.

נושא זה הגיע במטרה לעודד השתתפות פעילה של משתמשים בקהילות מקוונות.

אחד המטרות לפרויקט מסוג זה הייתה החיסכון בזמן הפיתוח למפתח שאינו צריך לדאוג ליצור מודל שכזה בכל פעם מחדש אלא לממש את התשתית שמגיעה מוכנה.

לשם כך יצרנו תשתית לכל סוג של אפליקציה ולכל סוג של סביבת פיתוח ועל כן כל מה שנשאר למפתח לעשות הוא להכניס את הנתונים המתאימים לאתר ולחבר את התשתית לאפליקציה אותה הוא יצר.

אך מהיכן המחשבה לנושא שכזה בכלל עלה?

כפי שצוטט ע"י ג'ייקוב נילסן- מקבוצת נילסן נורמן- " ברוב קהילות האונליין 90% מהמשתמשים הינם אורבים אשר אינם תורמים בכלל, 9% מהמשתמשים תורמים מעט, ו1% מהמשתמשים עושים את כל העבודה". כלומר הגענו למצב שלמשתמשים נהיה משעמם להשתמש באפליקציות מאחר ואין שום גורם שמושך אותם להישאר כמה שיותר. כל האפליקציות נהיו מאוד מונוטוניות ומשעממות שכל מטלה שעליהם לעשות מאוד משעממת אותם וגורמת להם לצאת.

לקהילה מספר מטרות:

עבור הממשלה המטרה הוא לתת יעד לכל המידע והשירותים של הקבוצות הספציפיות

אנחנו מזמינים חברים כדי לצאת למסע משקף שבו הם יהיו יותר מוכנים ומודעים לכל הסיטואציות האפשריות

אנחנו מביאים אנשים עם דאגות דומות אחד עם השני כדי לשתף חוויות, רעיונות וטיפים

כל המטרות הללו מעניקות תמיכה חברתית, אמוציונאלית ומוראלית אחד לשני

הגדרת הבעיה והנעשה בתחום:

באשר לבעיות בעולם הגיימיפיקציה ניתן למצוא 3 בעיות מרכזיות עליהם ניתן דגש:

1. בעיית האתחול- כיצד נביא יותר משתמשים אל תוך הקהילה ונקפיד שהם יהיו יותר מעורבים במהלך הזמן.
2. בעיית המעקב- כיצד נעקוב אחר פעילויות המשתמשים בקהילה תחת קטגוריות שונות.
3. בעיית קיימות- כיצד נשמר את פעילות ומעורבות המשתמשים בקהילה לא רק במהלך השלבים הראשוניים אלא למשך זמן חיים ארוך יותר.

בעקבות הבעיות שצצו, הבינו קהילות האונליין שצריך ללמוד יותר אודות המשתמשים והדרך שבה הם משתמשים במוצרי האפליקציה. עליהם להבין מה מושך אותם, מה מרתיע אותם ומה מאתגר אותם.

לשם כך החברות היו צריכים ליצור דוחות מעקב שתיארו את סה"כ המשתמשים הפעילים, מה ואיך כל משתמש יותר מעדיף ובעיקר כמה זמן הם מבזבזים לצורך שימוש בקהילת האונליין.

עם זאת החברות הבינו שהדוחות הללו הם תהליך ארוך, איטי ופרימיטיבי שמצריך המון זמן והשקעה בשביל להבין ולהפיק לקחים. על כן נכנס לתמונה נושא הגיימיפיקציה שמשלבת בתוכה את מערכת מכונת הלמידה שלוקחת בחשבון את כל האלמנטים המיוחדים הללו.

החברות לקחו את האפליקציות והאתרים והפכו אותם בתהליך הראשוני למשהו יותר מושך ויזואלית. הם עבדו על המשתמשים דרך העיניים. כך שהדרך לגרום למשתמשים להשתמש בשירותים שלהם הצליחה אך עדיין לא נמצאה השיטה לגרום להם להישאר יותר זמן ולנצל את השירותים עד מלואם.

עם זאת שגיימיפיקציה נמצאה שיטה מועילה בתחילת דרכה עדיין היה צריך לקחת בחשבון כמה נקודות חשובות לטובת המשתמשים:

* חשוב לשמור על שוויוניות בין המשתמשים
* חשוב לוודא שאין שום חלק במודל הגיימיפיקציה שנותן למשתמשים תחושה של שיפוט.
* חשוב לעודד ולקדם משתמשים שתורמים לקהילה ובנוסף לוודא שלא מפרידים אחרים אשר לא מבצעים זאת.

נשים לב שאחד הדברים המהותיים שעבורם נושא הגיימיפיקציה מומש אך עדיין לא הגיע למה שאנחנו מצפים לו הוא שכל מפתח אשר בונה אפליקציה או מקים אתר אינטרנטי מחויב לבנות בעצמו ולשלב את כל הנושא לתוך הפיתוח שלו. כלומר לא קיימת תשתית גלובאלית המעניקה שירותים למגוון רחב של מפתחים אלא מצריכה פיתוח אינדיבידואלי.

הדרכים השונות לשילוב אלמנטים משחקיים בסביבות שאינם מהווים משחק הם בין היתר:

* מחויבות
* נקודות
* לידר בורד
* תגים

פיתרון:

תפיסת הפתרון שלנו הייתה לבנות תשתית שתשמש את כל מפתחי האפליקציות לכל סביבות הפיתוח כך ש:

* המפתח בונה אפליקציה
* המפתח מוסיף את הכפתורים המתאימים לתוך האפליקציה
* המפתח מחבר את התשתית לאפליקציה שלו

בשלושת נקודות הללו יצרנו חיבור בין התשתית לאפליקציה.

כעת מה שנשאר למפתח הוא להיכנס לאתר ה-web אשר יצרנו ולהכניס לשם את כל סט החוקים להענקת נקודות עבור פעילויות שמשתמש יכול לעשות באפליקציה וסט של חוקים להענקת תגים עבור שלבים שונים שהמשתמש מגיע עליהם אחר פרק זמן באפליקציה.

כל מה שהמפתח מכניס לאתר – נכנס אוטומטית למסד נתונים שלנו שזהו ה-Firebase.

מה שנקבל בסופו של דבר שעבור כל פעולה שמשתמש מבצע, מתבצע טריגר שקורא לפונקציה הספציפית בתוך התשתית שמחוברת לאפליקציה ומעניקה לה נקודות בעת הצורך לאחר בדיקה מול סט החוקים שנמצא ב-Firebase. כל הנקודות והתגים באים לידי ביטוי בתוך הכפתורים שהמפתח בנה בתוך האפליקציה שלו שמציגה אותם.

כפי שכבר ציינו פרויקט זה שונה מאחרים בכך שיצרנו משהו מאוד גלובאלי מאשר אינדיבידואלי. יצרנו את היכולת למפתחים לקחת תשתית שהיא מוכנה מאשר לבנות ידנית עבור כל אפליקציה שמפותחת. כמו כן גם היכולת לחבר בין התשתית לאפליקציה היא מאוד נוחה ומובנת.

בתוך האתר שבו המפתח הכניס את כל החוקים הוא רואה בבירור טבלאות של מה שהוא יצר כפי שגם מופיעים ב-Firebase לאחר יצירתם. קיימת טבלה עבור סט החוקים שהוא שם להענקת נקודות עבור פעילויות וטבלה עבור סט החוקים שהוא שם להענקת תגים עבור כמות נקודות מסוימת.

ציינו בשקופית הקודמת שישנם אלמנטים מרכזיים אותם מפתחים מכניסים לאפליקציה כחלק מנושא הגיימיפיקציה על מנת לעורר תגובתיות ועניין מצד המשתמש.

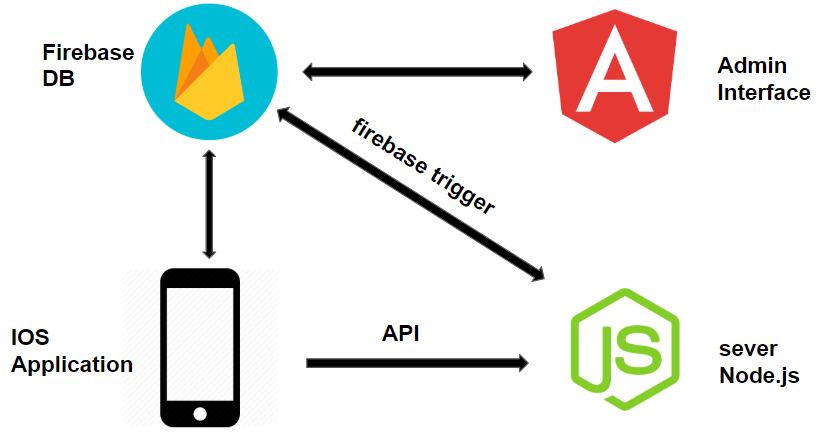
נשים לב שבחרנו לתת דגש לאלמנטים: נקודות ותגים ולא למשל לידר בורד. הסיבה לכך נעולה בנקודות שהעלנו קודם לכן על שמירת שוויוניות ואי שיפוט המשתמש שמגיעה מהמודל עליה הפרויקט התבסס.

**ארכיטקטורת המערכת בנויה מ-4 יחידות מרכזיות:**

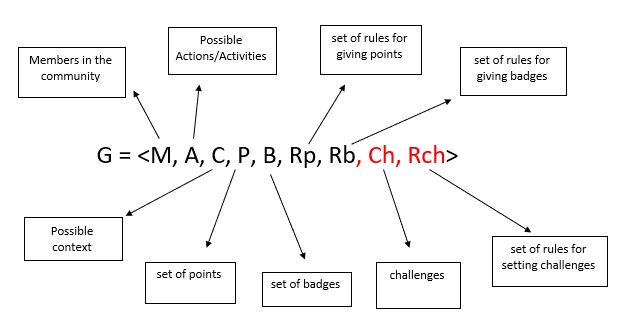
1. מסד הנתונים Firebase - ששומר בטבלאות את כל נתוני החוקים עבור הנקודות והתגים שהמפתח יצר.
2. האפליקציה- אפליקציה אותה המפתח פיתח.
3. שרת Node – שרת המכיל את כל הפונקציונאליות שעומדות מאחורי קבלת הטריגר עבור פעולות שהמשתמש מבצע באפליקציה. זהו השרת אשר מכיל את התשתית והמפתח צריך לחבר אותו אליו.
4. אתר web – אתר שפותח באנגולר שבו המפתח מכניס את כל החוקים לנקודות והתגים אשר הסברנו עליהם קודם.

שניתן לראות בתמונה מטה נראה את כל ההקשרים שבין ארבעת היחידות:

* מפתח בונה את האפליקציה
* האפליקציה מחוברת לשרת דרך API
* ה-Firebase שמהווה מסד נתונים שלנו מחובר עם האתר web שמקבל לתוכו את כל החוקים שנוצרו ע"י המפתח
* וה-Firebase מחובר עם השרת node בכך שהוא מפעיל את הטריגרים באותו זמן שהמשתמש מבצע פעולות מסוימות וקורא למסד נתונים בשביל לקלוט את החוקים



הצדקה אקדמית+ מודולים של המערכת:

נתחיל מהסבר על המודל עליו התבססנו:

ניתן לראות פה את שבעת העקרונות שמוצגים ע"י המודל המקורי ובנוסף עוד שניים שאנחנו הוספנו כחלק מהתהליך החדש (הוספנו את האתגרים עבור המשתמש).

פונקציונאליות והסברים:

* לכל חבר באפליקציה מוענקים מספר נקודות. זה תלוי הן בסוג הפעולה שנעשתה והן בהקשר שלה (Context). מה שהופך את הנקודות להיות מוקצים ומוענקים לחבר אלו הם החוקים/כללים כפי שמוצגים בפונקציה הבאה:

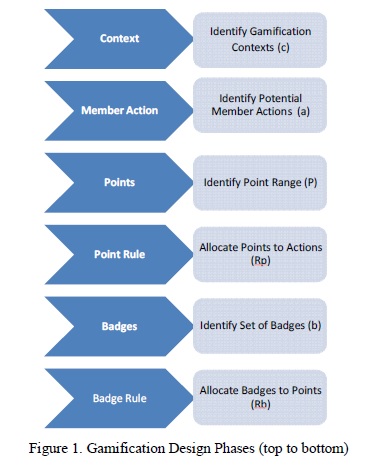
Rp: f(m, a, c) ---> p

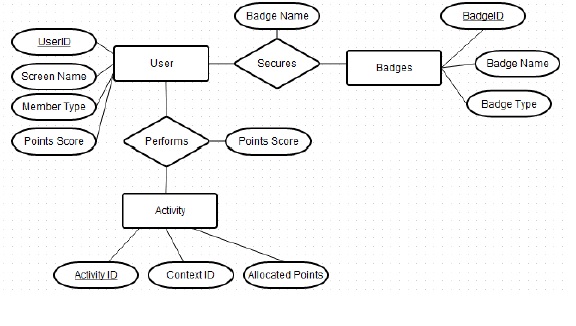
* משתמש יכול לקבל מספר נקודות באמצעות סוגים שונים של פעולות. חבר שיקבל מספיק נקודות יזכה בתג המייצג את הקונספט כפי שבא לידי ביטוי בפונקציה הבאה:

Rb: f(m, a, c, p) ---> b

* התגים יכולים להיות קבועים מה שאומר שאם משתמש קיבל את התג הוא שומר עליו לכל אורך פעילותו באפליקציה. מצד שני, תגים יכולים להיות זמניים מה שאומר שהמשתמש צריך לשמור על התנהגות מסוימת (עקביות בהתנהגות) בכדי לשמור עליו.

תכנון ועיצוב:

* הישגים בניקוד נחשבים להיבט חשוב גיימיפיקציה, שכן הם מהווים בסיס למדידת המשתמש. זה יוצא תחושה של הישג בקרב המשתמשים ועוזר בהנעת המשתמש ועליית המוטיבציה כדי לתרום יותר לקהילה.
* סטטיסטיקות משחקות משמעות חשובה בהשפעה על תגמול, מעמד, הישג ותחרותיות.
* הישגים מכונים נקודות אותם המשתמש מקבל לאחר ביצוע פעולות. בהמשך הנקודות מומרות לתגים שנראים באופן פרטני ע"י החברים כך שיוכלו לשקף את ההישגים שלהם.

ERD Diagram:

מתודולוגיות וכלי פיתוח, חומרה ותוכנה:

* את האפליקציה לצורך סביבת הבדיקה של התשתית שלנו הקמנו באמצעות שפת swift בתוכנת Xcode.
* את התשתית עצמה בנינו בתוך IntellJ. כתבנו אותה בעזרת JavaScript (NodeJS).
* את האתר Web בנינו בתוך WebStorm.

כתבנו אותה בעזרת Angular (web framework)

כלי פיתוח: Xcode, WebStorm, IntellJ

חומרה ותוכנה: מחשב אפל שקיבלנו ממחשוב ביה"ס של מדעי המחשב.

תיאור המערכת- כולל תמונות מסך:

**שרת Node.Js:**

פונקציות בשרת:

**addChallenge, addActionToChallenge, addBadge, addContext, addAction, addRp, addRb:**

פונקציות ששימשו אותנו לבדיקות ומילוי הDB לפני שהרמנו את האתר. כל הפונקציונליות של הפונקציות הנ"ל תפורט בהמשך תחת פירוט האתר.

**addUserActivityA/B/C:**

הן מקבלות מספר ימים ותפקידן "לחקות" התנהגות של משתמש בעל דפוס שימוש יומי מסוים(A\B\C).

**Rp**

מניע (TRIGGER): תוספת של user activity חדש לDB.

היא מנהלת את מתן או שלילה של נקודות עבור משתמשי הקצה של האפליקציה.

הפונקציה שולפת את הUserActivity שנוסף (תאריך (Date) , פעולה (Action) , הקשר (Context), מזהה של המשתמש (UserID)), שולפת את טבלת הניקוד (Rp)

בנוסף , הפונקציה מעדכנת (מוסיפה 1) את סך הפעמים שהמשתמש ביצע את הפעולה הספציפית.

יתרה מזאת , הפונקציה מעדכנת אם יש צורך בהתאם לחוקיות של הניקוד שהוגדר ע"י מנהל האפליקציה (דרך האתר): סך כל הנקודות ואת הנקודות בהקשר הספציפי.

כאשר יש עדכון נקודות עבור משתמש , נוצרת קריאה לפונקציה Rb והיא שולחת לה אתuserActivityID .

**Rb(userActivityID)**

נקראת ע"י Rp לאחר שלמשתמש עודכנו הנק'.

היא שולפת את: userActivity, סך כל הנקודות והנקודות ב Context של הuserActivity של המשתמש הספציפי ומעדכנת את התגים של המשתמש לפי טבלת התגים (שנמצאת באתר).

**challenges**

מניע (TRIGGER): תוספת של user activity חדש לDB.

היא מנהלת את המעקב אחר התקדמות המשתמשים באתגרים ומתן התגים ונקודות בעת השלמת האתגר.

הפונקציה שולפת את הUserActivity שנוסף (תאריך (Date) , פעולה (Action) , הקשר (Context), מזהה של המשתמש (UserID)) ושולפת את קובץ האתגרים (Challenge).

עבור כל אתגר פעיל שנמצא בקובץ האתגרים ולא הושלם עדיין , היא בודקת האם המשתמש ביצע פעולה ששייכת לאתגר(תת אתגר) ומעדכנת את הסכימה שלו אצל המשתמש. אם תת אתגר הושלם היא מעדכנת את זה אצל המשתמש. אם כל תתי אתגר פעיל הושלמו היא מוסיפה למשתמש נקודות (לסה"כ הנקודות שלו) ומוסיפה לו תג בהתאם לאתגר.

**addBadgeToUser(user\_id, badge\_id)**

הפונקציה מקבלת שם משתמש ותג , מחלקת את התג למשתמש (העתקה מלאה של התג).

**addPointsToUserActivity(points, userActivityID)**

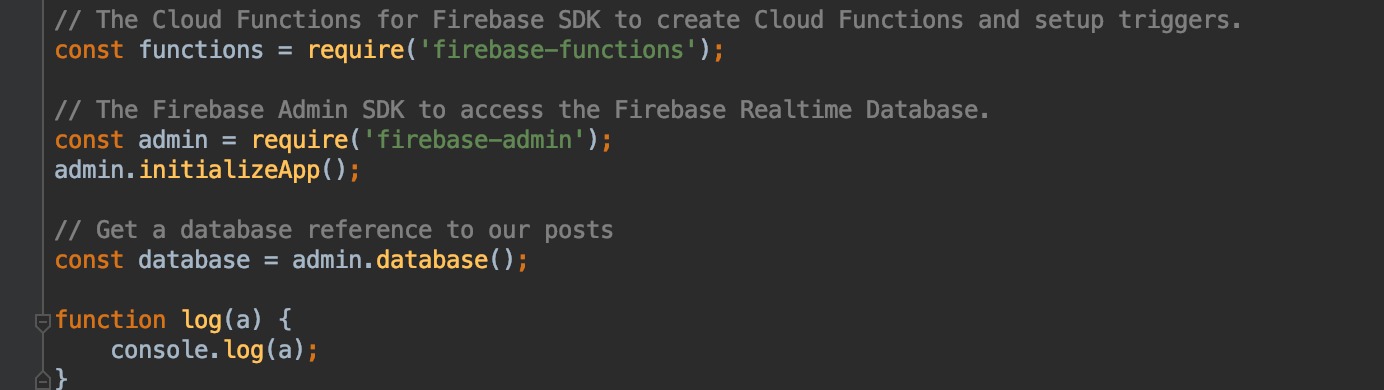
**addBadgeToUserActivity(userActivityID, bType)**

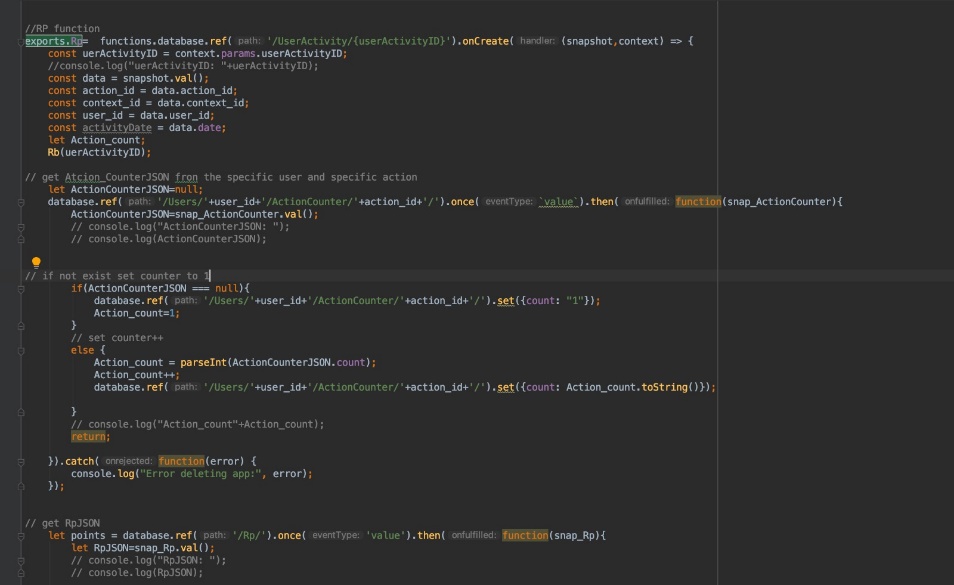
פונקציות עזר שמעדכנות את הניקוד והתגים שחולקו עבור הUserActivity הספציפי על מנת להציג זאת בתור סטטיסטיקה באתר שלנו.

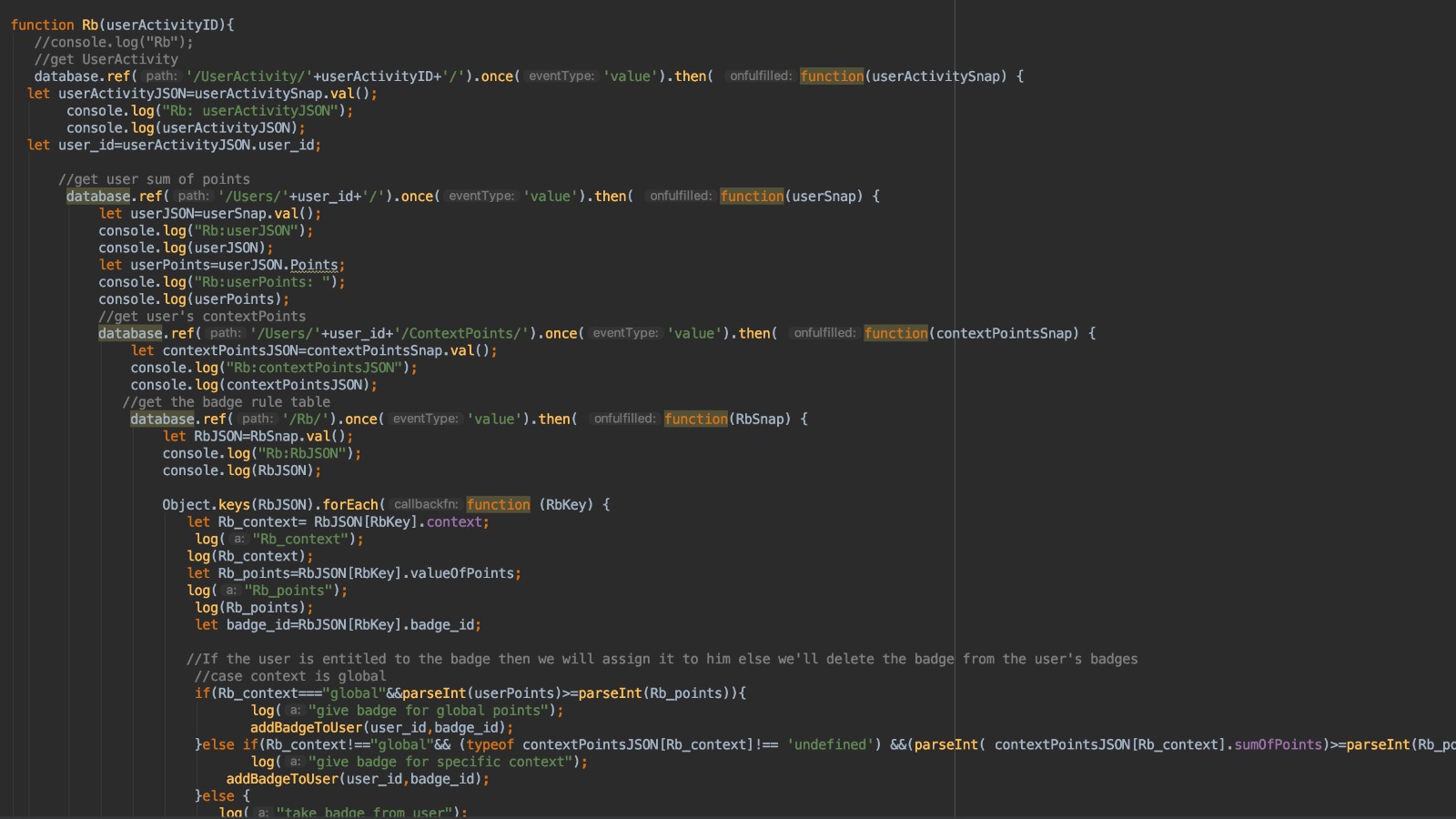
**addUserActivityApp**

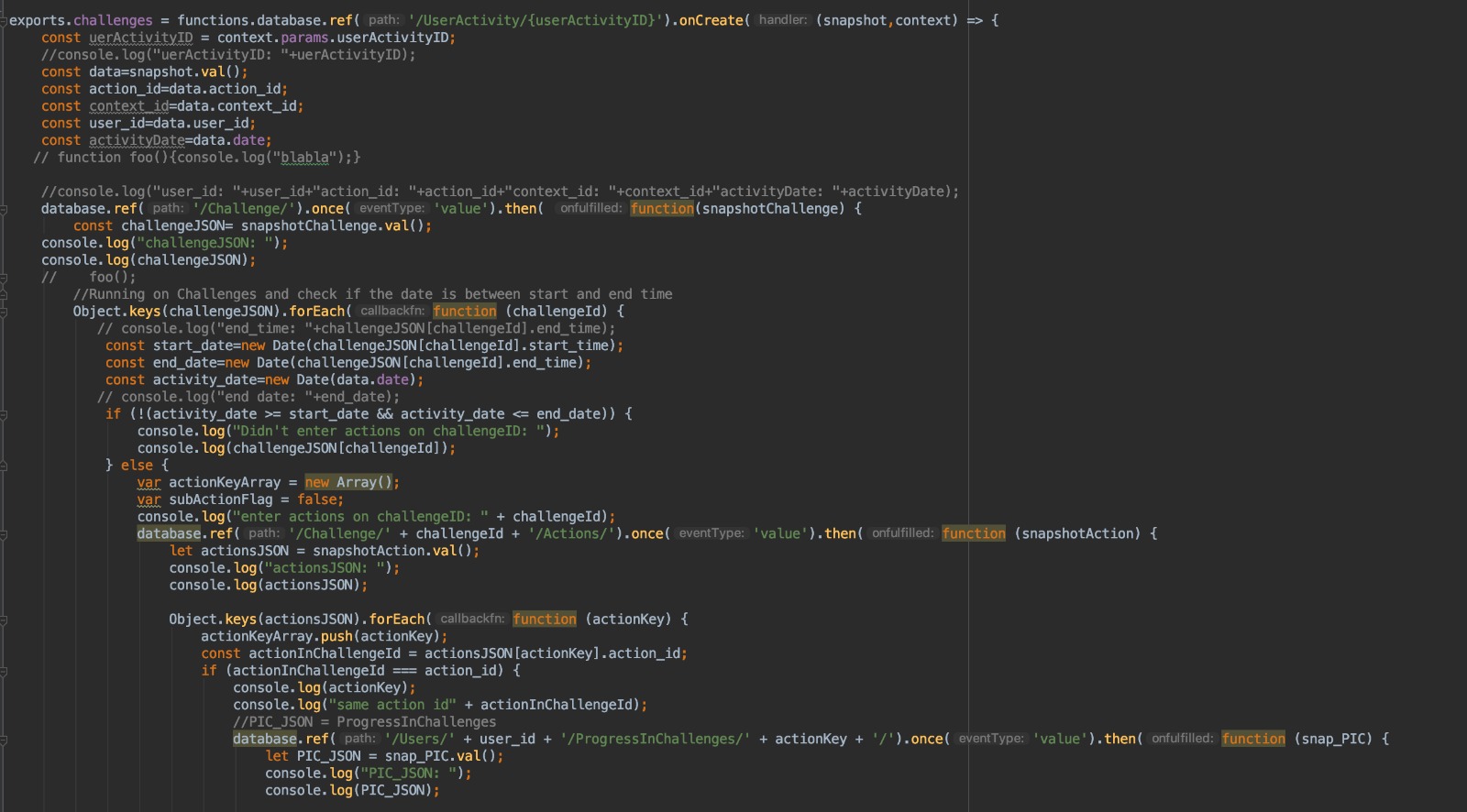
הוספת addUserActivity דרך האפליקציה.

תמונות שרת בפעולה:

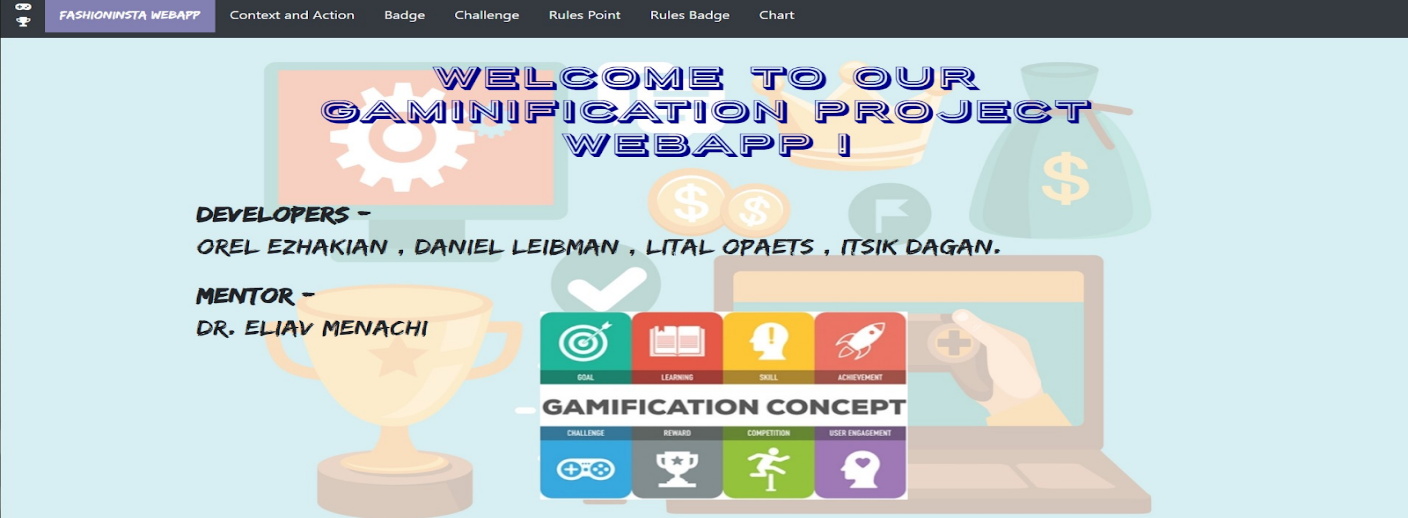
* התחברות השרת ל-Database מרוחק הרץ בטכנולוגיית Documents. (Firebase)
* דוגמה לתחילה של עבודה על פונקציית Rp, Rb ו-Challenges:

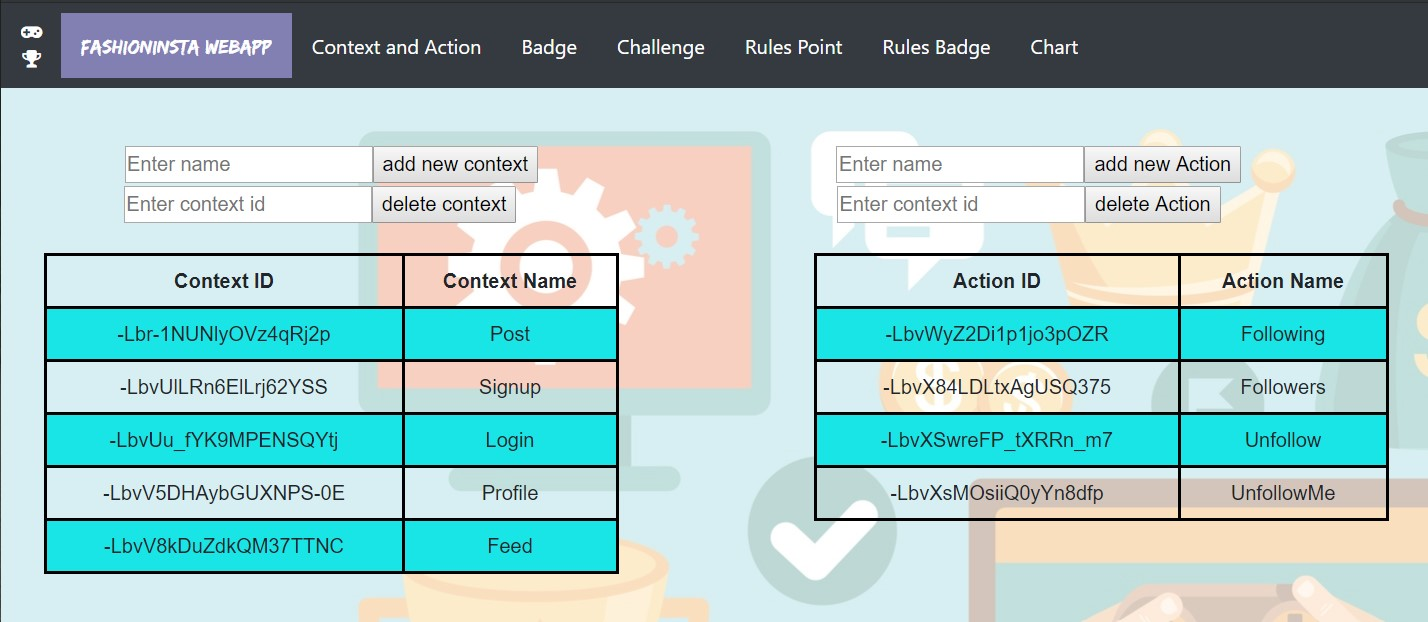




**האתר:**

האתר בא לתת למתכנת מודל נוח יותר על מנת שיוכל לקבוע את החוקיות וההגדרות שלפיהן ינהל את כל מודל הGamification.

בטבלאות האתר מוצגים המזהים מהDB (הID) בגלל שהמתכנת צריך לדעת אותם על מנת שיוכל להטמיע אותם באפליקציה עם פונקציית addUserActivityApp

  
מסך הבית של האתר: בחלק העליון באתר מוצג סרגל עם כל הפונקציות שמומשו , בדף עצמו מוצג דף נחיתה של האתר , המפתחים שלו ומנחה הפרויקט.

לשונית Context and Action :

בדף זה יוצג עבור כל Context ספציפי את הAction-ים הקשורים אליו כך שע"י לחיצה על ה-Context הרצוי בטבלה השמאלית נוכל לראות את כל הפעולות שניתן לבצע בו בטבלה הימנית. בנוסף , יהיה ניתן לבצע הוספה ומחיקה עבור Context וAction מעל כל טבלה- הוספה/מחיקה כזו ניתן יהיה לראות מיידית גם בטבלאות באתר וגם ישירות ב-Firebase.

כפי שניתן לראות בתמונה כל טבלה מכילה 2 עמודות: עמודת שם ה-Action/Context ועמודת ה-Action/Context id שזהו המספר הייחודי כפי שגם יופיע ב-Firebase.

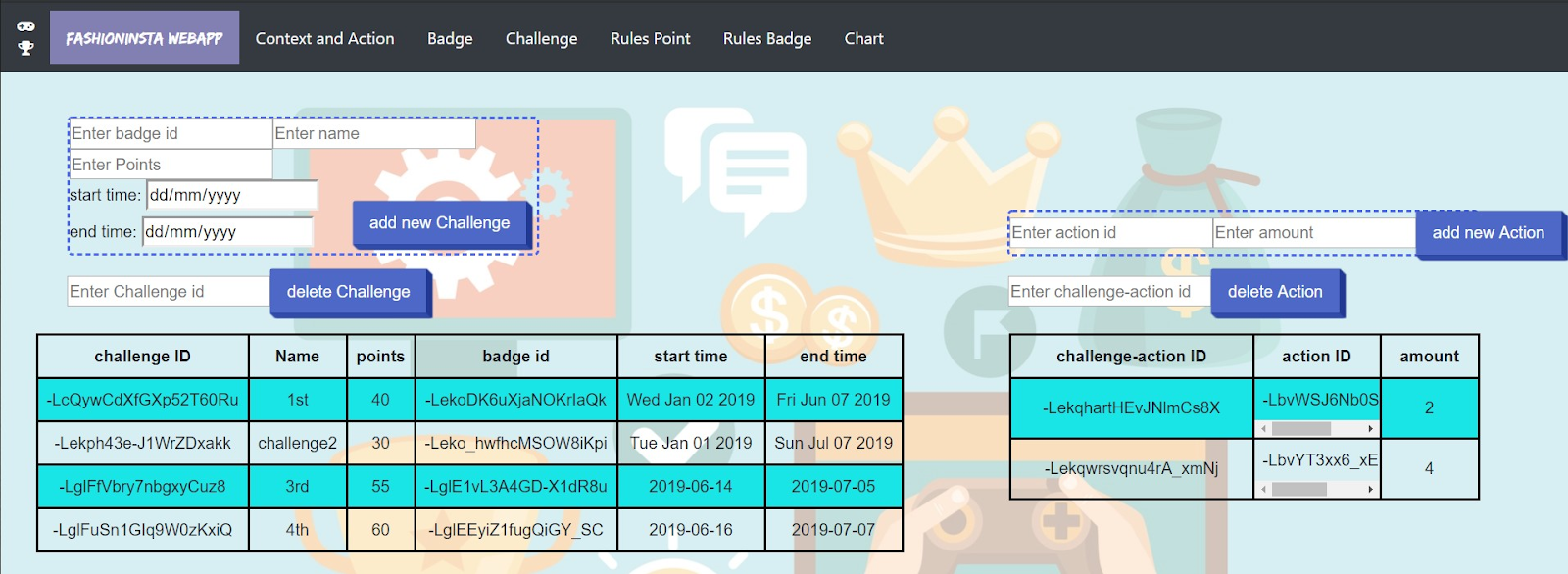


לשונית Badge:

בדף זה יוצגו לנו בטבלה סוגי ה-Badge-ים שהגדרנו (כלומר שמותיהם) , המספר הייחודי של ה-Badge כפי שיופיע גם ב-Firebase, לאיזה Context ספציפי הוא שייך (בהתאם ל-Context id) וקישור לתמונה שלו.

בנוסף תהיה לנו את יכולת ההוספה והמחיקה של Badge כרצוננו.

הוספה/מחיקה כזו ניתן יהיה לראות מיידית גם בטבלה באתר וגם ישירות ב-Firebase.

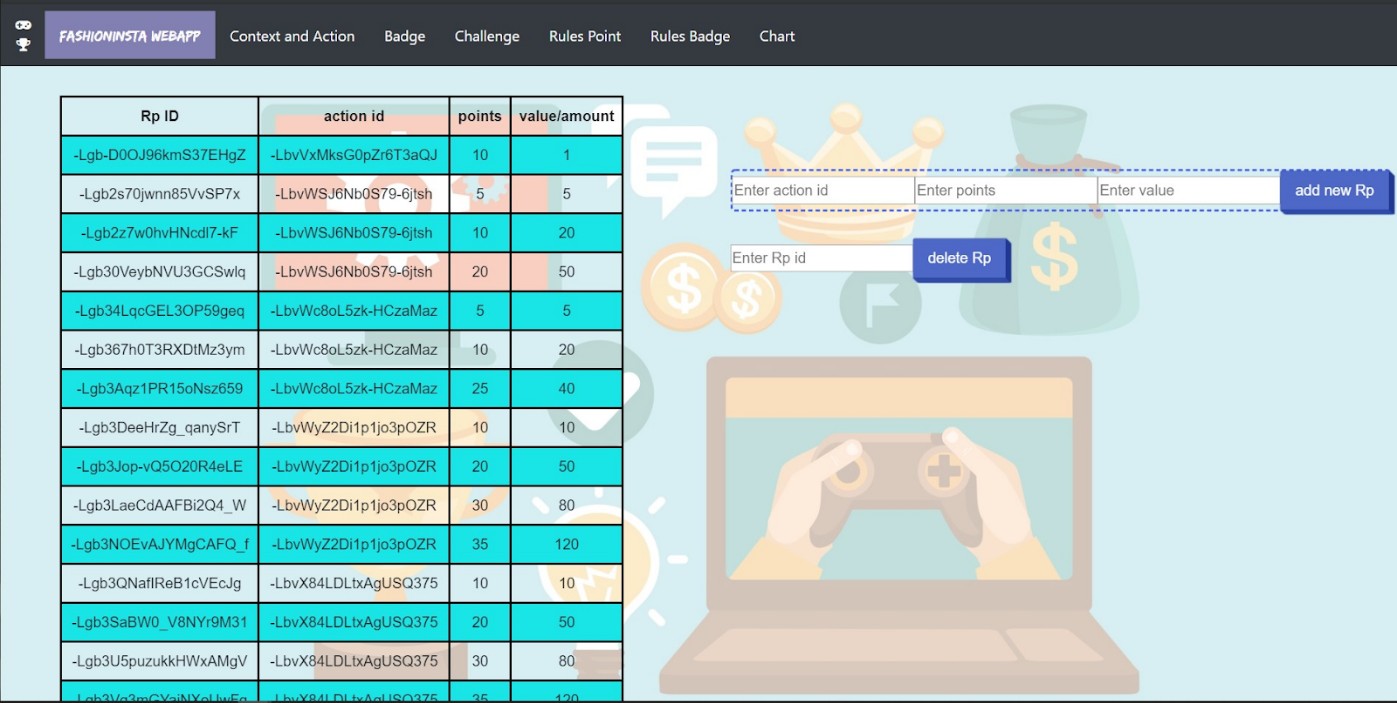


לשונית Challenge :

בדף זה יוצגו לנו רשימת כל הChallenges (האתגרים) שהגדרנו (השם של האתגר, המספר הייחודי של האתגר כפי שגם יופיע ב-Firebase, הנקודות שנקבל לאחר השלמתו , איזה תג ספציפי נקבל כאשר נשלים את האתגר (בהתאם ל-Badge id) , תקופת הזמן שהאתגר נמצא בתוקף (התחלה וסיום) ותמונה).

עבור כל לחיצה על אתגר יוצגו לנו תתי הפעולות בטבלה הימנית , כלומר תתי האתגרים שעלינו להשלים על מנת לסיים את האתגר במלואו. בטבלה זו מוצגים לנו: ה-Action אותו עלינו לעשות (לפי ה-Action id), המספר הייחודי של המשימה עבור אתגר זה שיופיע גם ב-Firebase וכמות הפעמים שעלינו לבצע את הפעולה הספציפית הזו.

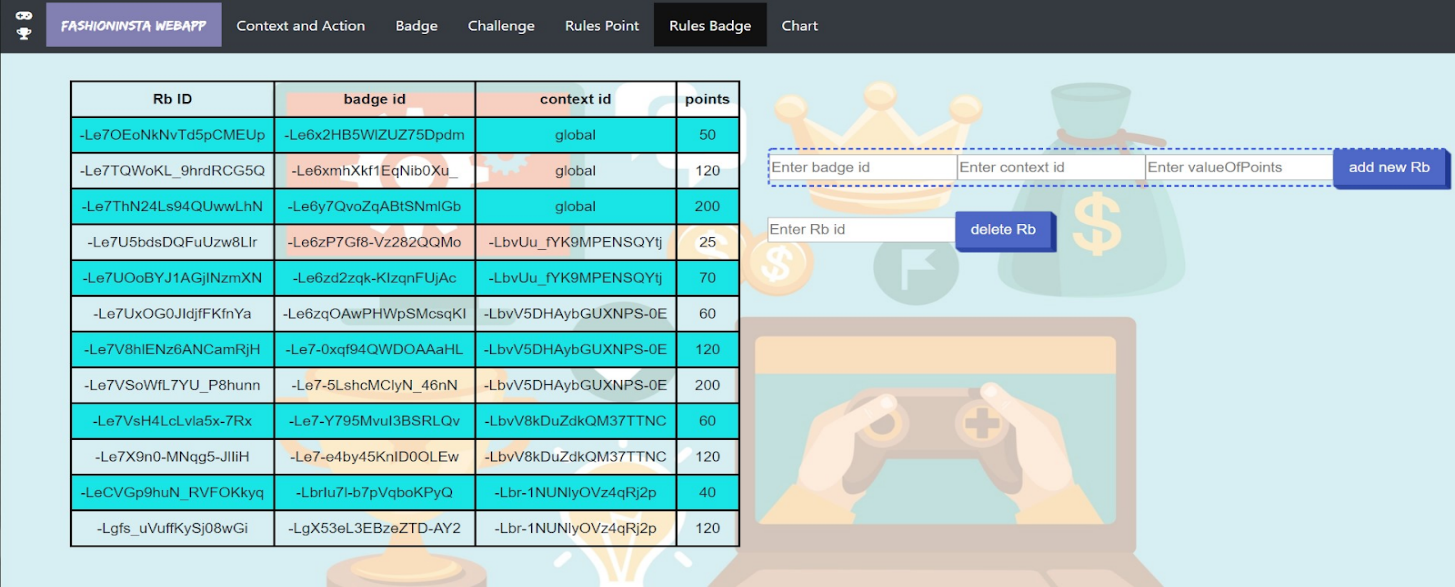
בנוסף בעמוד זה קיימת לנו האפשרות להוספה ומחיקה של אתגר ושל המשימות שלו. הוספה/מחיקה כזו ניתן יהיה לראות מיידית גם בטבלאות באתר וגם ישירות ב-Firebase.

  
לשונית Rules Point :

בדף זה יוצג לנו חוקיות מתן הנקודות למשתמש, כלומר עבור כל פעולה מוגדרת באפליקציה יהיה כמות פעמים שעל המשתמש יהיה לבצע את הפעולה על מנת לקבל את הניקוד שהמתכנת מחליט.

בטבלה זו נוכל לראות: ה-Action עבורו ניתן את הניקוד, המספר הייחודי של ה-Rp שנוצר שיופיע גם ב-Firebase, כמות הפעמים שצריך לבצע את הפעולה כדי לקבל את הניקוד ומספר הנקודות אותו המשתמש יקבל.

בנוסף בעמוד זה קיימת לנו האפשרות להוספה ומחיקה של חוקיות מסוימת. הוספה/מחיקה כזו ניתן יהיה לראות מיידית גם בטבלה באתר וגם ישירות ב-Firebase.



לשונית Rules Badge :

בדף זה יוצג לנו חוקיות מתן התגים כלומר עבור כל פעולה מוגדרת באפליקציה יהיה כמות נקודות שעל המשתמש יהיה להגיע  בcontext על מנת לקבל את התג שהמתכנת מחליט.

בטבלה זו נוכל לראות: המספר הייחודי של ה-Rb שגם יופיע ב-Firebase, התג שהמשתמש יקבל כאשר יגיע למספר הנקודות שהוחלט (לפי Badge id), ה-Context שעליו המשתמש מקבל את הנקודות וסה"כ הנקודות שהמשתמש צריך לצבור בשביל לקבל את התג.

בנוסף בעמוד זה קיימת לנו האפשרות להוספה ומחיקה של חוקיות מסוימת. הוספה/מחיקה כזו ניתן יהיה לראות מיידית גם בטבלה באתר וגם ישירות ב-Firebase.



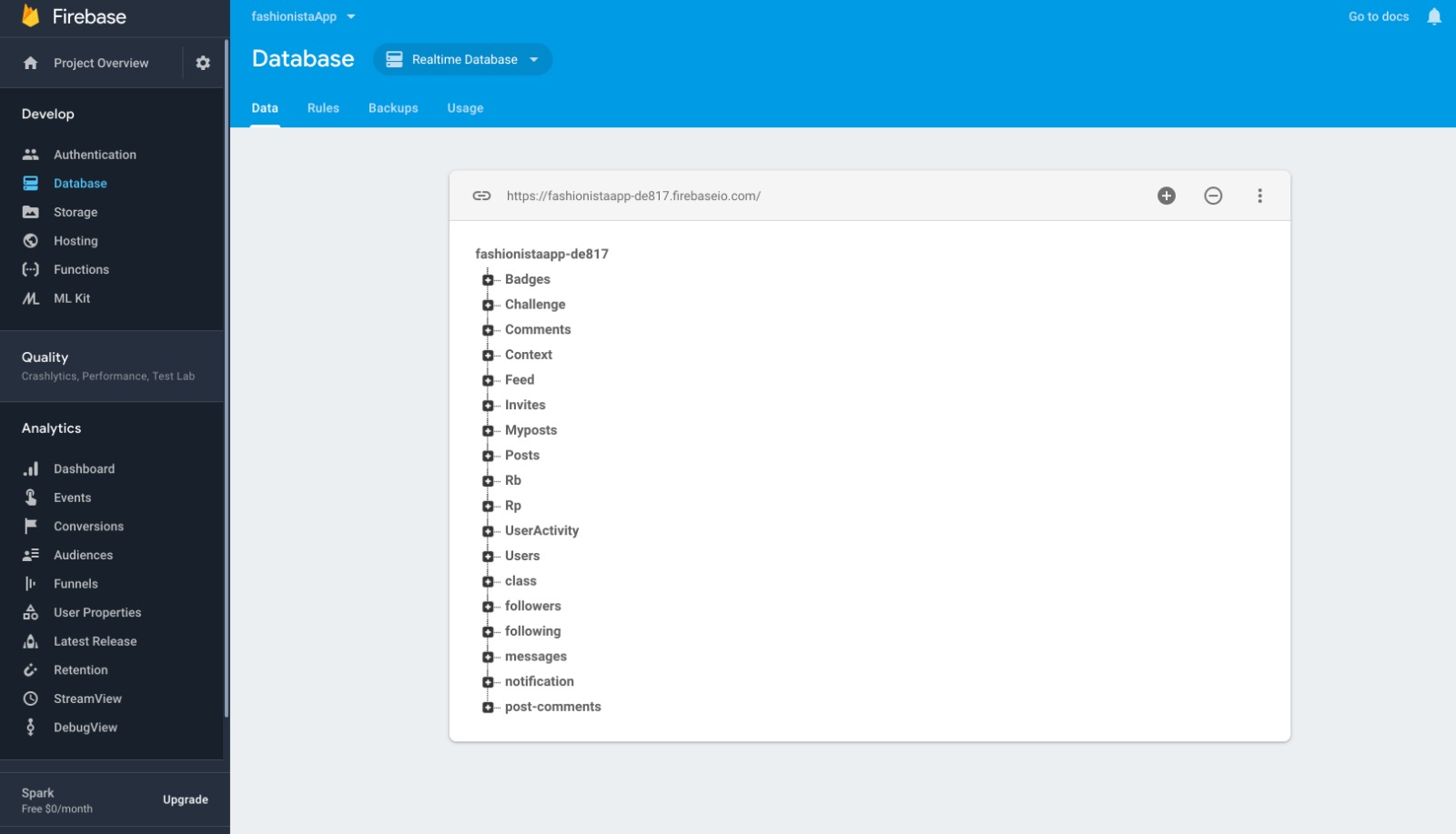
לשונית Chart:

יש גרף שמציג את סך כל הנקודות (עמודה ירוקה) והתגים (תג מטבלת Rb באדום, תג של אתגר בכחול) על פני ציר זמן. ניתן לבצע סינון לפי משתמש ו/או context.

**Firebase:**  
המסד נתונים זה נועד בכדי לאחסן את כל המידע שהמתכנת קובע בתוך האתר.

לאחר כניסה ראשונית של המתכנת לאתר ולאחר מילוי כל הפרטים ייוצרו במסד הנתונים הטבלאות הבאים:

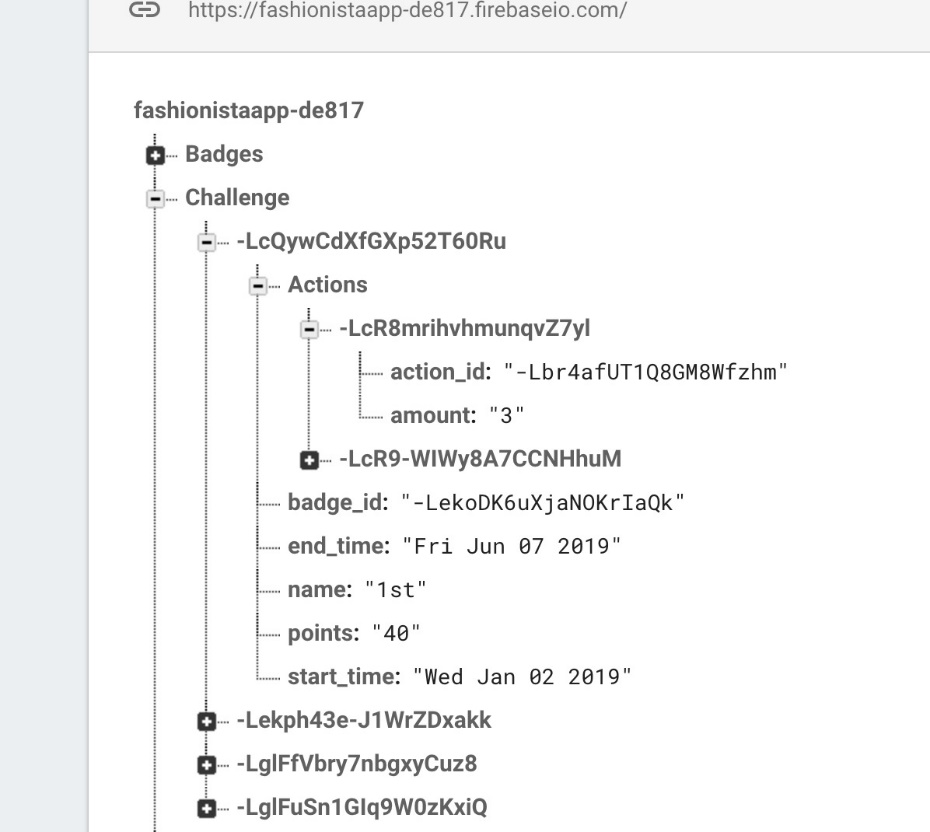
* Context
* Action
* Challenge
* Badge
* Rb
* Rp



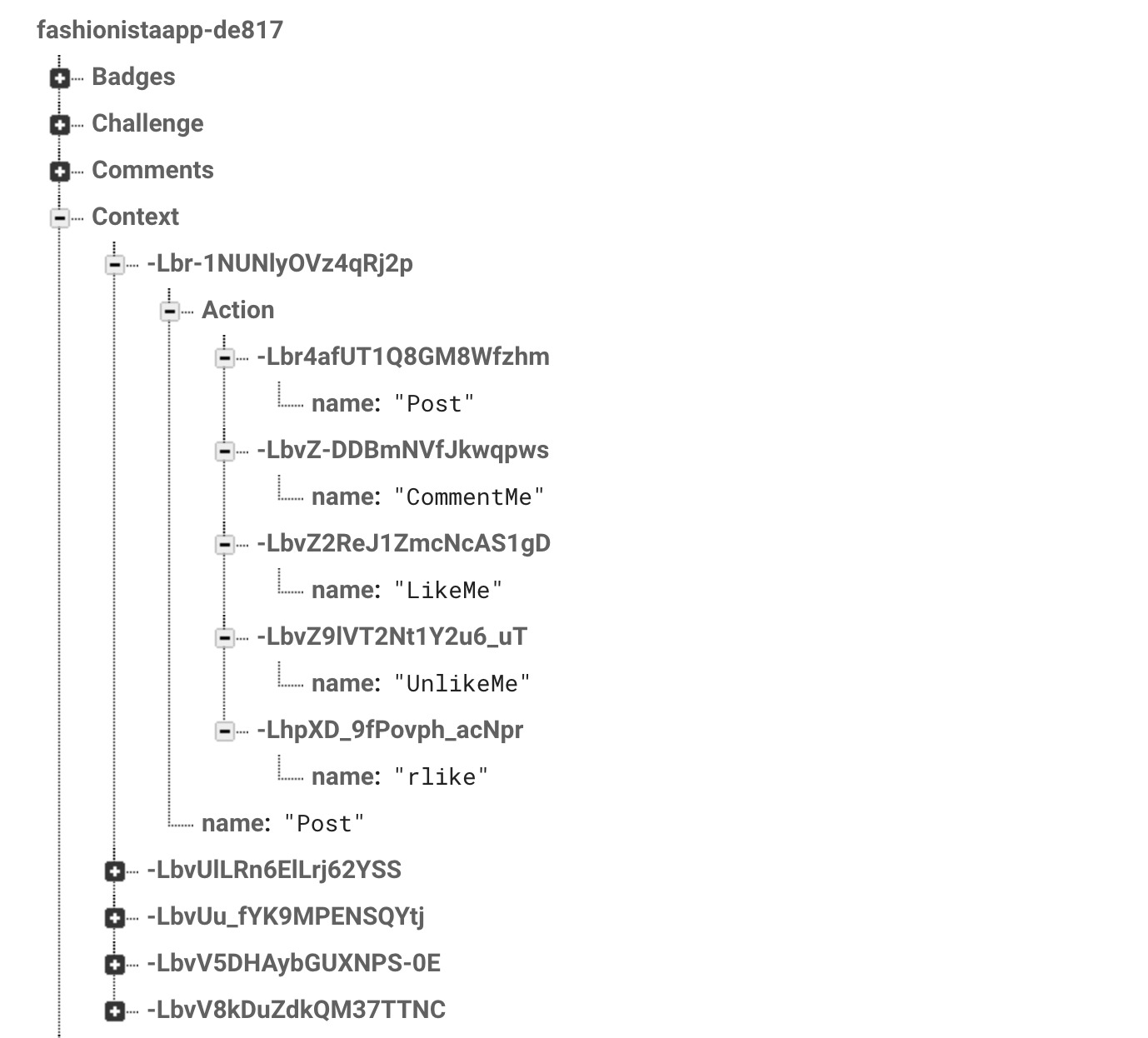
כאן נראה את המראה הכללי של המסד נתונים עם כל הטבלאות שנוצרו כתוצאה מהכנסת הנתונים של המתכנת לאתר כפי שרשומים למעלה.



כאן נוכל לראות את טבלת ה-Badges בתוך מסד הנתונים כאשר כל Badge מכיל את ה-ID הייחודי שלו ובלחיצה עליו ייפתחו הנתונים עבור אותו ה-Badge שכוללים: מספר ID של ה-Context עבור אותו ה-Badge, שם ה-Badge ותמונה.



כאן נוכל לראות את טבלת ה-Challenges בתוך מסד הנתונים כאשר כל Challenge מכיל את ה-ID הייחודי שלו ובלחיצה עליו ייפתחו הנתונים עבור אותו ה-Challenge שכוללים: ה-ID של ה-Badge שמקושר ל-Challenge זה, זמן התחלה, זמן סיום, שם של ה-Challenge, כמות נקודות שנזכה בהשלמה של אתגר זה וטבלת Action-ים שמכילה בתוכה את כל הפעולות שהמשתמש צריך לעשות בגדר האתגר הנוכחי שכוללים: ID של ה-Action וכמות הפעמים שצריך לבצע את הפעולה.



כאן נוכל לראות את טבלת ה-Context בתוך מסד הנתונים כאשר כל Context מכיל את ה-ID הייחודי שלו ובלחיצה עליו ייפתחו הנתונים עבור ה-Context הספציפי שכוללים: שם של ה-Context וטבלת ה-Action-ים שמציגה את כל הפעולות שניתן לעשות באותו ה-Context כך שנכללים הפרטים הבאים: ID הייחודי של ה-Action ובלחיצה עליו נראה את שם ה-Action.



כאן נוכל לראות את טבלת ה-Rp בתוך מסד הנתונים שמציגים את החוקיות עבור נתינת הנקודות כאשר כל חוק מכיל את ה-ID הייחודי שלו ובלחיצה עליו ייפתחו הנתונים עבור חוק ספציפי זה שכוללים: ID של ה-Action שמקושר לחוק הספציפי, כמות פעמים שעל המשתמש לבצע וסה"כ הנקודות שהמשתמש יקבל במידה ויבצע את הפעולה הספציפית כמספר הפעמים שרשומים.

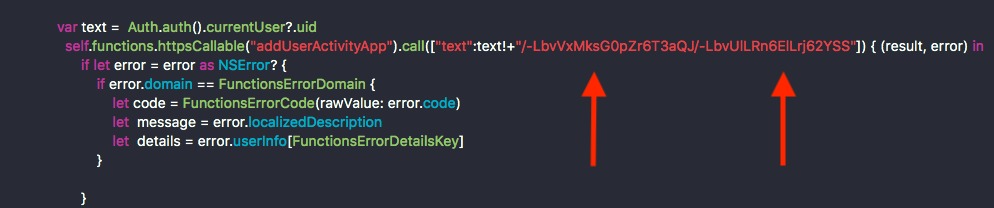


כאן נוכל לראות את טבלת ה-Rb בתוך מסד הנתונים שמציגים את החוקיות עבור נתינת תגים למשתמש כאשר כל חוק מכיל את ה-ID הייחודי שלו ובלחיצה עליו ייפתחו הנתונים עבור חוק ספציפי זה הכוללים: ID של ה-Badge המקושר לחוק הספציפי, ה-Context שבו החוקיות מתבצעת וסה"כ כמות הנקודות שעל המשתמש להגיע אליו על מנת להרוויח את התג.

**Application:**

זהו החלק של המפתח שבו הוא מקים אפליקציה ובה הוא רוצה להטמיע את נושא הגיימיפיקציה כלומר לחבר את התשתית אליו ולקבל את כל המידע הדרוש שמגיע מתוך מסד הנתונים בהתאם ל-API שמקושר לשרת.

בשביל לחבר את התשתית לאפליקציה על המפתח להכניס קוד קבוע לכל פעולה שהוא קבע באתר:



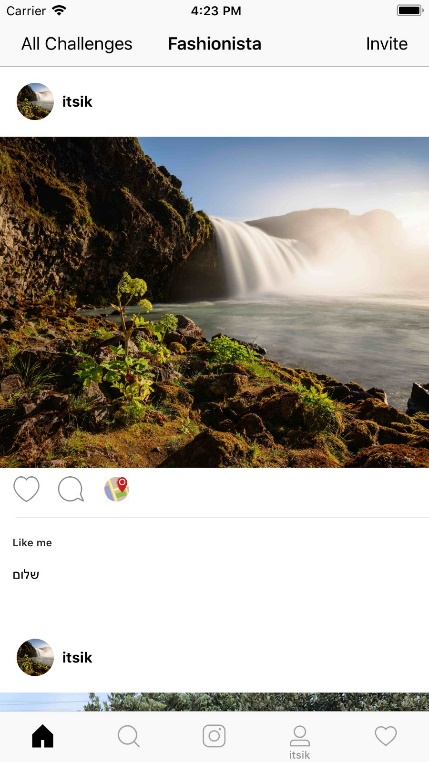
נשים לב שבתוך קוד זה – התבנית חוזרת על עצמה בכל פעולה שמתבצעת באפליקציה ולכן תהיה כתובה באופן קבוע אך ישנם 2 מקומות שבהם משתנים הנתונים וזה מה שמפריד את התבנית מפעולה לפעולה כפי שנראה בתמונה למעלה: נכניס את ה-ID של ה-Action בהתאם לפונקציה שבה אנחנו מכניסים תבנית זו (חץ שמאלי) ואז נכניס את ה-ID של ה-Context בהתאם ל-Action (חץ ימני). ככה נוכל בעת הפעלת פעולה זו באפליקציה לקרוא לפונקציה הנכונה בשרת.

לאחר שמתבצעים הפעולות של המשתמש באפליקציה נוכל לראות זאת בטבלת ה-UserActivity שבמסד הנתונים:

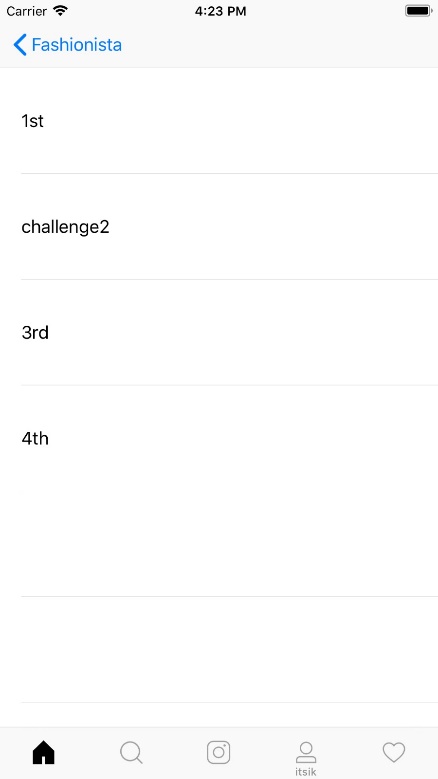


כאן נוכל לראות את טבלת ה-UserActivity של המשתמש הספציפי בתוך מבד הנתונים כאשר כל UserActivity מכיל את ה-ID הייחודי שלו ובלחיצה עליו ייפתחו הנתונים הבאים: ID של ה-Action אותו המשתמש ביצע, ID של ה-Context שבו המשתמש ביצע את הפעולה, תאריך ביצוע הפעולה, כמות הנקודות שהמשתמש קיבל עבור הפעולה ו-ID של המשתמש.

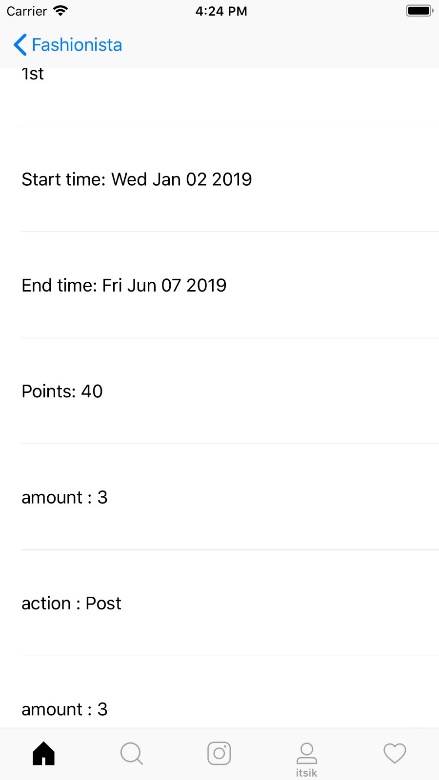
כעת נרצה לראות כיצד נראים הכפתורים שעל המפתח להוסיף בכדי לקבל את כל המידע של המשתמש בתוך האפליקציה:



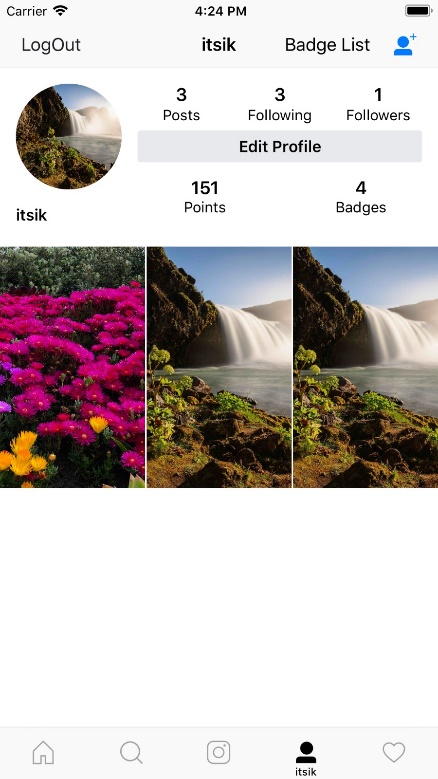
**כפתור "All Challenges"** – בלחיצה על כפתור זה נראה את כל האתגרים שקיימים באפליקציה



בלחיצה על כל אתגר – יפתח בפנינו כל המידע אודות האתגר הספציפי:



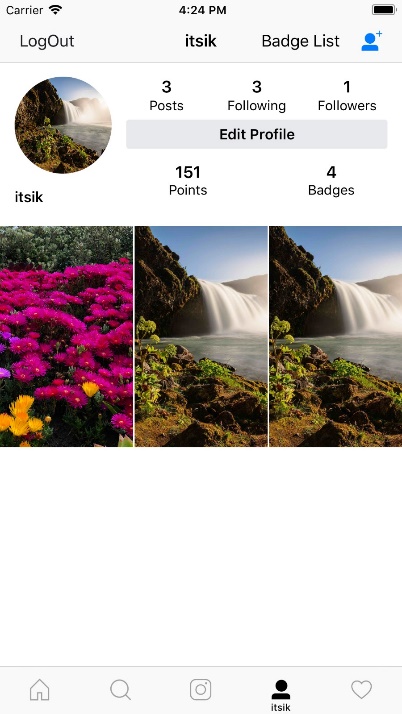
**שדות "Points" ו- "Badges" –** מציגים לנו את הנקודות והתגים שהמשתמש צבר.

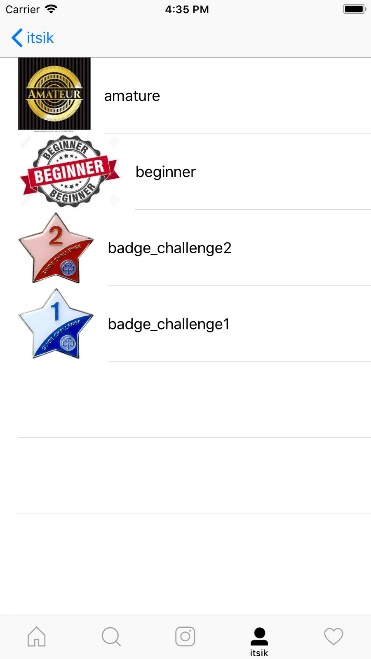


השדה "points" מקבל את ערכו מתוך מסד הנתונים תחת המידע אודות המשתמש וכמות הנקודות שצבר לאורך פעילותו באפליקציה.

השדה "Badges" מקבל את ערכו מתוך מסד הנתונים תחת המידע אודות המשתמש וכמות התגים שצבר לאורך פעילותו באפליקציה (מתבצע לפי ספירה של כמות התגים שנשמרת בטבלת "Badges" אצל המשתמש הספציפי).

* כלומר בתוך מסד הנתונים בטבלת משתמשים- יהיה לכל משתמש שדה נוסף שנקרא "points" והוא יתעדכן לאחר כל צבירה של נקודות מפעולות אותן ביצע.



****

**כפתור "Badge List"-** בלחיצה על הכפתור נראה את כל התגים אותם המשתמש הרוויח לאורך פעילותו באפליקציה:

תוצרי הפרויקט:

צד המפתח:

כאשר המפתח בנה אפליקציה, הוא נכנס לאתר ה-web שיצרנו ומכניס לשם את כל הנתונים עבור :

* Context – המיקום שבהם הפעולות מתבצעות
* Action – הפעולות שניתן לבצע
* Challenge - האתגרים
* Badge - התגים
* Rb – חוקים לקבלת תגים
* Rp – חוקים לקבלת נקודות

כל המידע שהוכנס באתר מוצג בפניו בטבלאות השונות וכמו כן מוקם אוטומטית במסד הנתונים.

כעת המפתח נכנס בקוד לכל פעולה שקיימת אצלו ומכניס תבנית קבועה שמותאמת עבור הפעולה הספציפית ובכך מקשר כל פעולה שכזו לשרת אותו בנינו.

צד המשתמש:

המשתמש מתחיל לבצע פעולות באפליקציה. כתוצאה מכך מופעל טריגר אשר קורא לפעולה הספציפית בשרת. כמו כן מתבצעת בדיקה מול הנתונים הקיימים במסד הנתונים ובמידת הצורך מתעדכן מידע אצל המשתמש גם במסד הנתונים (כמות פעמים לביצוע פעולה, זמנים ועוד..) וגם בעמוד האישי שלו בנקודות שצבר ובתגים שהרוויח.

הגיימיפיקציה שיצרנו יודעת להציג את התוכן הרלוונטי למשתמש.

אימנו את המערכת ולבסוף זו מזהה את הפעילות המשתמש באפליקציה ובכך לומדת את המשתמש ומעדכנת אותו בעת הצורך.

התוצר הסופי של הפרויקט הוא לבסוף **תשתית גיימיפיקציה** המוטמעת בשרת שיודעת לבצע את כל הפעולות הנדרשות בעת קריאתם מהאפליקציה ו- **אתר web** שיודע לקבל את הנתונים מהמפתח, להציג אותם באתר ולהכניס אותם למסד הנתונים.

ביבליוגרפיה:

* המאמר האקדמי שעליו התבססנו בהבנת הנושא

<https://www.researchgate.net/publication/260422760_Using_Gamification_in_an_Online_Community>

* <https://www.researchgate.net/publication/326021420_A_Gamification_Engine_Architecture_for_Enhancing_Behavioral_Change_Support_Systems>