

Finder's Keeper

מערכת חברתית לאיתור והשבת אבידות ומציאות מבוססת AWS

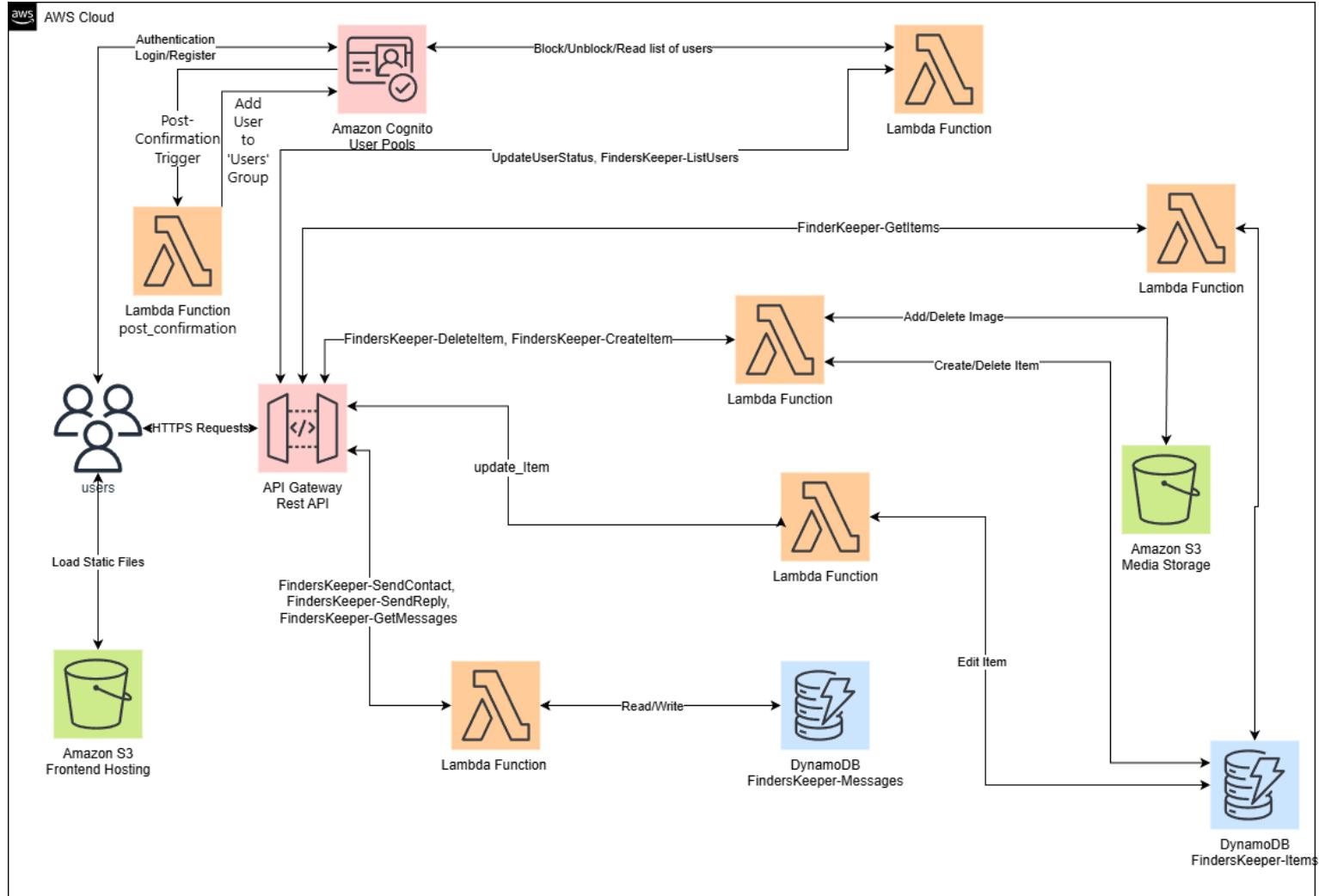
מג'יסטים:

אנס פרחאת : 212607345

מוחמד נור אבו עסבה : 315226472

רואה חשב : 213262801

1. סרטוט של כל הקיימות הארכיטקטוניות המעורבות בפרויקט והחיבורים ביניהם (Draw.io)



מצורף קובץ XML

2. הארכיטקטורה של הפרויקט

להלן הסבר מפורט על רכיבי המערכת והזרימה ביןיהם:

1: סקירת הארכיטקטורה מבוסס על `Finder.drawio`

AWS Lambda הינה אפליקציית אבדות ומציאות serverless שבנויות כולה על תשתית AWS. התרשים מציג הפרדה מלאה בין שכבות ה-frontend, API, compute ו אחסון נתונים.

זרימת המערכת מתחילה כמשמעותם ניגשים לתוכנים סטטיים מ-S3-מתחררים דרך Cognito, AWS Lambda ו API Gateway. הבקשות מנותבות לפונקציות DynamoDB שמצוות Logika Usability ו מתקשרות עם שני טבלאות FindersKeeper-Messages (Items) ו S3-FindersKeeper-Items לאחסן תמונות.

2: קשרים בין רכיבים וזרימת נתונים

משתמשים S3 Frontend Hosting נכנסים לכתובת האתר, הדף שולח בקשה HTTPS://S3 שמחזיר קבצי HTML כגון index.html, postItem.html, profile.html, AWS SDK JavaScript ו CSS. קבצי adminPage.html, adminPage.css ו CSS. AWS SDK כולל תצורת JavaScript. AWS SDK מספק קבצים אלה בהשניה נמוכה ללא צורך בתחזוקת שירותי. הקבצים הסטטיים מכילים את הלוגיקה של הלקוח בהתאם לקריאות API לשירות backend.

משתמשים Amazon Cognito → אימות והתחברות/הרשמה (אימות משתמשים עובר דרך Cognito User Pools באופן עצמאי. בהרשמה, משתמשים שולחים מייל, סיסמה ושם מלא יישורות ל- AWS SDK. AWS SDK שולח קוד אימות למייל שהוזן. לאחר שהמשתמש מאשר את המייל על ידי הזנת הקוד, החשבון הופך לפעיל.

בהתחרבות Cognito, מאמת את פרטי הכניסה ומוחזק שלושה JWT tokens: token-ID המכיל את פרטי המשתמש (user ID, email, name וכו'), token-access שולח לאחר תפוגה. token-refresh-Shomer את token-ID ו יכול אותו בcoturrtת Authorization של כל בקשה ה-API. אימות מבוסס token-Zה אפשר לAPI-Gateway ולפונקציות Lambda לשזהות את המשתמש ולא לארוף הרשות sessions בצד השרת.

3: ניתוב בקשות בעקבות API Gateway

API Gateway משמש כניסה יחידה לכל פעולה ה-backend, מקבל בקשה HTTPS מהמשתמשים ומנתב אותה לפונקציות Lambda ספציפיות לפי ה-HTTP-endpoint והשיטה. התרשים מציג שמות נטבי Lambda שונים:

נקודות קצה ציבוריות: • FindersKeeper-GetItems: טיפול בעיון בפריטים אבודים ונמצאים ללא אימות.

ניהול פריטים מאומת: • FindersKeeper-CreateItem: יצירת פרט חדש (דרש JWT token). • FindersKeeper-DeleteItem: removal (דרש אימות בעל או ID). • FindersKeeper-UpdateItem: Update Item (דרש אימות בעל).

מערכת הודעות: • FindersKeeper-SendContact: שליחת הודעות ראשונית לבני פריטים. • FindersKeeper-SendReply: FindersKeeper-GetMessages: FindersKeeper-SendReply: אחזור תיבת דואר נכנס של המשתמש.

פעולות אדמין: • FindersKeeper-ListUsers: רישימת כל המשתמשים הרשומים (אדמין בלבד) .
UpdateUserStatus: חסימה או ביטול חסימה של חשבון משתמש (אדמין בלבד).

עבור נקודות קצה מאומתות API Gateway, משתמש ב-Cognito authorizer, שמאמת את JWT-הTOKEN קרייה לפונקציה Lambda. אם ה-TOKEN לא תקין, פג תוקפו או חסר backend, מוחזר תגובה Unauthorized 401 מיד ללא הפעלת קוד backend כאשר ה-TOKEN חסר.

מחלוקת בין claims מה JWT-ומעביר אותו לLambda-Gateway בהקשר הבקשה.

4: פעולות פונקציית Lambda

פונקציות ניהול פריטים

DynamoDB FindersKeeper-GetItems מבצע שאלתה לatable FindersKeeper-items כדי לאחזר פריטים אבודים ונמצאים. הפונקציה תומכת בסינון לפי סטטוס (אבוד/נמצא), קטגוריה, מקום ותאריך באמצעות GSI StatusIndex לשינון יעיל תוך שמירה על סדר כורנולוגי. התוצאות מוחזרות כערך JSON שהפונקציה מציג ככרטיסי פריטים.

FindersKeeper-CREATEitem מבצע פריט נתוני (כותרת, תיאור, סטטוס, קטgorיה, מקום, תאריך, צבע) יחד עם תמונה מקודדת base64 מה frontend. הפונקציה מחלצת את מידע המשתמש המאומת (user ID, email, name) מה claims מה API Gateway לאחר מכן מפענחת את נתוני התמונה מ-base64, מייצרת שם קובץ UUID ייחודי, ומעלת את התמונה הבינארית לדלי S3 Media Storage boto3.put_object של boto3. דלי S3 מוחזר URL ציבורי HTTPS לתמונה שהועלתה. לבסוף, הפונקציה יוצרת רשומת פריט חדשה בטבלה FindersKeeper-items המכילה את כל פרטי הפריט, מידע המשתמש URL, התמונה וחומרת זמן של יזרה.

FindersKeeper-DeleteItem מבצעת אימות בעלות לפני מחיקה. היא מאחררת את הפריט מ-DynamoDB-ID-שסוף, ואז משווה את שדה user_id של הפריט מול ה-ID-של המשתמש המאומת מה claims-Cognito. אם המשתמש אינו הבעלים, הפונקציה בודקת אם הוא שיר לקבוצת "Admins" על ידי בדיקת ה claim cognito:groups. רק בעלי פריטים או מנהלים יכולים להמשיך במחיקה. אם מאושר, הפונקציה מוחקת את רשומת הפריט מ-DynamoDB-ובאופן אופציוני מסира את התמונה המשויכת מ-S3.

update_item מעדכנת את השדות צוינו ברשות הפריט ב-DynamoDB. admin, היא מעדכנת את הרשות, אחר אותה לוגיקה של אימות בעלות. לאחר אישור שהמשתמש הוא הבעלים או בעל הרשות, מוחקת רשומת/pritem.

פונקציות מערכת ההודעות

SendContact מבצע קשר למשתמשים באמצעות שרורי שיחה עבור המשתמש המאומת. היא מבצעת כשר משתמש שלוח הודעה על פריט, הפונקציה תחילת מאחררת את הפריט מטבלת FindersKeeper-items כדי לזהות את ID user והמייל של הבעלים. לאחר מכן היא מזودדת שהשלוח לא מנסה ליצור קשר עצמו על ידי השוואת ID user של השלוח (user ID-Cognito claims) מ-shield (user ID-Cognito claims). לאחר מכן יוצרת רשומת הודעה חדשה בטבלה FindersKeeper-messages של בעל הפריט. אם האימות עבר, הפונקציה יוצרת רשומת הודעה חדשה בטבלה FindersKeeper-messages. מכילה את item ID, מידע שלוח, מידע גמען, טקסט הודעה, דגל סטטוס קרייה (false) בתחילת (וחומרת זמן יזרה).

GetMessages מבצעת רשורי שיחה עבור המשתמש המאומת. היא מבצעת שתי שאלות: אחת באמצעות GSI RecipientIndex למציאת הודעות שבנה המשתמש הוא הנמען, וסיקת טבלה למציאת הודעות שבנה המשתמש הוא השולח. הפונקציה מקבצת הודעות אלה לשדרורי שיחה לפי השילוב של item ID וזוג משתפים, מחשבת ספירת הודעות שלא נקרו לכל שדרור, ומוחזירה את הנתונים המקובצים כתגובה JSON.

FindersKeeper-SendReply מטפלת בתגובה לשדרורי הודעות קיימים. היא ממattaת שהשלוח הוא משתמש בשיחה (השלוח או הנמען המקוריים) לפניה יצירת רשומת הודעה חדשה המוקשחת לאותו פריט ומשתתפים.

fonkzot adamim

FindersKeeper-ListUsers משרות את לוח הבקרה של האדמין על ידי אחזר כל המשתמשים מהפונקציה 초기ה תחיליה מודדת שהמשתמש המבקש שיר לקבוצת "Admins" על ידי בדיקת ה-Cognito.

403 forbidden token. claim cognito:groups עבור בקשות מורשות, הפונקציה קוראת ל Cognito API-ל-Sh list_users כדי לאחזר את כל המשתמשים הרשומים עם התכונות שלהם (מייל, שם, סטטוס חשבון, מצב, enabled/disabled) תאrik יירה (ומחזירה נתונים אלה כמערך JSON).

UpdateUserStatus מאפשרת למנהל לחסום או לבטל חסימה של חשבון משתמש. לאחר אימות הרשות אדמין, הפונקציה מקבלת שם משתמש ופרמטר פעולה ("block" או "unblock") לחסימה, היא קוראת ל admin_disable_user API-Sh של Cognito, שמנע מיד מהמשתמש לקבל Sessions tokens חדשים עד שה JWT-פג תוקפו, אבל המשתמש לא יכול להתחבר שוב. לביטול חסימה, היא קוראת ל user-admin_enable_user כדי לשחרר גישת חשבון.

5: ארכיטקטורת אחסון נתונים

DynamoDB Tables

FindersKeeper-Items מאחסנת את כל פריטים האבודים והנמצאים עם מבנה מפתח פשוט באמצעות GUID כמפתח partition. כל רשומה מכילה את פרט הערך (כתובת, תיאור, סטטוס, קטgoria, מקום, תאריך, צבע), מייל יוצר (userId, userName, userEmail, userIdn) שהועתק מ-S3 (תמונה)boolean resolved S3 דגל (מצין אם הפרט הוחזר/nמצא, וחوتמת זמן יירה. הטבלה כוללת Index Global Secondary Index בשם StatusIndex עם מפתח partition-createdAt ו-keysort, המאפשר שאלות ייעילות של "הציג את כל הפריטים האבודים" או "הציג את כל הפריטים שנמצאו" ממוינים כרונולוגית.

FindersKeeper-Messages מתחזקת שיחות פרטיות בין משתמשים על פריטים ספציפיים. כל רשומת הודעה מקשרת ל itemId-recipientIndex GSI של user IDs ומיילים של שלוח ומען, טקסט ההודעה, דגל booleanRecipientKey וחותמת זמן יירה. ה-Index משמש ב-userId-keypartition, המאפשר שאלות ייעילות של "הציג את כל ההודעות שנשלחו למשתמש זה" ללא סריקת הטבלה כולה.

DLL3S

הארQUITktורה משתמשת בשני DLL3S נפרדים עם מדיניות גישה שונות. DLL Hosting Frontend כולל קבצי HTML, CSS, JavaScript ו-Media API-Gateway.dll-לו יש אירוח אתר סטטי מופעל ו-CORS-RecipientIndex GSI-RecipientIndexdll-לו יש תמונות פריטים שהועלו עם גישת קרייה ציבורית ל-CloudFront.

DLL Media Storage מאחסן תמונות פריטים שהועלו עם גישת קרייה ציבורית אך מגביל הרשות כתיבה לתפקיד הרצה של Lambda בלבד. תמונות מאוחסנות עם שמות קבצים GUIDים למניעת קונפליקטים ולהבטחת ייחודיות.

6: מודל אבטחה והרשות

המערכת מיישמת אבטחה רב-שכבותית דרך קבוצות משתמשים של API-Gateway Cognito, authorizers Lambda. משתמשים שייכים לקבוצת משתמשים ברירת מחדל או לקבוצת "Admins". חברות באדמין מאוחסנת ב Cognito-claim cognito:groups ומקודדת ב JWT tokens.

API מאמנת את כל הtokens AWS-בנוקודת הכניסה לפני קרייה לפונקציות Lambda, מונע בקשות לא מושות מלצורך משאבי backend. נקודות קצה ציבוריות כמו-*FindersKeeper*, *GetItems* ווקפות בדיקה זו כדי לאפשר עיון ללא אימוט. עברו נקודות קצה מאומנות API, מחלץ claims משתמש מ-tokens ומעבר אותם ל-Lambda-*Gateway* שהקשר הבקשה.

פונקציות Lambda מבצעות הרשאה נוספת עבור פעולות מבוססות בעלות. כאשר משתמש מסזה למחוק או לעורוך פריט, הפונקציה משווה את שדה *userId* של הפריט מול ID של המשתמש המאומנת מ-*Cognito claims*. אם לא תואמים, הפונקציה בודקת חברות בקבוצת "Admins" ב-*claim*-groups. יוצריםCRC בעלים או מנהלים יכולם להמשיך בפעולה.

טבלאות DynamoDB אין נגישות לציבור; רק פונקציות Lambda עם תפקיד IAM מתאימים יכולות לבצע פעולות קרייה וכ כתיבה. דלי S3 משתמשים במדיניות נפרדות: דלי ה-frontend-מאפשר קרייה כיבורית לקבצי אחר, בעוד שDAL התמנות מאפשר קרייה כיבורית אך מגביל העליות לתפקיד הרצה של Lambda. כל התקשרות מתרחשת על HTTPS מצפינה נתונים במהלך, או-*Cognito* מנהל hashing.

7: יתרונות ארכיטקטוניים

ארQUITECTורת serverless זו מבטלת תחזוקת שרתים תוך scaling אוטומטי מ敞开 לאלפי משתמשים במקביל AWS. מטפלת בתasking-provisioning, תיקונים, עדכוני אבטחה ותוכנו קיבולת-עלויות משתנות ליניארית עם השימוש בפועל דרך pay-per-request עבור קריאות Lambda, בבקשתות API Gateway ופעולות קרייה/כתיבה ב-DynamoDB.

יעצב המיקרו-שירותים מבודד כשלים ומפשט תחזקה. כל פונקציית Lambda יכולה להתעדכן באופן עצמאי ללא השפעה על רכיבים אחרים API Gateway. מספק משקל REST יציב בין frontend ל-backend, שינה-אפשר backend, לא צריך(frontend) שינויים. ההפרדה של אוסף תוכן סטטי APIs (S3) מ-*DynamoDB* (Lambda + scaling | caching) מאפשרת אסטרטגיות עצמאיות לכל tier.

3. שרטוטים והספרט של תכנון מוצר המשמש(**UI**) מצורף בקובץ.

4. רשימת ה **Features** המלאה שלuproיקט (**Use Cases**) מצורף בקובץ.

5. קישור ל-**AWS PRICING CALCULATOR**:

<https://calculator.aws/#/estimate?id=8d8d76ddc94e6dfc6bd74638a18cfcd82ab727>

9e

(קובץ ה-PDF נמצא בקובץ ה- ZIP)

6. הנחות יסוד לחישוב עלויות(Cost Calculation Assumptions)

תיחסיב ה

עלויות עבור פרויקט Finder's Keeper מtabסס על הרכבת שימוש עבור סביבת דמו/פיילוט הראשוני. הארכיטקטורה נבנתה במודל Serverless (לא שרתים) המאפשר תשלום לפי שימוש בפועל, (Pay-As-You-Go) מה שמבטיח עלויות אפסיות כאשר המערכת במצבה או בשימוש נמוך.

להלן פירוט הנחות לכל שירות:

1. משתמשים ונפח פעילות:(Users & Traffic)

- כמות משתמשים(MAU): התיחסיב מניח בסיס של **100 משתמשים פעילים בחודש** (Monthly Active Users).
- דף שימוש:** כל משתמש מבצע ממוצע 3-4 כניות למרכז בחודש, ובוצע פעולות כגון חיפוש פריטים, צפיה בפריטים ודיוח על אבידה/מציאה.

2. עיבוד ותעבורה:(API Gateway & Lambda)

- כמות בקשות(Requests) אנו מניחים כ**11,000-בקשות HTTP בחודש**.
- חישוב משוער: 100 משתמשים * 110 בקשות למשתמש (כולל טעינת דפים, סינונים, העלתה פריטים וקריאות רקע).
- משאבי עיבוד(Compute):** כל ריצת פונקציית Lambda מוערכת ב**500ms**-(חצי שנייה) בפועל, עם הקצאת זיכרון של **128MB**. נתוניים אלו נופלים ברובן המוחלט תחת ה-AWS Free Tier.

3. אחסון קבצים ותמונות(S3)

- נפח אחסון: הוקצה **1GB** עבור אחסון סטטי של קבצי האתר(Frontend) ותמונות של פריטים שאבדו/נמצאו.
- תעבורת רשת(Data Transfer): מוערכת ב**1GB**-תעבורה יוצאה(Outbound) בחודש, בהנחה שרוב התמונות עוברות אופטימיזציה לפני הציגה.
- פעולות(Requests): כ-200 פעולות כתיבה (העלאת תמונות חדשות) וכ-2,000 פעולות קריאה (צפיה בפריטים) בחודש.

4. מסד נתונים(DynamoDB)

- מודל תמחור:** נבחר מודל On-Demand למינית תשלום על קיבולת לא מנוצלת (idle capacity).
- נפח נתונים: הוקצה **1GB** לשמירה המידע הטקסטואלי(Metadata) של הפריטים, המשתמשים וההודעות. מכיוון שפריט ממוצע שוקל כ-4KB בלבד, נפח זה מספיק לשמירת כ-250,000 רשומות, הרבה מעבר לצרכי הפילוט.

5. אימות(Cognito)

- אימות משתמשים:** עלויות Amazon Cognito עבור 100 משתמשים הן \$0 (במסגרת ה-Free Tier המאפשר עד 50,000 משתמשים).

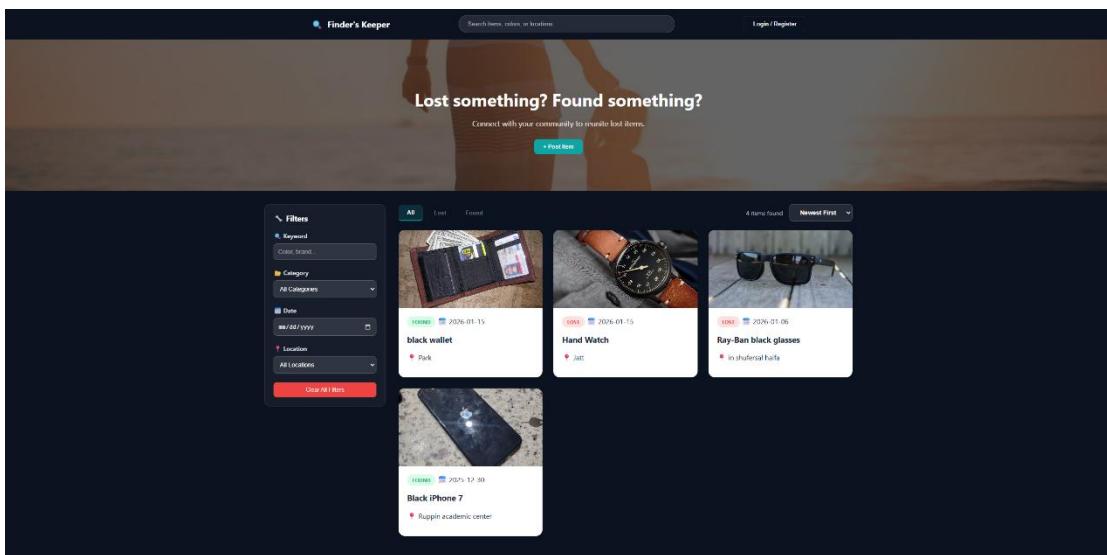
סיכום: מתחת הנחות יסוד אלו, העלות החודשית המוערכת עומדת על פחות מ \$0.50-מה שמודגים את היעילות הכלכלית של ארכיטקטורת ה-Serverless-עבור הפרויקט.

7. מדריך למשתמש - מערכת **Finder's Keeper**

Finder's Keeper, מערכת חברתית לאיתור והשבת אבידות ומציאות. מדריך זה יספק הוראות לשימוש במערכת בקלות וביעילות.

כניסה למערכת:

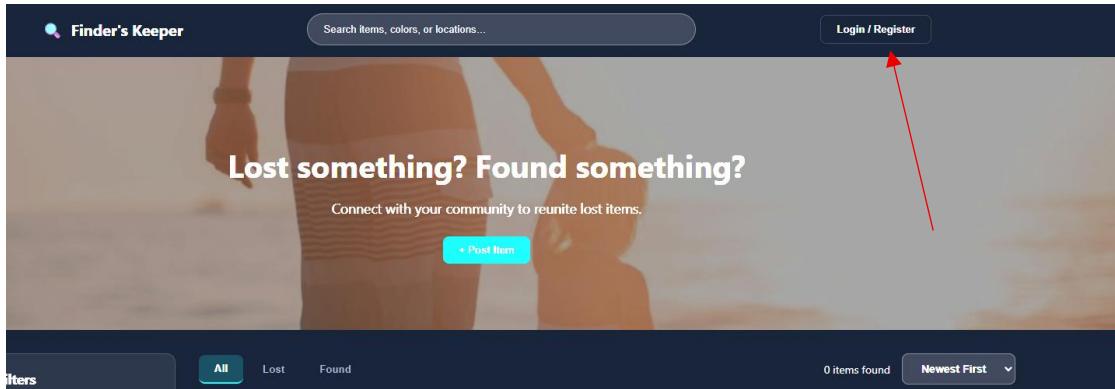
1. דף הבית ולוח המודעות: (The Feed) הדף הראשון שמופיע לכל משתמש בכניסה לאתר הוא הדף (לוח המודעות הראשי). דף זה פתוח לצפייה לכל המבקרים ("מצב אורח") ללא צורך בהרשמה מוקדמת.



• מבנה הדף:

- **סריג עליון:** מכיל שורת חיפוש ("Search items, colors, or locations") וכפתור כניסה/הרשמה ("Login / Register").
- **אזור ראשי (Hero Section):** כולל כוורת מזמין וכפתור בולט "+ Post Item" לדיווח מהיר על אבידה או מציאה.
- **סריג צד (Filters):** מאפשר סינון מתקדם של התוצאות לפי מילוט מפתח, קטגוריה, תאריך (Category), ומקום (Location). קיים כפתור All Clear All Filters לאיפוס הסינון.
- **רשימת הפריטים:** תצוגת כרטיסיות של כל הפריטים שדווחו. כל כרטיסייה מציגה תמונה, סטטוס (Lost/Found), תאריך, שם הפריט ומקום. ניתן למין את התוצאות למשל ("All, Lost, Found" / "Newest First") ולסנן לפי סוג הדיווח (Lost, Found).
- **הגבלות פעולה לאורחים:** משתמשים שאינם מחוברים יכולים לצפות במידע, אך לחיצה על כפתורים הדורשים זיהוי – כגון "+ Post Item" – י帶duce הבית או יצירת קשר עם מפרסם – תעביר אותו אוטומטית למסך ההתחברות/הרשמה.

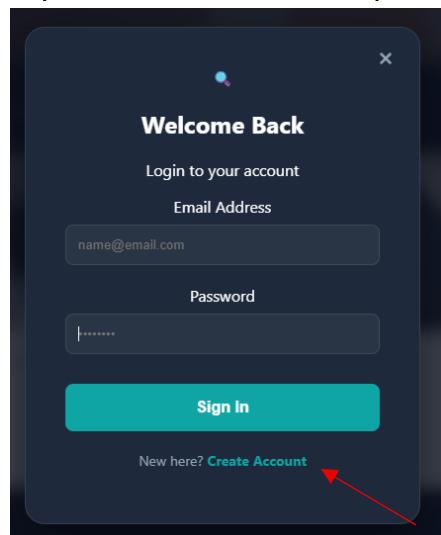
2. התחברות, הרשמה ואיומות: (Sign In & Verification): כדי לבצע פעולות אקטיביות במערכת, המשתמש נדרש להתחבר:



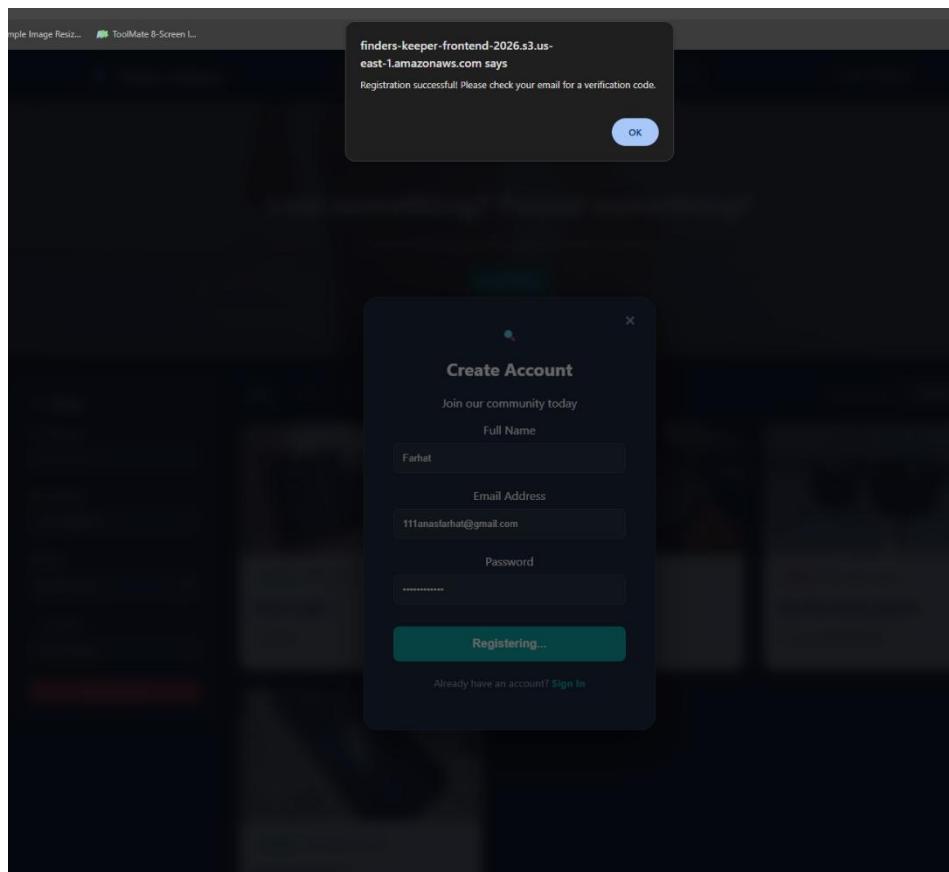
- **התחברות: (Sign In)** בחלון שנפתח, יש להזין את כתובת המail והסיסמה וללחוץ על ."Sign In"

- **הרשמה ואיומות: (Create Account)** משתמשים חדשים חייבים לאמת את כתובת המail שלהם:

1. יש ללחוץ על "Create Account" בחלון ההתחברות.



2. יש למלא שם מלא, אימייל וסיסמה וללחוץ על "Register".

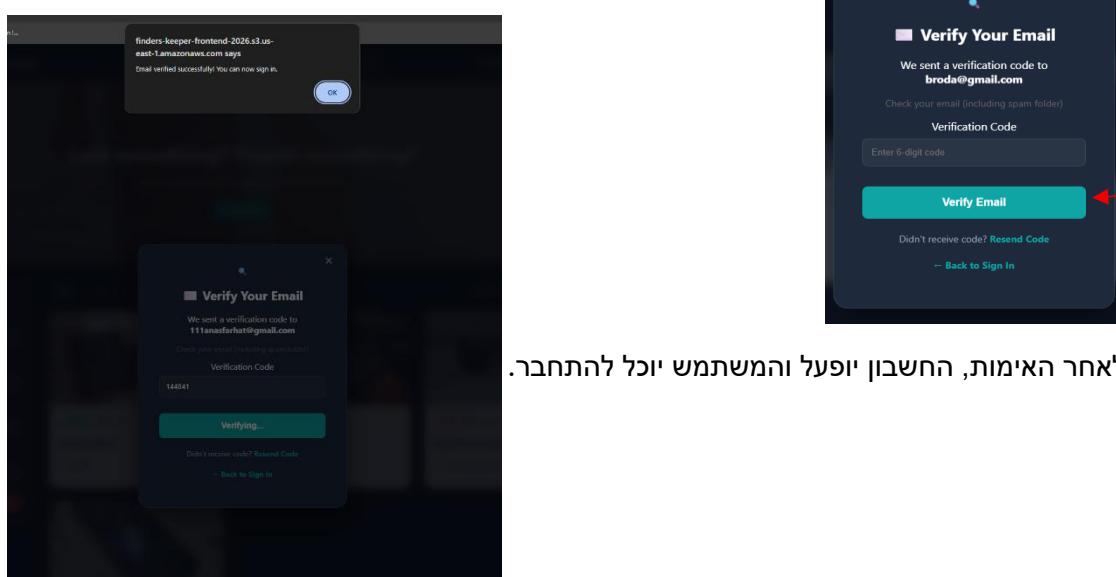


3. שלב האimotoת: המערכת תשלח קוד אimotoת למייל ותציג הודעה : "Please check your email for a verification code".

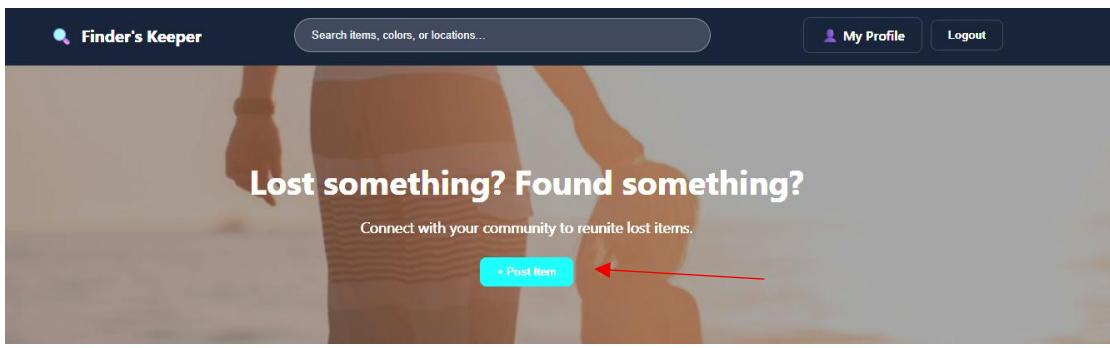
4. יש להעתיק את הקוד מתיבת המail (לדוגמא: 144041).



5. יש להזין את הקוד בחלון האimotoת באתר וללחוץ על "Verify Email".



6. לאחר האimotoת, החשבון יופעל והמשתמש יוכל להתחבר.



3. פרסום פריט (Post an Item): פועלה זו זמינה למשתמשים מחוברים בלבד (משתמש המונין) לבדוק על אבידה או מציאה יכול לעשות זאת בקלות באמצעות טופס ייעודי. לחיצה על **"+ Post"** תפתח את המספר הבא:

העלאת תמונה (Image Upload) בצד שמאל של המספר ישנה מסגרת ייעודית.

- לחץ על **"Select Photo"** כדי להעלות תמונה ברורה של הפריט. תמונה איקונית מגדילה משמעותית את הסיכוי לזהותו.
- **מילוי פרטי הפריט:**
 - .1. **Status:** בחלק העליון, בחר את סוג הדיווח:
 - לחץ על **"Lost"** (cptor אדום) אם איבדת פריט.
 - לחץ על **"Found"** (cptor אפור כהה שיופיע לפועל) אם מצאת פריט.
 - .2. **Item Title**: הacen כותרת קצרה ומתאatta למשל ("Blue iPhone 13").

.3 **Category**: בחר את הקטגוריה המתאימה ביותר מהרשימה הנפתחת (למשל: ("Pets", "Electronics")

.4 **Date**: בחר בלוח השנה את התאריך בו הפריט אבד או נמצא.

.5 **Location**: הגדיר את המיקום המדוייק או האזור הכללי (למשל: "Ruppin Academic Center").

.6 **Description**: בשדה זה ניתן להוסיף פרטים נוספים שיעזרו לזהות, כגון סימנים מיוחדים, צבע, מצב הפריט ועוד.

- **סיום ופרסום**: לאחר מלאי כל השדות והעלאת התמונה, לחץ על הכפתור הירוק הגדול בתחתית הטופס "**Publish Item**" המודעה תפורסם מיד בפיד הראשי ותהייה גלויה לכל המשתמשים.

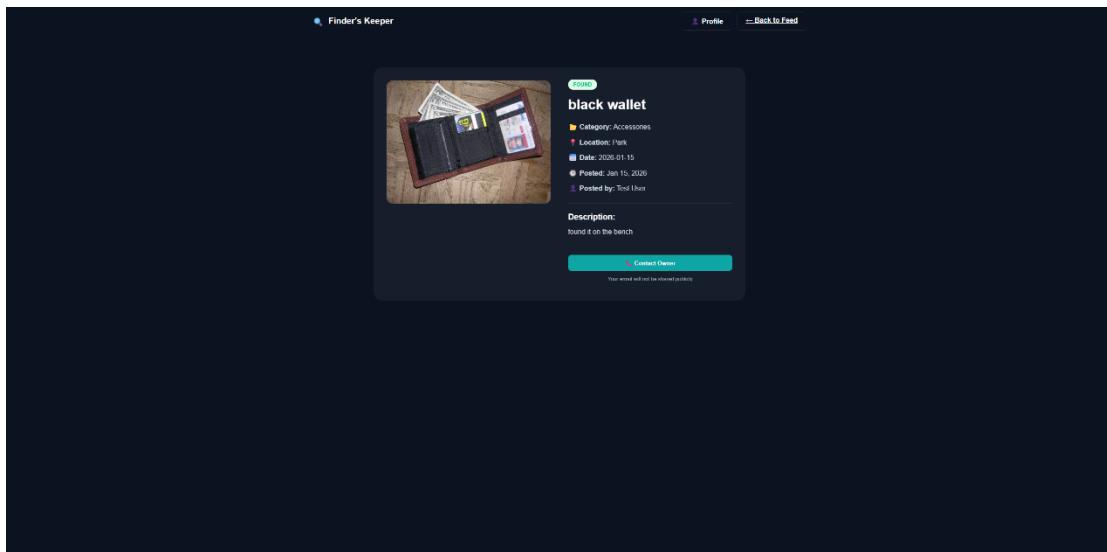
The screenshot shows the 'Post an Item' form on the Finder's Keeper app. The form is titled 'Post an Item' and includes a placeholder 'Fill in the details to help the community.' At the top right are 'Profile' and 'Back to Feed' buttons. The form fields are as follows:

- Status**: A dropdown menu with 'Lost' (highlighted in red) and 'Found' options.
- Item Title**: A text input field containing 'black glasses'.
- Category**: A dropdown menu set to 'Accessories'.
- Date**: A date picker showing '01/02/2026'.
- Location**: A text input field containing 'Bus station Haifa'.
- Description**: A text area containing 'Black glasses from Boss company.'

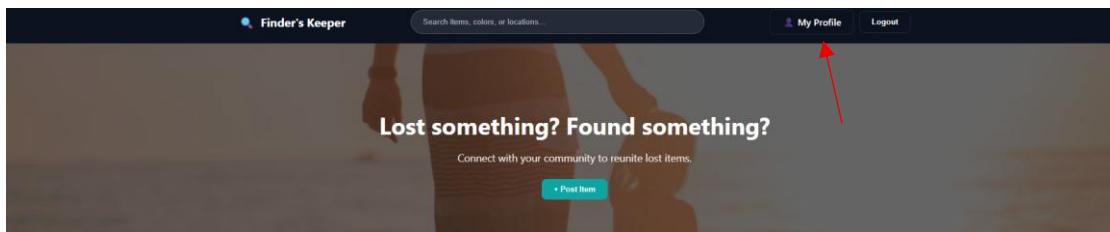
A large blue 'Publish Item' button is at the bottom. A red arrow points to this button from the bottom right of the image.

4. צפיה בפריטים ופרטי פריט: משתמשים יכולים לצפות בפריטים שפורסמו על ידי אחרים. לחיצה על פריט תפתח את מסך הפרטים המלא:

- **תצוגה ויזואלית:** תמונהגדולה של הפריט.
- **פרטים:** קטגוריה, מיקום, תאריך האירוע, תאריך הפרסום ושם המפרסם.
- **Description:** פירוט הטקסט שכתב המפרסם למשל ("found it on the bench").
- **יצירת קשר:** לחיצה על כפתור "Contact Owner" תפתח צ'אט ישיר עם מפרסם המודעה לתיאום החזרה.



5. ניהול הפריטים שלי (My Posts): באזורי האישית (My Profile) המשתמש יכול לראות סטטיסטיקה וניהול של המודעות שלו:



- **סטטיסטיקה:** כרטיסיות המציגות כמה "Lost Items" וכמה "Found Items".

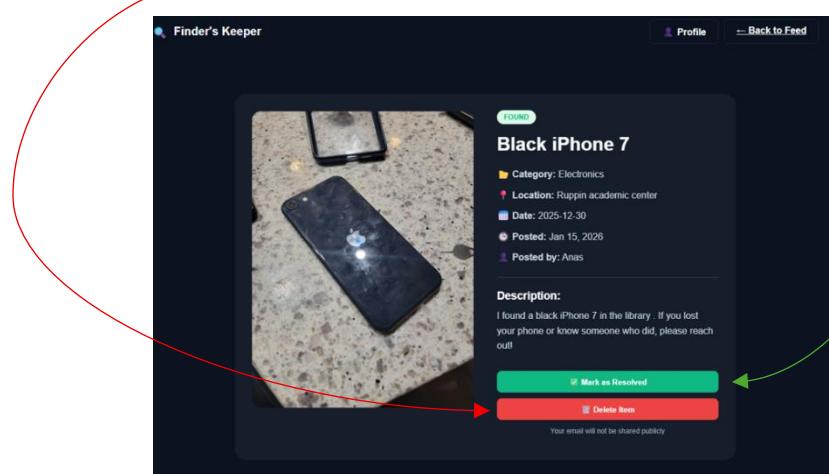
The screenshot shows the 'My Posts' section of the app. At the top, there is a profile card for 'Anas' with a green circle containing the letter 'A'. Below the profile card, there are three summary boxes: '2 Total Posts', '1 Lost Items', and '1 Found Items'. The 'My Posts' section lists two lost items and one found item:

- Lost Items:**
 - A black iPhone 7 posted on 2025-12-31.
 - A pair of sunglasses posted on 2026-01-01.
- Found Item:**
 - A black iPhone 7 posted on 2025-12-31 by 'ddddd'.

• **פעולות על מודעה:**

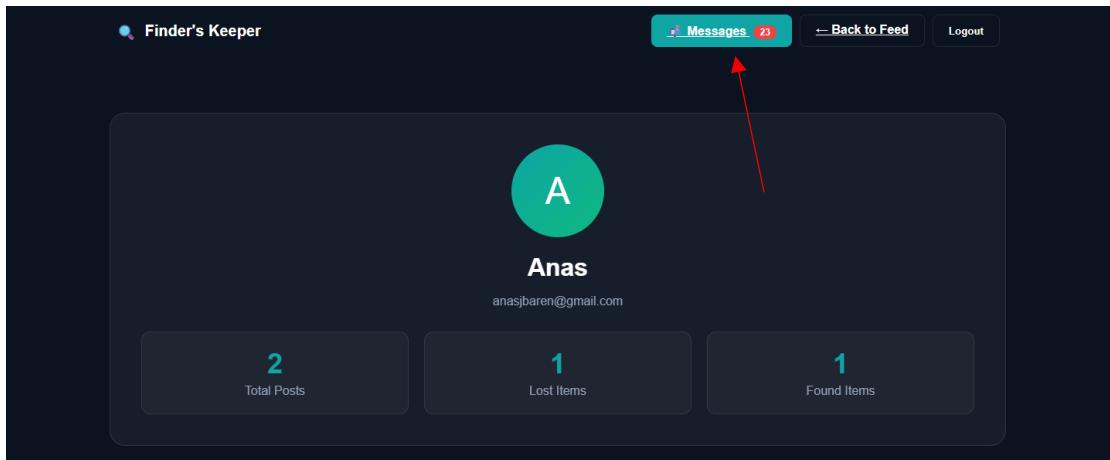
אם הפריט הוחזר לבעליו, המשתמש יכול לסמן אותו כפתרו (ירוק).

מחיקת המודעה מהמערכת (אדום).

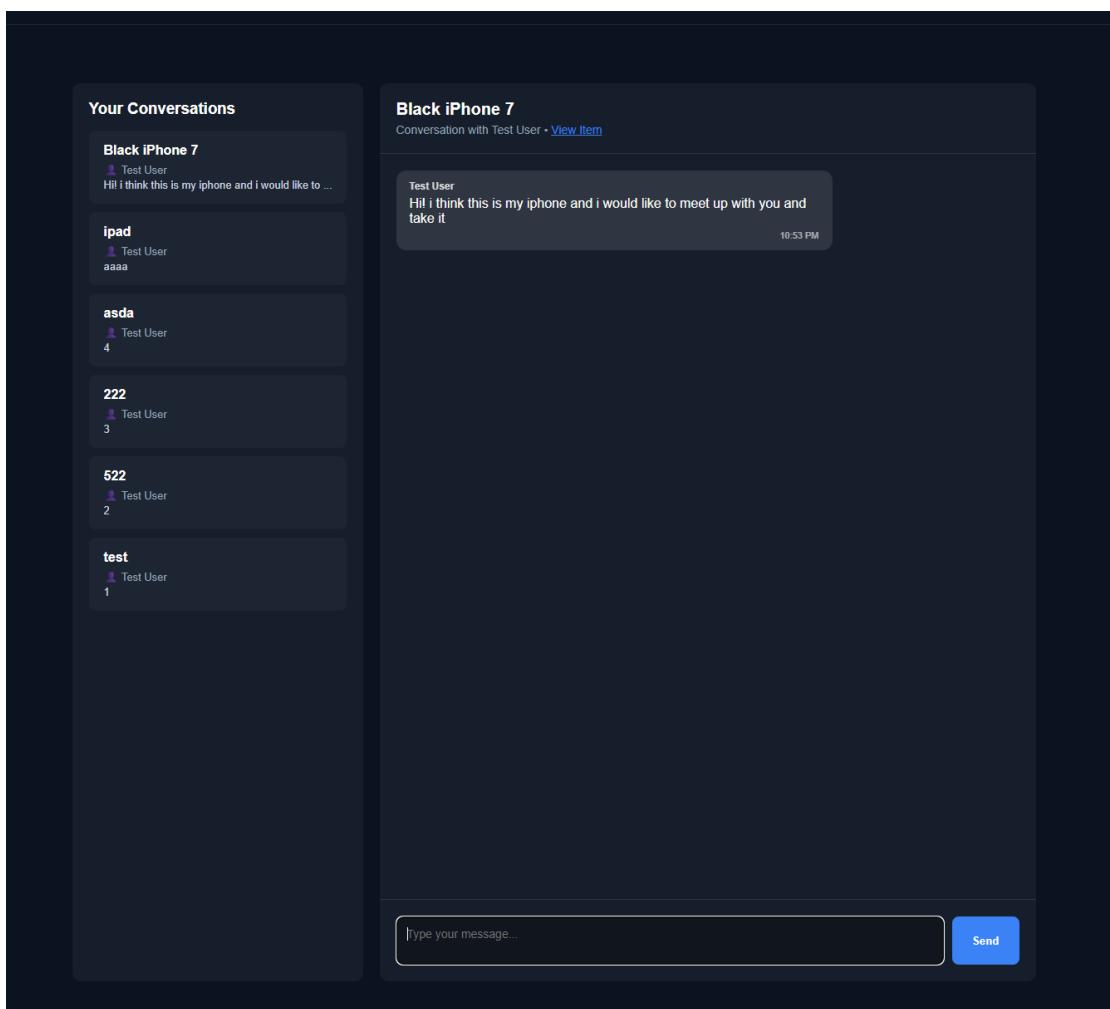


6. מערכת הודעות (Messages) המערכת מאפשרת תקשורת פרטית בין המוצא למאבד.

- לחיצה על כפתור ה-Messages בסריגת הניווט העליון תוביל למסך השיחות.



- רשימת שיחות: בצד שמאל מוצגות כל השיחות הפעילות לפני שם הפריט.
- חלון הציג: בצד ימין מתבצעת ההתכתבות בזמן אמת.



8. מדריך למנהל המערכת (Admin Manual)

למנהל המערכת (Admin) יש גישה לממשק ניהול ייעודי ("Control Center") המאפשר בקרה על התוכן והמשתמשים באפליקציה.

- **כניסה לממשק ניהול:** לאחר התחברות עם משתמש בעל הרשותות מנהל, תופיע גישה ל Control Center.

(1) **לוח מחוונים (Dashboard):** בחלק העליון מוצגים נתונים בזמן אמת על פעילות המערכת:

- **Total Items:** סה"כ הפריטים שפורסמו במערכת.
- **Active Users:** מספר המשתמשים הפעילים הרשומים (משתמשים שעברו אימוט).
- **Resolved Cases:** כמות המקרים שנפתרו (פריטים שהוחזרו).

(2) **ניהול פריטים ומשתמשים:** המנהל יכול לעבור בין הלשונות:

- **Manage Items:** ניהול כלל המידעות במערכת.
- **Manage Customers:** ניהול המשתמשים הרשומים.

(3) **טבלת ניהול פריטים:** הטבלה מציגה את כל הפריטים שפורסמו עם המידע הבא:

- **Item Title:** שם הפריט (למשל. Ray-Ban black glasses).
- **Status:** תגית המצינית האם זה **LOST** (אבד) או **FOUND** (מצא).
- **Location:** המיקום שבו נמצא.
- **Date Reported:** תאריך הדיווח.
- **Actions:** כפתור **Delete** (מחיקה) המאפשר למנהל להסיר תוכן לא הולם או פריטים קופליים מהמערכת בלחיצה אחת.

(4) **חיפוש:** ק"מ שדה חיפוש ("Search for items or customers") המאפשר למנהל לאתר במהירות פריט או משתמש ספציפי לצורכי טיפול.

The screenshot shows the Control Center interface. At the top, there's a dark header with the title "Control Center" and a "Logout" button. Below the header, the dashboard displays three summary cards: "Total Items" (4), "Active Users" (2), and "Resolved Cases" (0). Under "Manage Items", there's a "Delete" button. A search bar at the bottom allows users to search for items or customers. The main content area shows a table of items with columns: Item Title, Status, Location, Date Reported, and Actions (with a "Delete" button). The table contains five entries:

Item Title	Status	Location	Date Reported	Actions
Ray-Ban black glasses	LOST	in shufersal haifa	2026-01-06	<button>Delete</button>
Black iPhone 7	FOUND	Ruppin academic center	2025-12-30	<button>Delete</button>
black wallet	FOUND	Park	2026-01-15	<button>Delete</button>
Hand Watch	LOST	Jaff	2026-01-15	<button>Delete</button>

Total Items

4

Active Users

2

Resolved Cases

0

Manage Items

Manage Customers

Search for items or customers.

Customer

Email Address

Join Date

Items Posted

Actions



Anas

Active

anasbareni@gmail.com

January 11, 2026

2

Block User



Test User

Active

mn.abusbo@gmail.com

January 11, 2026

2

Block User

9. קבצי המקור מצרפים בקובץ.

.10. מצורף קובץ README

.11. קישור למערכת העבודה:

<https://finders-keeper-frontend-2026.s3.us-east-1.amazonaws.com/index.html>

.12

User: mn.abuasbe@gmail.com

password : Test1234!

Admin User: noor2000asbe@gmail.com

passowrd: Admin1234!

14. רישימת ה-Services**, מה ה-Interface- של כל אחד מהם, ומה כל אחד משמש, למי הם קוראים ומי קורא להם:**

1. Amazon DynamoDB

- **ממשק:(Interface)**
 - הגישה מתבצעת بصورة תכניתית דרך AWS SDK בamuot ספריית boto3.
 - ניהול השירות מתבצע גם דרך הקונסולה של AWS ליצירת טבלאות וניהולן.
- **תפקיד:**
 - משמש כմסד הנתונים המרכזי לאחסון המידע המבני בפרויקט, כולל:
 - **נתוני משתמש** (בטבלה Users): UserID, FullName, Email,况名。
 - **נתוני פריטים** (בטבלה Items): ItemID, Title, Status (Lost/Found), Location, ImageURL, Description.
 - **נתוני הודעות/שיחות** (בטבלה Conversations): או Conversations (Messages): הר��ה נושא הקשר בין המוצא למאבד.
 - מספק ביצועים מהירים ומדרגות (Scalability) לאחסון הנתונים.
- **מי קורא לשירות:**
 - פונקציות Lambda מבצעות את כל הפעולות במסד הנתונים (קריאה, כתיבה, עדכון, מחיקה).
 - לדוגמה: הפונקציה PostItemFunction מבקשת פריט חדש לטבלה, והפונקציה GetItemsFunction מבצעת סריקה (Scan/Query) לשילוף פריטים.
- **מי הוא קורא:**
 - השירות לא קורא לשירותים אחרים ישירות, אך DynamoDB Streams יכול להפעיל פונקציות Lambda נוספת (למשל, בעת הוספה פריט חדש).

2. Amazon S3

- **ממשק:(Interface)**
 - גישה תכניתית דרך AWS SDK בamuot boto3-Python-להעלאת קבצים.
 - גישה דרך פרוטוקול HTTP/HTTPS (דף) לצפייה באתר ובקבצים.
- **תפקיד:**
 - **אחסון אתר סטטי**: (Static Website Hosting) מארחן את קבצי ה-Frontend (HTML, CSS, JS) ומגיש אותם לדף של המשתמש.
 - **אחסון מדיה**: אחסון תמונות של הפריטים שאבדו או נמצאו שימושות על ידי המשתמשים.
- **מי קורא לשירות:**

- **הדף (Client):** טוען את קבצי האתר ומציג את התמונות.
- **פונקציות Lambda:** מתקשרת עם S3 כדי לשמר תמונות שהועלו ב-Base64-או לייצר כתובות גישה.(Presigned URLs).

- **למי הוא קורא:**

- השירות אין קורא לשירותים אחרים ישירות.

3. AWS Lambda

- **ממשק (Interface):**

- מופעלת בצורה תכנית דרך אירועים (Triggers) של API Gateway, Cognito, S3, AWS Lambda ו-S3.
- ניהול הקוד והלוגיקה מתבצע דרך הקונסולה של AWS.

- **תפקיד:**

- מכילה את כל הלוגיקה העסקית (Business Logic) של המערכת, כגון:
 - **PostItemFunction:** מקבל פריט ותמונה, שמירת התמונה בס3 ושמירת המידע ב-DynamoDB.
 - **SearchItemsFunction:** סינון פריטים לפי מקום או קטgorיה.
 - **ContactOwnerFunction:** טיפול ביצירת קשר ושליחת הודעה בין משתמשים.
 - **SignUpToDynamoTrigger:** המופעל לאחר הרשמה לשימרת פרט המשתמש ב-DynamoDB.

- **מי קורא לשירות:**

- **Amazon API Gateway:** מעביר בקשות HTTP מהמשתמשים לפונקציות לעיבוד.
- **Amazon Cognito:** מפעיל טריגרים (Post Confirmation) כגון (Post Confirmation) בסיום תהליכי הרשמה.

- **למי הוא קורא:**

- **DynamoDB:** לשימירה ושליפה של נתונים.
- **S3:** לאחסון תמונות.
- **Amazon SNS:** לשילוח התראות במייל.

4. Amazon API Gateway

- **ממשק (Interface):**

- נחשף כAPI RESTful-למשתמשים ול-Frontend.
- מנהל דרך הקונסולה להגדרת Routes כגון /users, /items ו/integrations.

- **תפקיד:**

- מהוות את נקודת הכניסה היחידה (Entry Point) של הבקשות מהדפדפן לשרת Backend).
- מנתב את הבקשות (GET, POST, DELETE) לפונקציית ה Lambda-Authorizer.
- מספק שכבת אבטחה ואמות באמצעות המודול **Cognito Authorizer** שרק משתמשים מחוברים עם Token תקין יכולים לבצע פעולות (כמו פרסום פריט).
- **מי קורא לשירות:**
- **(h) Frontend-Aתור הלקוח:** (שולח בקשות Ajax/Fetch לbijouter פועלות במערכת).
- **למי הוא קורא:**
- **פונקציות Lambda:** מעביר אליו את המידע לעיבוד ומחזיר את התשובה ללקוח.

5. Amazon Cognito

- **ממשק:(Interface)**
- מנהל דרך AWS SDK API ו-HTTP-Lצורך אימות וניהול זהויות.
- **תפקיד:**
- ניהול משתמשים (User Pool): רישום, התחברות, שחזור סיסמה וניהול פרופילים.
- אימות (Authentication): הנפקת אס'מוני גישה (JWT Tokens) לאחר התחברות מוצלחת.
- אבטחה: וידוא כתובת אימייל (Email Verification) באמצעות שליחת קוד OTP.
- **מי קורא לשירות:**
- **(h) Frontend:** מבצע מולו שירות את תהליכי ההרשמה (Sign Up) וההתחברות (Sign In).
- **API Gateway:** משתמש ב Cognito-Token לאמת את ה Token-Usage בבקשת ה- API.
- **למי הוא קורא:**
- מפעיל פונקציית Lambda (SignUpToDynamo) באופן אוטומטי לאחר אישור הרשמה של משתמש חדש.

6. Amazon SNS (Simple Notification Service)

- **ממשק:(Interface)**
- מופעל תכניתית דרך (boto3) AWS SDK מתוך פונקציית Lambda.
- **תפקיד:**
- שירות שליחת הודעות והתראות. בפרויקט זה משמש בעיקר לשילוח הודעות דוא"ל (Email Notifications).
- שימושים: שליחת קוד אימות בהרשמה (נעשה אוטומטית ע"י Cognito המשתמש בתשתית זו (ושליחת התראות למנהל המערכת או למשתמשים (למשל: "משהו מצא את הפריט שלך").

- **מי קורא לשירות:**

AWS Lambda: ○
Amazon Cognito: ○

- **למי הוא קורא:**

○ שולח את הודעה לכתובת המail של הנמען.(End User)