

Team: Idan Lazar, Yonatan Ben Avraham, Ronel Shemtov, Ori Litman, Yuval David

Supervisor: Eliav Menachi

**Final Project Book**

**תוכן עניינים**

שער

תוכן עניינים

תקציר הפרויקט ................................................................................................................................ 3

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

מה נעשה בתחום ............................................................................................................................. 5

הפתרון ............................................................................................................................................ 6

הצדקה אקדמאית ............................................................................................................................. 9

תרשימים ....................................................................................................................................... 10

DR: Design Requirements ........................................................................................................... 13

SRS: Software Requirements Specification ................................................................................. 15

מתודולוגיות וכלי הפיתוח, חומרה ותוכנה ........................................................................................... 16

תיאור המערכת – כולל תמונות מסך ................................................................................................. 17

תוצרי הפרויקט ............................................................................................................................... 18

ביבליוגרפיה ................................................................................................................................... 20

תקציר פרויקט באנגלית ................................................................................................................... 21

**תקציר פרויקט**

העולם בו אנו חיים מוצף במידע בכל תחומי החיים וכיום צרכני המידע עושים שימוש באפליקציות מידע רבות אשר בהן נדרשים לחפש אקטיבית את המידע אשר הם מעוניינים לצרוך.

לדוגמא, שוטטות באפליקציית Ynet אחר כתבות שמעניינות אותנו. עבור **כל 10 כתבות** בממוצע שהמשתמש נחשף אליהן הוא מוצא **אחת** שמעניינת אותו ועליה הוא מקליק ונכנס לקרוא בעיון.

אפליקציית LikeiT מקנה למשתמש את האפשרות לקבל את המידע אותו הוא מעוניין לצרוך "בדחיפה" מבלי לחפש אחריו, ע"י למידת המשתמש ותחומי העניין שלו.

LikeiT היא אפליקציית מידע ולה בינה מלאכותית חכמה אשר לומדת את תנועותיו של המשתמש, מנתחת אותו ומסיקה מסקנות על כל משתמש בנפרד ועל קבוצות של משתמשים דומים זה לזה.   
LikeiT היא גם פלטפורמה לפידבקים ותגובות של משתמשים. למשתמש ישנה אפשרות לכתוב תגובה על כתבה מסוימת והמערכת יודעת להציג את התגובה הזו לאנשים הרלוונטיים לאותו קטגוריה,  
וכך מתאפשר שיח ציבורי על הנושאים החמים ברשת באמצעות LikeiT.

LikeiT מביאה עמה מהפכה באופן שבו נגשים למידע ולמעשה מספקת מערכת הפוכה לגוגל, לפייסבוק ולכל מה שקיים היום בשוק - במקום לחפש אחר המידע, בעולם של LikeiT המידע יחפש אותנו.

ארכיטקטורת המערכת שלנו מורכבת משלוש יחידות מרכזיות –

1. **יחידת איסוף תכנים** – המאתרת בצורה חכמה את התכנים הרלוונטיים ברשת
2. **מערכת המלצות חכמה** – המבצעת ניתוח של תכנים והתנהגויות משתמשים וממליצה   
   על התאמות בין תכנים למשתמשים.
3. **אפליקציית המשתמש** – המנהלת את האינטראקציה מול המשתמשים.   
   האפליקציה מבוססת על מודל ה-HOOKED שבאמצעותו נותנים למשתמש חווית צריכה נעימה יותר וממכרת הרבה יותר!

**הגדרת עולם הבעיה**

*מתוך ויקיפדיה: "היצף מידע או עודף מידע הוא מושג המתייחס למצב שבו יש יותר מדי מידע, הכמות העודפת של המידע מפריעה לקבלת החלטה וליכולת להתעדכן.  
היצף מידע הוא מצב קיומי של הפרט, בו אדם מסוים סובל מחוסר יכולת להשתלט על כל הגירויים והעובדות המוצבים בפניו. זהו מצב החורג מיכולת עיבוד המידע של האדם.."*

בשנתיים האחרונות ייצרה האנושות יותר מידע משייצרה בכל ההיסטוריה עד אז. גם גוגל ופייסבוק רחוקות מלמצות את המידע שנערם. כדי להתמודד עם האתגר, צרכני המידע — שהם כולנו — עוברים מחיפוש לסינון. שפע המידע, שאמור היה לאפשר לצרכן להפוך לחוקר עצמאי, דווקא דוחף אותו להגיע כמה שיותר מהר רק למה שרלוונטי. מי שיסייע לו להגיע במהירות לשורה התחתונה, יעניק לו ערך אמיתי ויהפוך את עצמו למקור מידע חשוב וקבוע.

מהפכת המידע הביאה לשינוי מהותי גם בעולם העסקי: כיום גם חברות מסחריות הן גופי תוכן. הן מייצרות מידע ומשתתפות בזרם המידע שאופף את הצרכנים. דירוגים ומדדים הופכים מכלי המסייע לסידור המידע, לכלי שיווקי המסייע להעברת המסר לצרכן.

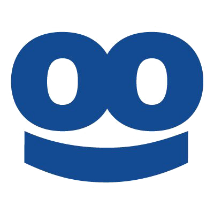
אנו חיים בעולם של BIG DATA, ומוצפים במידע שעולה לרשת בכל שנייה ולכן לא מצליחים להגיע תמיד לכל מה שבאמת מעניין אותנו.   
בכל כניסה לאפליקציית מידע כזו או אחרת אנו מבזבזים המון זמן בחיפוש ברשת.

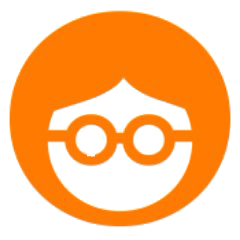
הבעיה איתה LikeiT רוצה להתמודד הינה בעיית הצפת המידע שחווים צרכני המידע.

כיום, כל אדם בעל סמארטפון נחשף למס' אפליקציות אשר נועדו לחשוף אותו למידע בכל תחומי החיים.   
כל אפליקציה כזו הינה מתחרה של האחרות ולכן שואפת תמיד להישאר אקטואלית ורלוונטית עבור המשתמשים ופועלת בשיטה של דחיפת התראות והנגשה של כמה שיותר מידע באפליקציה מכל תחומי החיים, ללא סינון וללא יכולת של התאמת המידע למשתמש הספציפי.   
כלומר, גרסאות האפליקציות של כלל חברות החדשות והמידע כמו Ynet ומקבילותיה יהיו זהות לכלל המשתמשים וכולם יראו את אותן הכתבות באותו זמן נתון.   
מצב זה הינו מצב אבסורדי שכן בעולם טכנולוגי ומפותח בו אנו חיים כיום, אין לכך כל סיבה.   
לכל משתמש יש תחומי עניין שונים והעדפות שונות מחבריו וכל משתמש מחפש אחר תכנים שמעניינים אותו באותו מאגר ענק של מידע בו הוא מוצף.

**מה נעשה בתחום**

בתוך עולם של שטף מידע, היכולת להתאים את המידע למשתמש היא כלי צרכני ממעלה ראשונה.   
האתר Buzz-Feed הפך לאחד המצליחים בעולם באמצעות סיקור תמציתי וחיבה לדירוגים.   
אתר החדשות Axios מסקר כל אירוע, גדול כקטן, בנקודות ספורות ולסיכום מסביר מדוע הידיעה חשובה;   
גם אתרים ישראליים עושים שימוש גובר ברשימות דירוגים.

חברות מובילות בשוק הישראלי אשר מנסות להתמודד עם בעיה זו של שטף המידע הן Taboola ו- OutBrain.  
חברות אלו מציעות רשת פרסום לקידום כתבות באתרי התוכן הגדולים באמצעות "תוכן שיווקי" –   
שילוב בין פרסומת רגילה, לבין תוכן "רגיל" - כמו כתבה בעיתון או באתר אינטרנט, ראיון בתוכנית אירוח בטלוויזיה, מוצרים המופיעים בהבלטה בסרט ועוד.

Outbrain ו Taboola - אלו הן שתי החברות העיקריות בישראל המציעות המלצות תוכן בכל אתרי התוכן המוכרים, ומציגות קישורים לכתבות על השירות או המוצר של המפרסם.

הבסיס לפרסום בהן הוא כתבה הנכתבת בדומה לכתבות עיתונאיות רגילות – בניסוח הכותרת, במבנה הכתבה, ובכך שהיא מביאה לקורא תוכן מעניין וחשוב עבורו. תוך כדי דיון בנושא או בתופעה מסוימת, מציינת הכתבה גם את שמו של בעל עסק מסוים הפועל בתחום ומציע מוצר או שירות הקשורים בו. אלא שבניגוד לפרסומות רגילות, בחברות אלו הכתבה אינה שיווקית אלא יותר אובייקטיבית (אחרת היא לא תאושר לפרסום מלכתחילה), ומי שקורא אותה לאו דווקא מרגיש שמנסים למכור לו מוצר או שירות כלשהו.

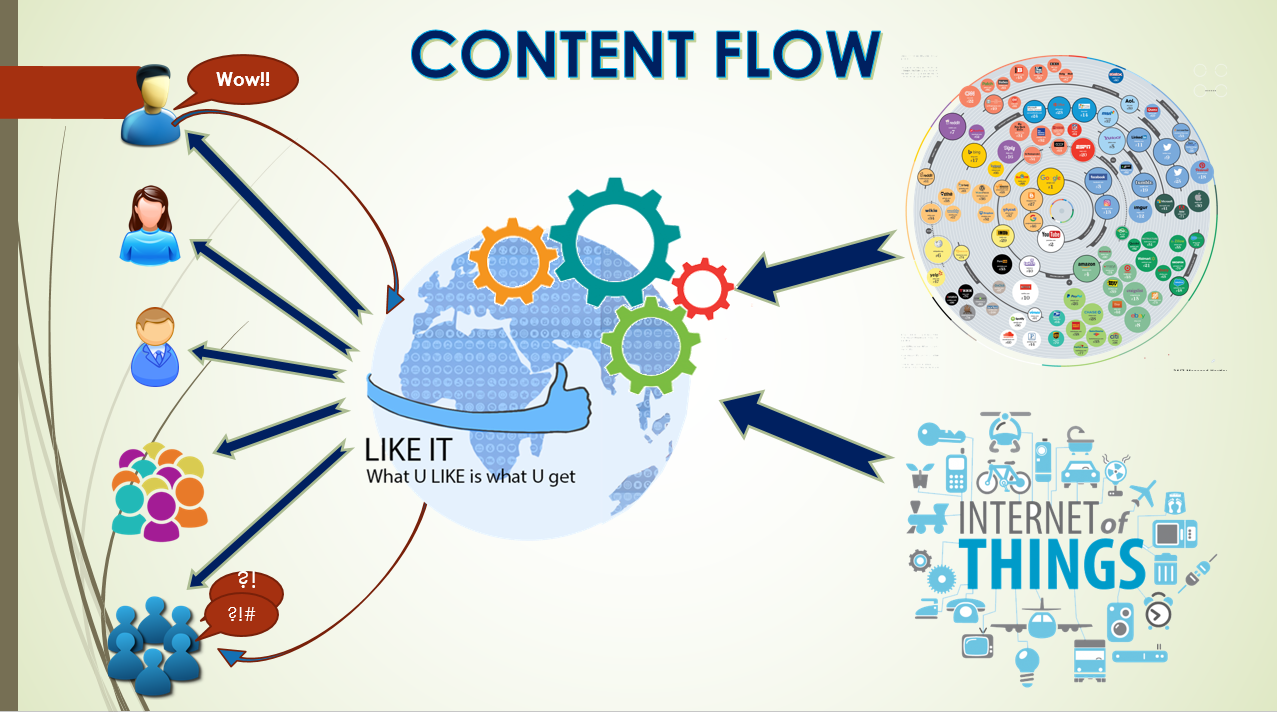
לאחר שהכתבה מוכנה, החברות הללו מאפשרת לשלב אותה באתרי תוכן נפוצים, ביניהם "מאקו", "הארץ", "דה מארקר", "כלכליסט" ועוד. מי שקורא כתבה "רגילה" באתר ומגיע לסופה, רואה כמה לינקים לכתבות אחרות, תחת כותרת בנוסח כמו "כתבות נוספות שעשויות לעניין אותך". בלחיצה על הלינק, הוא יגיע לכתבת התוכן השיווקי, המעוצבת בדומה לשאר הכתבות באתר. בסוף הכתבה לרוב יהיו ממוקם טופס לידים – המאפשר למתעניינים להשאיר פרטי קשר, כדי שיחזרו אליהם.

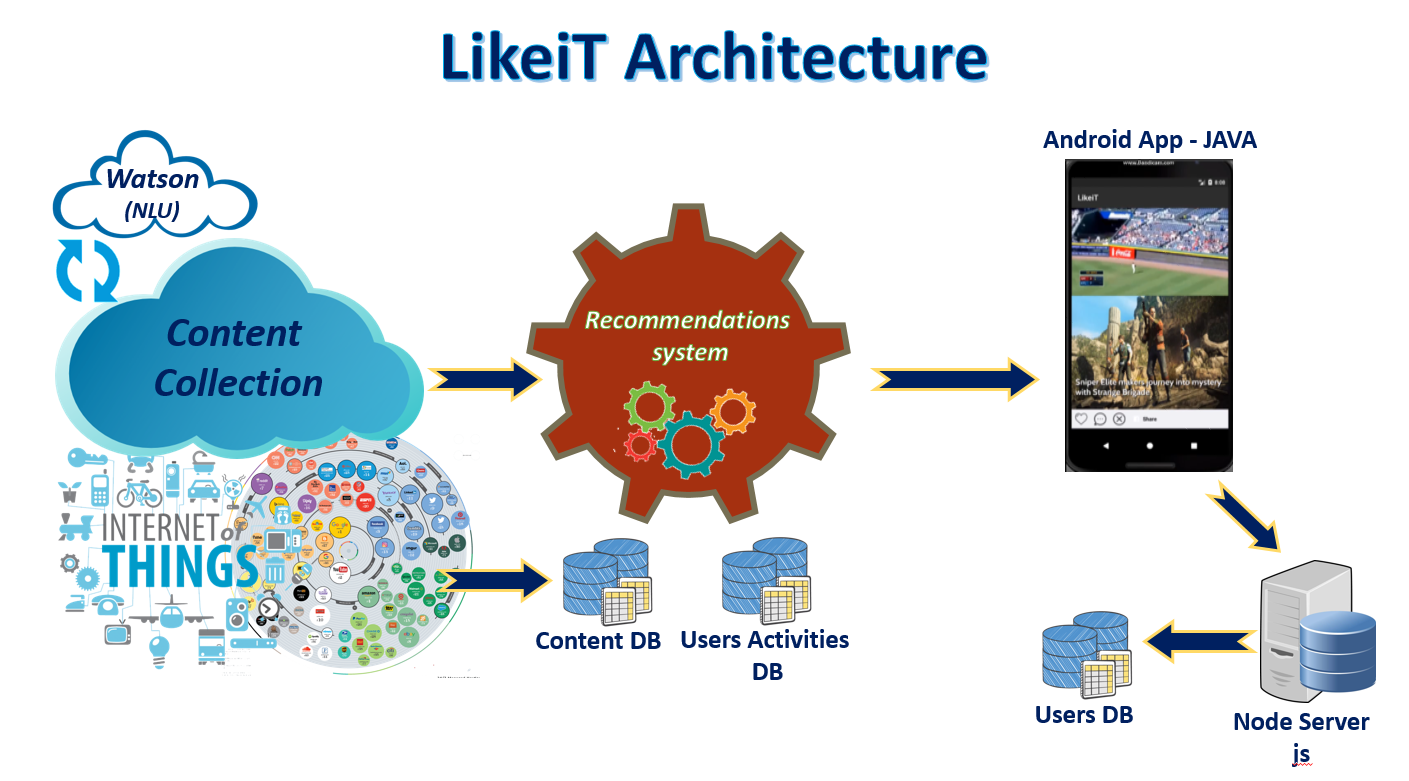
לרוב, כתבות שהן "תוכן שיווקי" לא מתאימות למוצרים או שירותים שקהל היעד שלהם מצומצם מאוד.   
זאת, מאחר שבמקרים כאלה רק קוראים מעטים יטרחו להקליק על הלינק ולקרוא את הכתבה.   
קמפיינים אלה מתאימים יותר למוצרים או שירותים בעלי קהל יעד של עשרות אלפי אנשים או יותר.   
ככל שיותר אנשים ימצאו את הכתבה מעניינת עבורם ויקליקו לקריאתה, הקמפיין יהיה יעיל יותר.

**הפתרון**

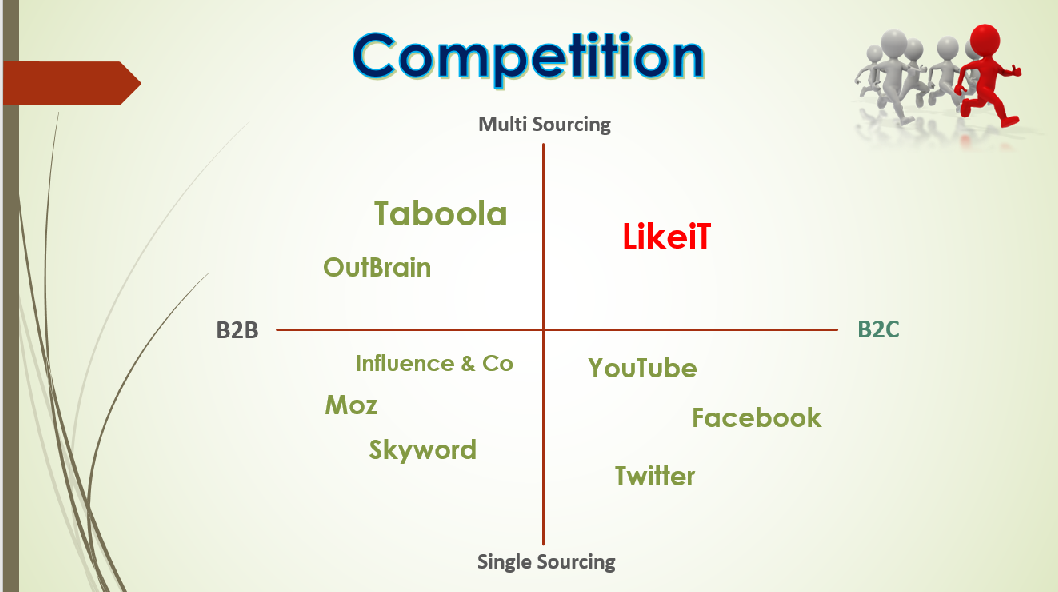
LikeiT נועדה לתת מענה לצרכני המידע אשר מוצפים במגוון תחומים דרך מגוון רחב של אפליקציות מידע מתחרות. היא נועדה לרכז עבור כל משתמש אינדיבידואלי את התכנים המעניינים אותו בלבד ולבטל את תחושת ההצפה והרדיפה אחר מידע מעניין.

LikeiT יושבת בין יצרני המידע המזינים את האינטרנט בתכנים לבין צרכני המידע השונים.   
המערכת אוספת בצורה חכמה את המידע מהמקורות השונים, ובאמצעות מנגנון למידה חכם ומערכת המלצות יעילה – מזרימה את המידע למשתמשים הרלוונטיים לו.

****LikeiT היא גם פלטפורמה לפידבקים ותגובות של המשתמשים. למשתמש ישנה אפשרות לכתוב תגובה על כתבה מסוימת שראה והמערכת מציגה את התגובה הזו לאנשים הרלוונטיים לה ובכך מאפשרת שיח ציבורי על הנושאים החמים ברשת.  
LikeiT למעשה זוהי אפליקציית מידע פשוטה, המהווה מעין מתווך בין יצרני המידע לצרכנים שלו.   
היא איננה אפליקציה שמייצרת את המידע בעצמה, אלא אוספת מידע באמצעות אלגוריתמים חכמים אשר מנתרים מידע ברשת ממקורות רבים ומגוונים זאת ע"י דירוגים ופידבקים של משתמשים אשר מצביעים על תכנים אקטואליים ומעניינים. כך, היא אוספת אליה מידע מגוון ככל האפשר, ובאמצעות מערכת למידה מורכבת אשר לומדת כל הזמן את התנהגויות המשתמשים ואת תכני הכתבות, יודעת להתאים בין השניים.

**ארכיטקטורת המערכת בנויה משלוש יחידות מרכזיות:**   
1. יחידת איסוף התכנים – המאתרת בצורה חכמה התכנים הרלוונטיים והחמים ברשת.  
2. מערכת המלצות חכמה – המבצעת ניתוח תכנים והתנהגויות וממליצה על התאמות בין תכנים למשתמשים.  
3. אפליקציית המשתמש- המנהלת את האינטראקציה מול המשתמשים.

LikeiT מהווה פתרון גלובלי לעולם הבעיה שהוצגה לעיל כאשר היא מאפשרת לצרכני המידע השונים לקבל בדיוק מה שמעניין אותם ללא צורך במעורבות אקטיבית שלהם, ללא חיפושים ושוטטות ברשת. ההבדל המרכזי בינה לבין החברות האחרות שהוצגו הוא שכך שהם אינם משחקים על אותו מגרש של קהל יעד ומטרותיהם שונות בתכליתן. חברות כמו Taboola ו- OutBrain – הן כולן חברות B2B, שכבר הוכיחו את הצורך לעשות התאמת תכנים למען מטרה עסקית – השארת הגולשים כמה שיותר זמן באתר מסוים, כי הזמן הזה זה שווה כסף מחשיפה לפרסום.

LikeiT לוקחת את הצורך הזה הלאה למגרש של ה-C2B ומממשת את אותן המטרות- עבור חווית המשתמש בלבד. בנוסף, LikeiT היא מערכת גנרית שמתאימה עצמה למגוון רחב של מקורות מידע ולא לאתר או מקור מידע נקודתי אחד כמו החברות האחרות.

LikeiT זו מהפכה בעולם טכנולוגי בו הכל נעשה מתוחכם ומוצף במידע. היא מסוג האפליקציות שתהלנה בעתיד את החיים של צרכני המידע תיתן להם בדיוק מה שהם רוצים, במקום ובזמן המתאים.   
LikeiT מציעה למשתמשיה להפסיק לרדוף אחר המידע, ולתת למידע להתחיל לרדוף אחריהם.

**הצדקה אקדמית**

החלק הראשון הוא איסוף נתונים מהAPI-ים השנויים. איסוף זה מבוצע על ידי סקריפט שנכתב בשפת Python אשר רץ בשרת איסוף, פועל כל מספר דק ודוגם את API-ים המוגדרים לנו.

לאחר מעבר על כל הAPI-ים ובדיקה האם הכתבות לא נמצאות ב-DB, השרת כותב ל-DB את כל הכתבות החדשות.

החלק השני הוא מערכת הלמידה שכתובה על השרת ויודעת למצוא לכל המשתמש את הפריטים הרלוונטיים עבורו.

המערכת פועלת בצורה היברידית ומשתמשת בשתי מערכות המלצה (לפי התאמה למשתמש)

בנוסף נאסף מידע אשר מראה לנו איך המשתמש מגיב לכל כתבה המוצגת עבורו וכך ניתן להשתמש בצורה חכמה במערכת ההמלצה.

החלק השלישי הוא אפליקציית אנדרואיד אשר יודעת להציג למשתמש את כל התוכן הנדרש ויודעת להביא למערכת ההמלצה את המעקב של התנועות אחרי המשתמש.

ניתן לראות כי בכל חלקי הפרויקט יחד, השתמשנו במגוון רחב של כלים ושפות אותם למדנו לאורך התואר.

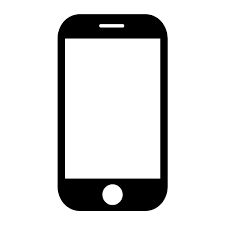
**תרשימי בלוקים ומודולים כללי של המערכת**

Mongo

DB

Server

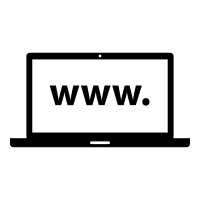
(Node.js)



Android or

IOS App.

(React Native)



Office website

(React.js)

**Software Models**

User

First name – User’s first name

Last name – User’s last name

Id – User’s id

Birthday – User’s birthday

Authorization – The authorization level in the app.

Password – User’s password for the app

Phone – User’s Phone number

Gender – User’s gender

Branch - The store name he works in

Link

Title – Link’s description

Link – The actual link

Daily Briefing

Title – Briefing’s title

File Name – The name of the file needed to be displayed in the app

Minhal

Title – Minhal’s title

File Name – The name of the file needed to be displayed in the app

Job

Title – job’s title

Salary – job’s salary

Message

Title – Message’s title

Content – Message’s content

Link – Message’s link

Created Date – Message’s creation date

Guidance

Title – Guidance’s title

File Name – The name of the file needed to be displayed in the app

**Sequence Diagram**

**Login sequence**

User

View

Server

DB

User

View

Server

DB

Type user and password

Click login

Sending a login request with user and password in the body

Searching for a user with the given id and password in the DB

No user was found

Returning an error to the client

A user was found

Creating a token and sending it to the user

Saving the token in the cache and moving to the main screen

Printing an invalid user error to the screen

Saving the token to the DB for the given user

User

View

Server

DB

User

View

Server

DB

Making an action that requires getting a model

Sending a request to the server for getting the wanted model

Checking the token and if the user has an authentication for getting the wanted model.

If user has the required authentication, getting the model/models

Returning a response with the wanted model/models

Shows the wanted screen with the models if no error occurred

Sending a response with unauthorized error to the client

Shows the unauthorized error to the user.

If user has the required authentication returning an error

**Getting a model sequence sequence**

User

View

Server

DB

User

View

Server

DB

Making an action that requires saving a model

Sending a request to the server for saving the wanted model. With the model in the body

Checking the token and if the user has an authentication for saving the model

If user has the required authentication, saving the model

Returning a response with 200 OK to indicate the action’s success

Shows a success message to confirm the action

Sending a response with unauthorized error to the client

Shows the unauthorized error to the user.

If user does not have the required authentication returning an error

**Saving a model sequence sequence**

User

View

Server

DB

User

View

Server

DB

Making an action that requires deleting a model

Sending a request to the server for deleting the wanted model. With the model id in the body

Checking the token and if the user has an authentication for deleting the model

If user has the required authentication, deleting the model

Returning a response with 200 OK to indicate the action’s success

Shows a success message to confirm the action

Sending a response with unauthorized error to the client

Shows the unauthorized error to the user.

If user does not have the required authentication returning an error

**Deleting a model sequence sequence**

User

View

Server

DB

User

View

Server

DB

Making an action that requires updating a model

Sending a request to the server for updating the wanted model. With the model id to update and the new model in the body

Checking the token and if the user has an authentication for updating the model

If user has the required authentication, updating the model

Returning a response with 200 OK to indicate the action’s success

Shows a success message to confirm the action

Sending a response with unauthorized error to the client

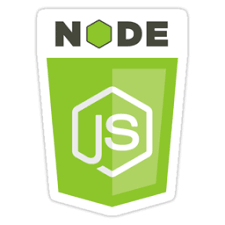
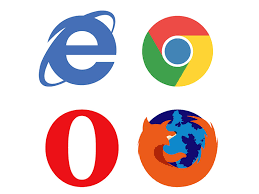
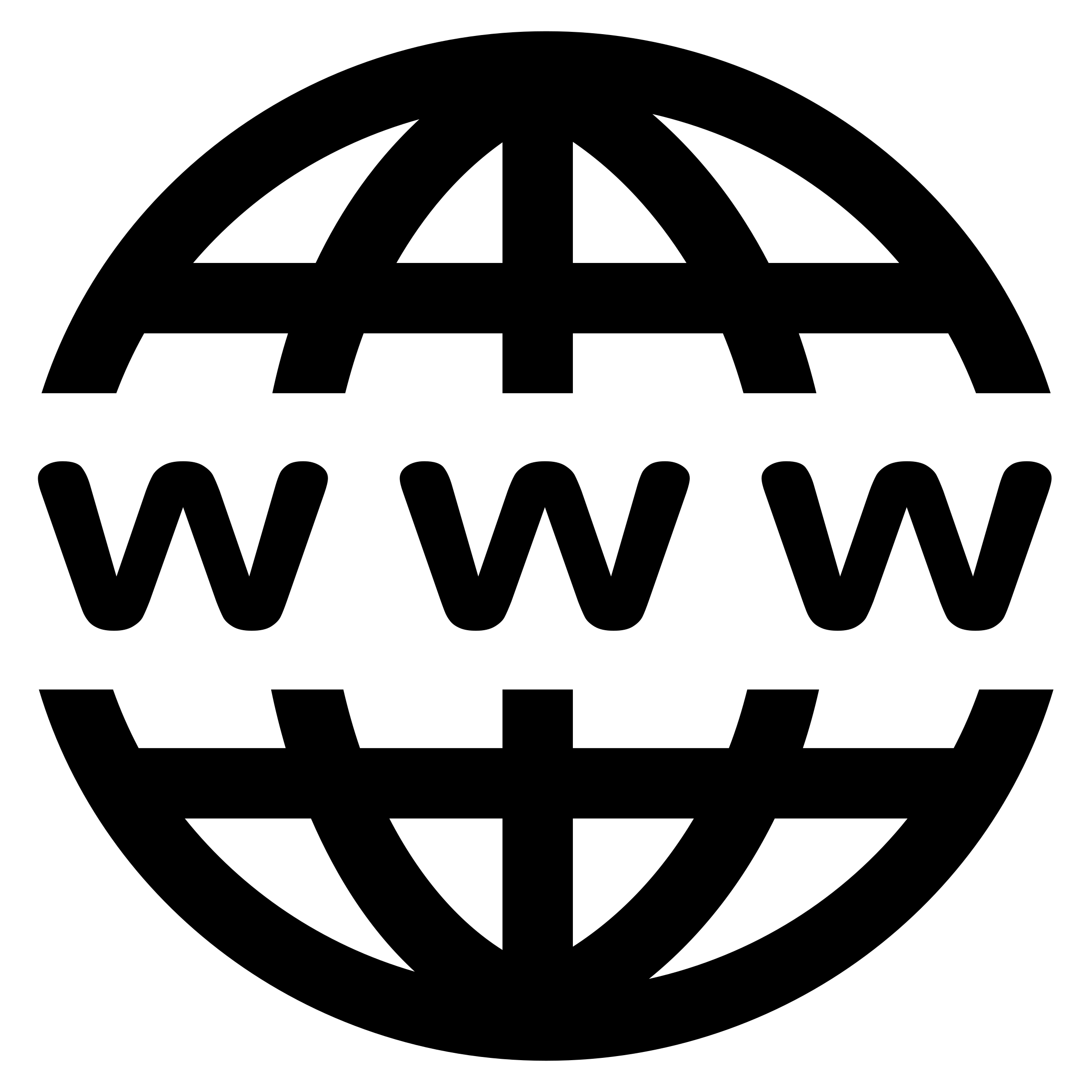
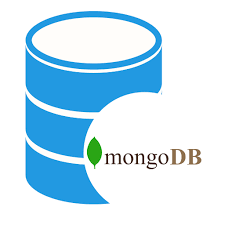
Shows the unauthorized error to the user.

If user does not have the required authentication returning an error

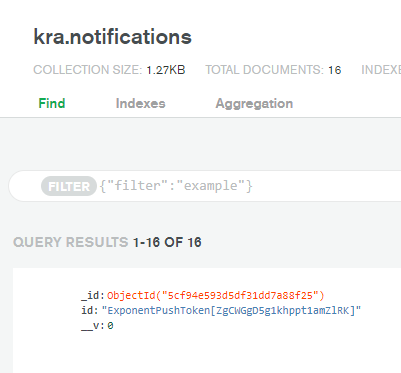
**Updating a model sequence sequence**

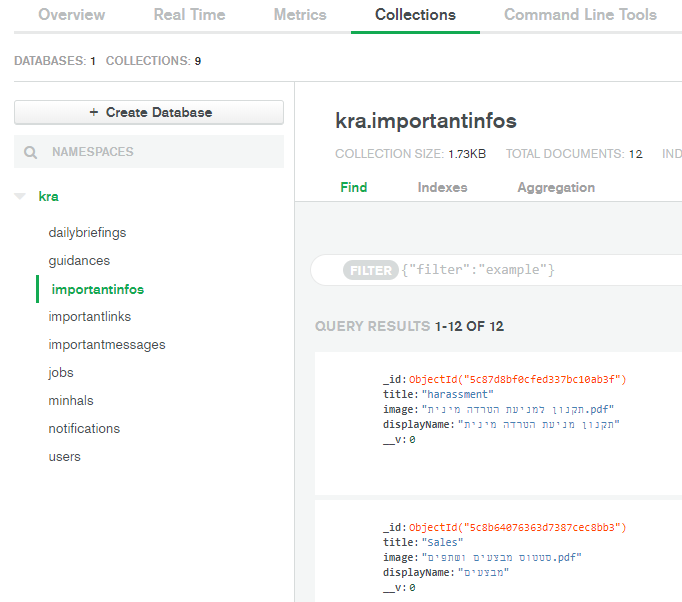
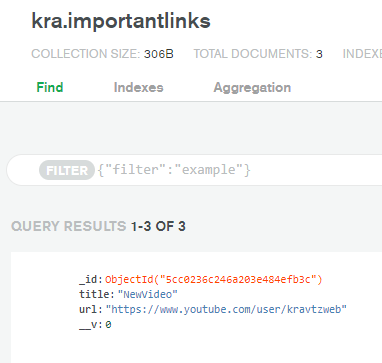
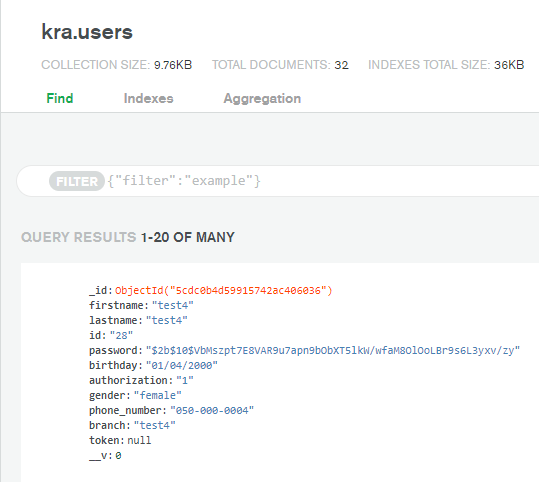
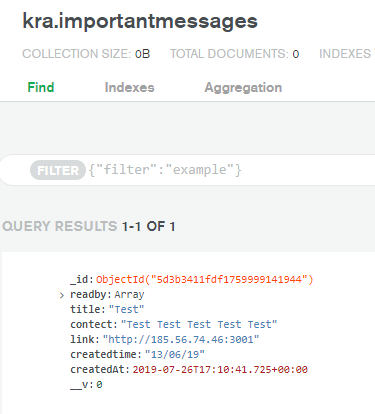
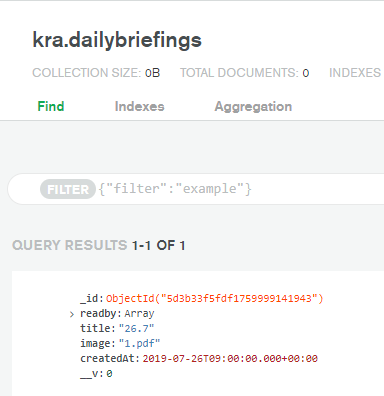
**DR: Design Requirements**

מערכת Kravitz בנויה לפי ארכיטקטורת Client – Server



* 1. **Data tables**

We'll have a MongoDB database, that will save all the models in our system. Each model may have different properties. The mongoDB is a non relational database which means there will be no relations between the models.



**Technologies Description**

The client side includes 2 part:

Client side – Android and IOS app

development environment: Visual Code

language: React Native

Libraries: Described in the package.json file

Client side – Office Web Site

development environment: Visual Code

language: ReactJS

Libraries: Described in the package.json file

The server side includes:

Server side – Node.js

development environment: Visual Code

language: JavaScript

DB: MongoDB

Libraries: Described in the package.json file

**Main classes:**

User, Messages, Daily Briefing, Jobs, Notifications, Info, Guidance, Minhals, Link.

**Main functions:**   
Login – when login into the website or application.

Get/GetAll – Gets am instance or a list of a specific model from the database.

Save – Saves a new model to the database.

Update – Updates a model in the database with new values.

Delete – Deleting a model from the database by its IP.

**SRS: Software Requirements Specification**

**Use Cases**

1. המשתמש מתחבר לאפליקציה עם תעודת זהות וסיסמה.
2. המשתמש מקבל תדריכים יומיים והדרכות ועליו לאשר את קריאתו.
3. המערכת מקבלת מסמכים מהadmin ומציגה אותו למשתמשים.
4. המשתמש יכול לחפש את ימי ההולדת של העובדים.
5. המשתמש יכול לחפש משרות של "חבר מביא חבר".

**דרישות כלליות מהמערכת**

1. המערכת תדע בזמן אמת את כל אישורי הקריאה של כל העובדים.
2. המערכת תדע לטפל במספר משתמשים במקביל.
3. המערכת תיהיה נוחה וקלה לשימוש.
4. המערכת תנהל בסיס נתונים עבור המשתמשים והמידע.

**דרישות פונקציונאליות מהמערכת**

* המערכת חייבת לתקשר עם המכשיר הסלולרי
* המערכת חייבת להציג למשתמש מסמכי pdf .
* המערכת חייבת לזהות את בחירות המשתמש (אישור קריאה, מסכים שונים)
* המערכת חייבת לאפשר הזנת מסמכים חדשים ע"י שירותי API

**דרישות לא פונקציונאליות מהמערכת**

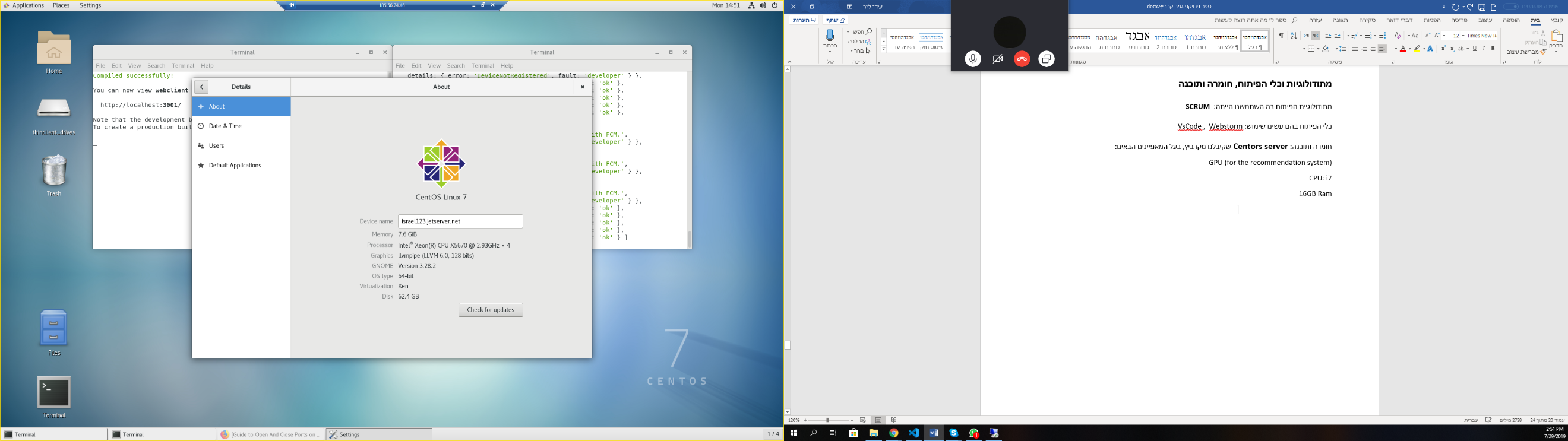
* המערכת חייבת להתבסס על מסד נתונים שיתחזק את כל הנתונים
* המערכת חייבת לתמוך בריבוי משתמשים במקביל
* המערכת חייבת לאפשר הצגת מסמכים.
* המערכת חייבת לתמוך בכל המכשירים.

**מתודולוגיות וכלי הפיתוח, חומרה ותוכנה**

מתודולוגיית הפיתוח בה השתמשנו הייתה: **SCRUM**

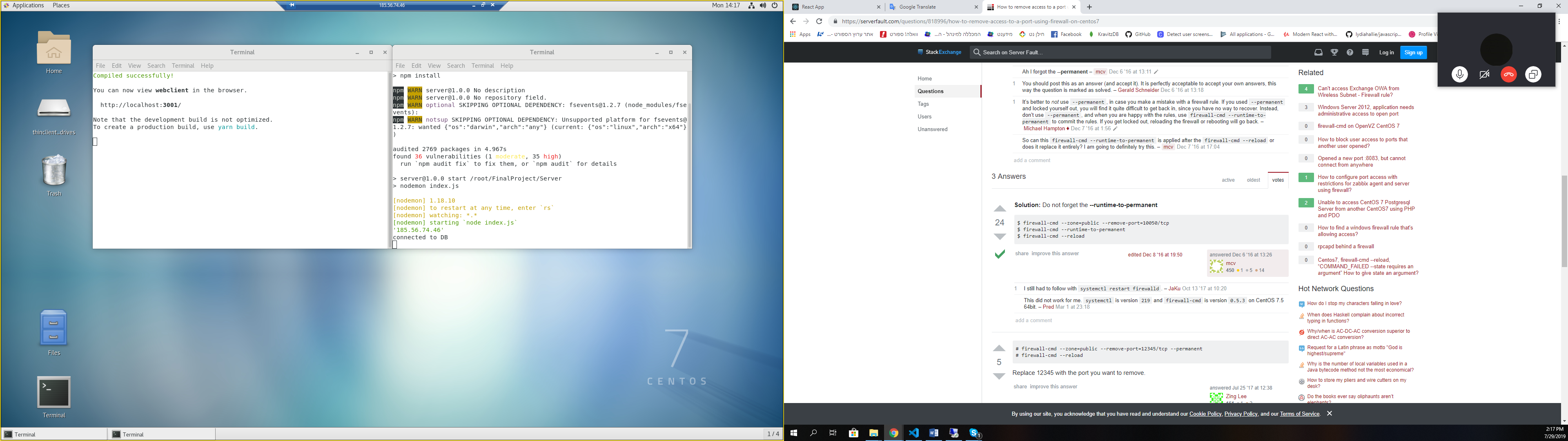
כלי הפיתוח בהם עשינו שימוש: Webstorm , VsCode

חומרה ותוכנה: **CentOS server**שקיבלנו מקרביץ, בעל המאפיינים הבאים:



**תיאור המערכת – כולל תמונות מסך**

צד הלקוח כולל אפליקציית Android המופעלת ע"י שרת Node.js

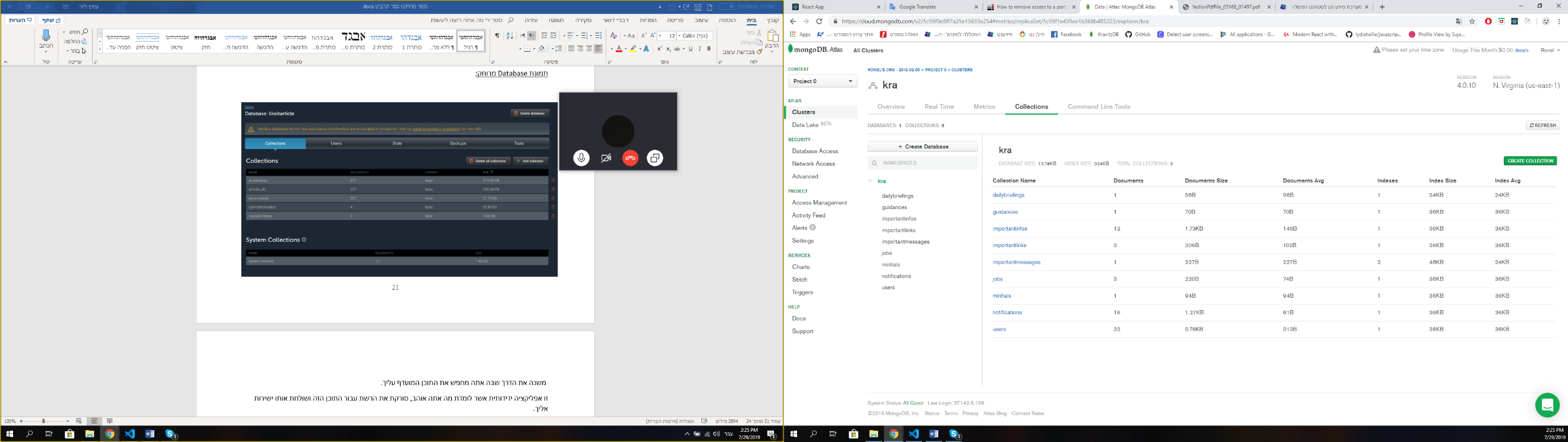
תמונת השרת בפעולה:

dddd

חיבור האתר על פורט 3001

חיבור השרת על פורט 3000

תמונת Database מרוחק:



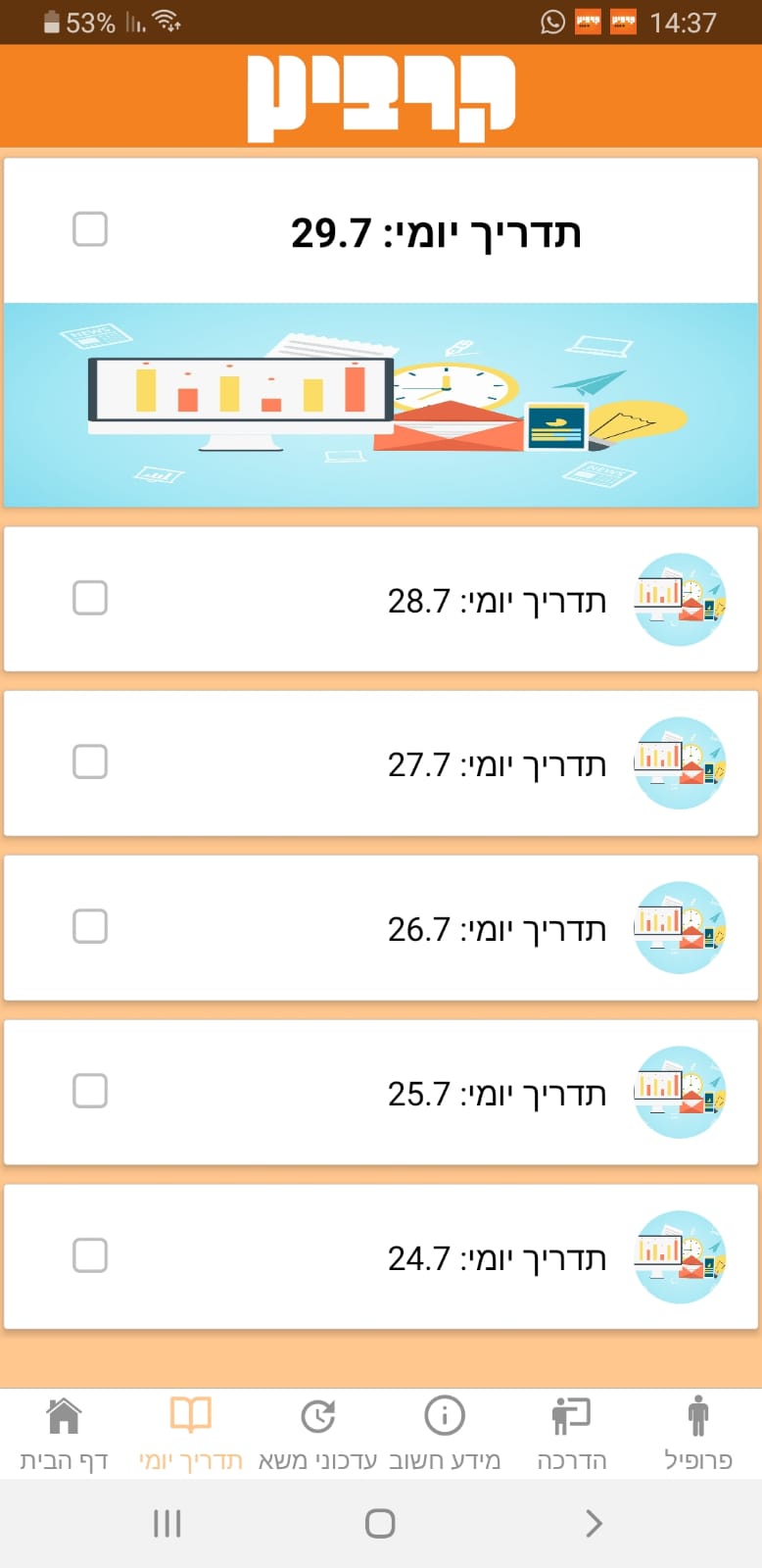
קרביץ משנה את הדרך בניהול עובדים.

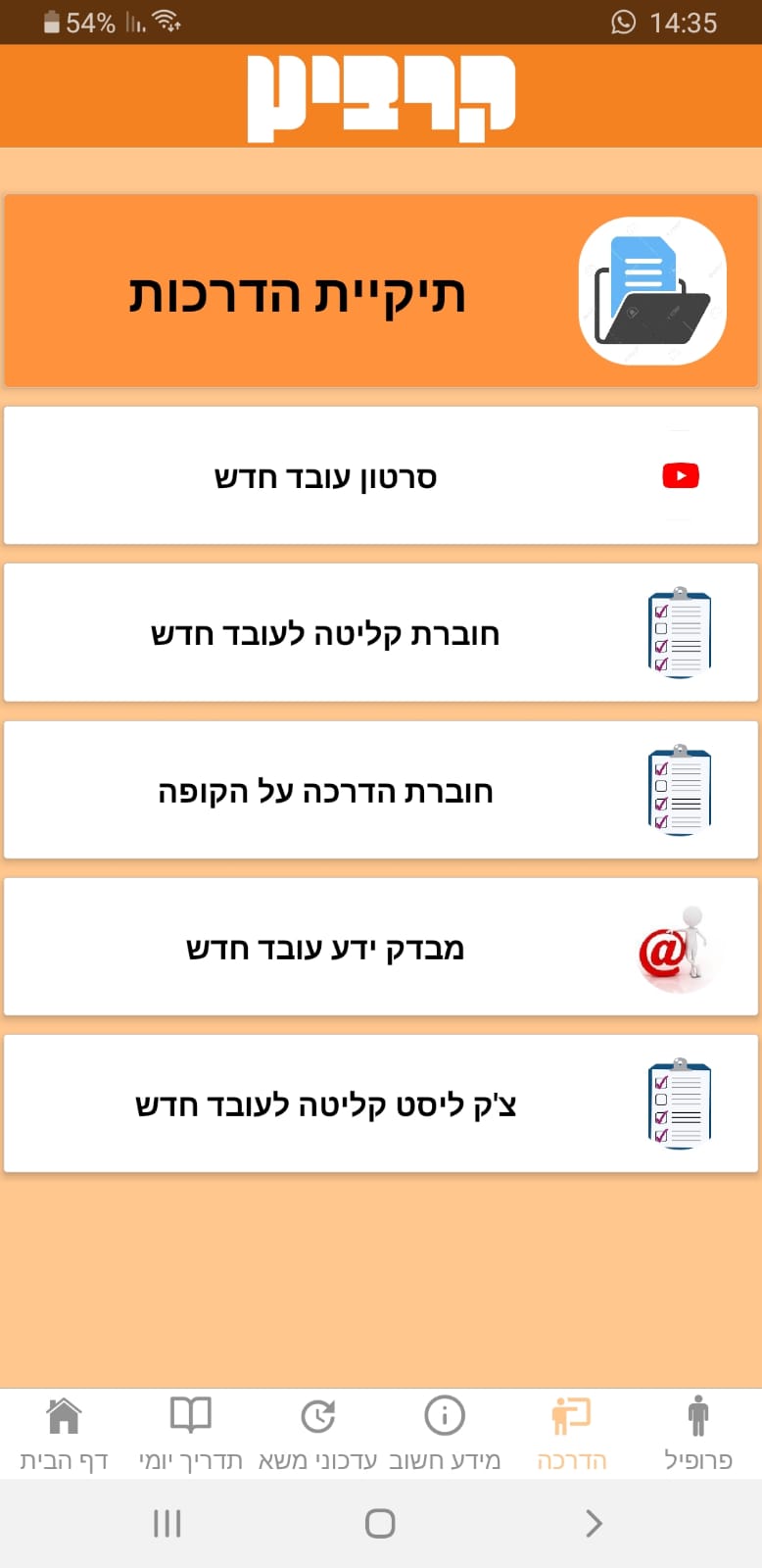
זו אפליקציה ידידותית אשר עוזרת לנהל את העובדים בצורה יעילה ומיטבית תוך שמירה על פרטיות והיכולת לנהל ולשתף מסמכים בצורה מהירה ומסודרת.

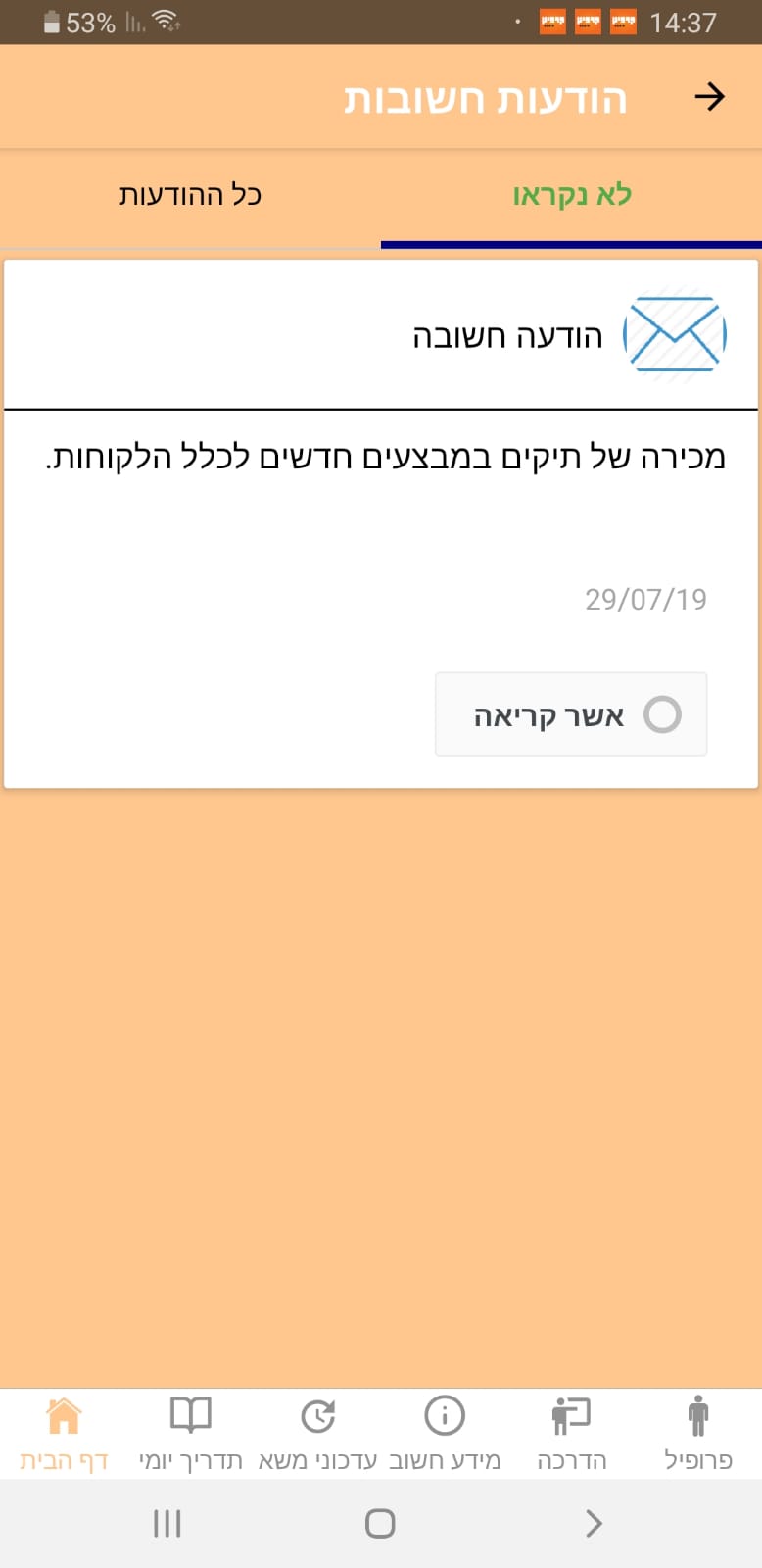
הייחודיות של האפליקציה היא בניהול העובדים מה שלא קיים כיום בשוק בצורה קלה ובמחיר שפוי.

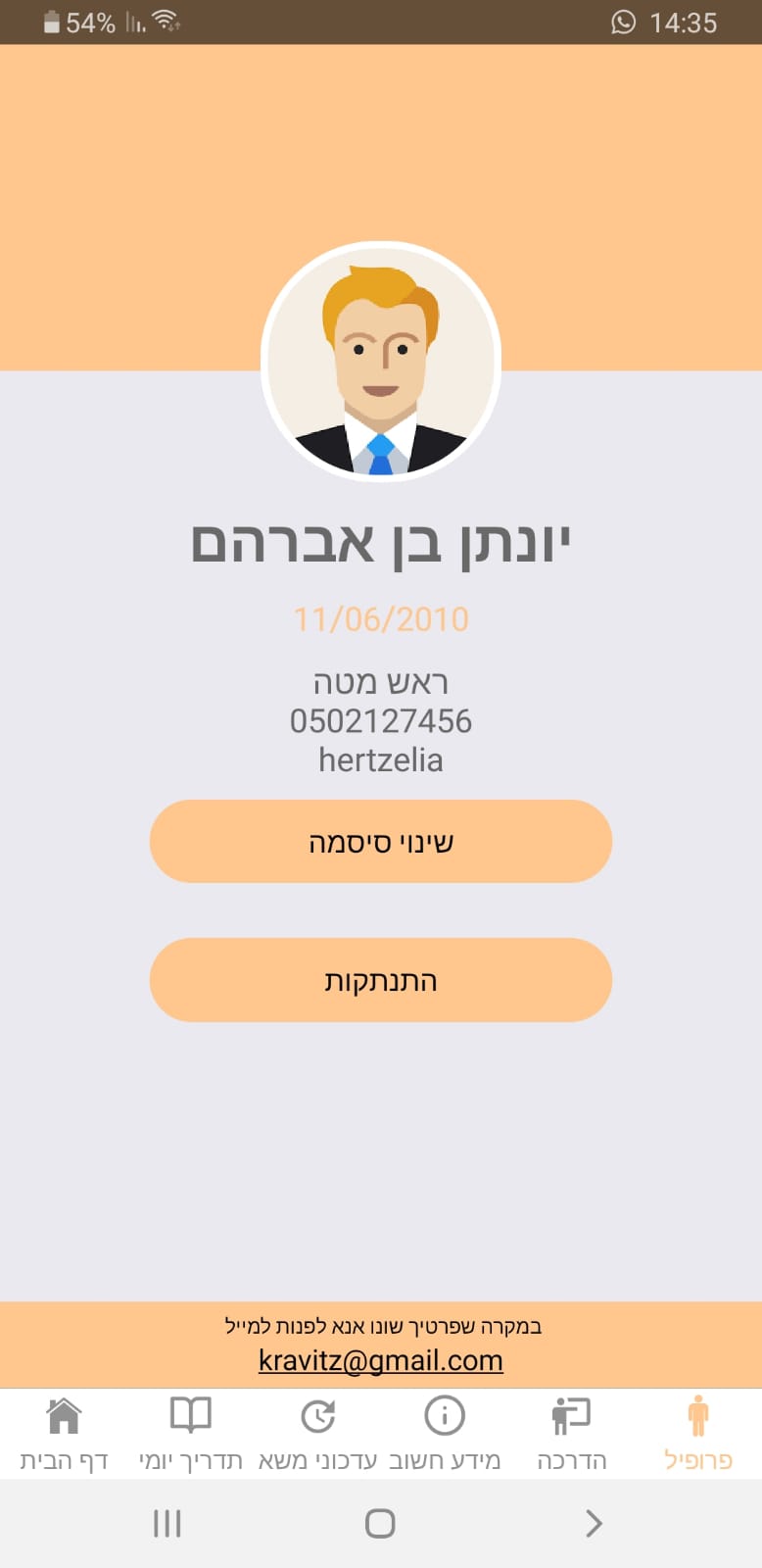
קרביץ מפרסמת הודעות ותדריכים והעובדים יכולים לאשר את קריאתם ובכך העובדים תמיד מעודכנים והמנהלים מודעים לידיעה של העובדים.

תמונת מתוך האפליקציה:









**תוצרי הפרויקט**

כאשר המשתמש נכנס לאפליקציה הוא מבצע log-in באמצעות תעודת זהות וסיסמה .

לאחר מכן מוצגות למשתמש כותרות הקטגוריות ותמונותיהן בהתאם.

באפשרות המשתמש להיכנס לתדריכים יומיים, להודעות שנשלחו, לימי הולדת , למידע חשוב, להדרכה ולפרופיל.

המערכת מוחקת לבד הודעות לאחר שלושה ימים ממועד הפרסום שלהן ואחרי כשבועיים תדריכים יומיים.

כאשר המשתמש יוצא מהאפליקציה ולא מתנתק או שהוא כבר לא עובד בקרביץ הוא לא צריך לבצע כניסה

חוזרת עם התעודת זהות.

התוצר הסופי של הפרויקט הוא לבסוף אפליקציית המשתמש אשר מאחוריה בצד השרת – עומדת מערכת מותאמת לעובדים ולמנהלים ככה שלכל אחד מהם יש שליטה על תכנים שונים ואת הפריווילגיה לבצע פעולות שונות.