# מסמך אפיון אפליקציה לשיתוף תמונות

כרגע את השרת וגם את האתר רשת והאפליקציה אנחנו יוצרים בצורה מקומית

(אולי:

אחרי שנסיים ליצור אותם אנחנו נעבור ל PRODUCTION בו נשתמש באתר <https://railway.com/> לצורך יצירת כתובות URL אמיתיות עבור השרת והאפליקציה)

את האפליקציית רשת אנו יוצרים עם react Vite

את אפליקציית מובייל אנו יוצרים עם expo

שיפורים :

* הוספת מסך הודעת באתר ובאפליקציה
* אפשרות להוריד תמונה פומבית ולא חסומה למכשיר או לאתר cloudery אם יש למשתמש חשבון באתר

(אפשרות שליחת הודעות בין משתמשים)

תוכן

[מסמך אפיון אפליקציה לשיתוף תמונות 1](#_Toc204548362)

[בסיס נתונים: 2](#_Toc204548363)

[שרת: 3](#_Toc204548364)

[אתר אינטרנט/מובייל: 5](#_Toc204548365)

[תיאור: 5](#_Toc204548366)

[עמודים כללים וapi: 5](#_Toc204548367)

[העמודים/המסכים: 5](#_Toc204548368)

[תוספות: 7](#_Toc204548369)

[עץ הקבצים: 8](#_Toc204548370)

[קצת על המודלים: 10](#_Toc204548371)

## בסיס נתונים:

ישנם 5 טבלאות בבסיס הנתונים:

טבלת משתמשים:

השדות: מספר מזהה(מפתח), שם, אי-מייל(ייחודי), סיסמה מוצפנת, תפקיד, האם חסום, תאריך יצירה

טבלת תמונות:

השדות: מספר מזהה(מפתח), מספר מזהה של משתמש(מפתח זר), שם, כתובת url, האם כללי, האם חסום, תאריך יצירה, תאריך עדכון, מספר מזהה של אלבום(מפתח זר) יכול להיות null

טבלת תגובות:

השדות: מספר מזהה(מפתח), מספר מזהה של תמונה(מפתח זר), מספר מזהה של משתמש(מפתח זר), תגובה, דירוג(1-5), תאריך יצירה, תאריך עדכון

טבלת אלבומים:

השדות: מספר מזהה(מפתח), שם האלבום, מספר מזהה של המשתמש(מפתח זר), האם פומבי, האם חסום, תאריך יצירה

טבלת התראות:

השדות: מספר מזהה, מספר מזהה של המשתמש(מפתח זר), הודעה ,תאריך יצירה, האם נקרא

## שרת:

Base URL = [http:/ 10.0.0.18:8000](http://localhost:8000)

הפונקציות של קבלת עמוד וקבלת תגובות צריכות לכלול גם את המספר עמוד כשאילתה בurl ברירת המחדל היא עמוד 1 ומקסימום 25 לעמוד ,הפונקציה חיפוש מקבלת את המילת חיפוש בשאילתה ומחזירה רק את הפריטים הפומביים

|  |  |  |  |
| --- | --- | --- | --- |
| **מסלול** | **סוג בקשה** | **הרשאה** | **תיאור** |
|  |  |  |  |
| ***Base URL/auth*** |  |  |  |
| /register | POST | - | רישום למערכת  צרך להזין שם, אי-מייל וסיסמה |
| /login | POST | - | כניסה למערכת צריך להזין אי-מייל וסיסמה  מוחזר טוקן, מזהה משתמש, תפקיד |
|  |  |  |  |
| ***Base URL/users*** |  |  |  |
| / | GET | מנהל | קבלת כל המשתמשים במערכת |
| /me | GET | המשתמש עצמו | קבלת כל הפרטים על המשתמש |
| /me/name | PUT | המשתמש עצמו | עריכת השם משתמש |
| /block | PUT | מנהל | חסימת/שיחרור משתמש והוספת הודעה |
| /me | DELETE | המשתמש עצמו | מחיקת המשתמש |
| /notifications/{user\_id}/not\_read | GET | - | קבלת כל ההודעת במערכת ששייכים למשתמש שלא נקראו |
| /notifications/{user\_id} | GET | - | קבלת כל ההודעת במערכת ששיכים למשתמש |
| /notifications/{notification\_id}/read | PUT | - | סימון הודעה כנקראה |
|  |  |  |  |
| ***Base URL/images*** |  |  |  |
| /addImage | POST | המשתמש | הוספת תמונה צריך להזין שם תמונה וכתובן URL |
| /AIgenerate | POST | המשתמש | הוספת תמונה שנוצרת על ידי AI צריך להזין שם תמונה ותיאור של התמונה(הAI בונה את התמונה בהתאם לתיאור) |
| /upload\_image\_file | POST | המשתמש | הוספת תמונה כקובץ  צריך להזין שם מתמונה ואת הקובץ |
| /me | GET | המשתמש | קבלת שם וurl על כל התמונות של המשתמש |
| /public | GET | - | קבלת שם, url ,תאריך עדכון ושם יוצר התמונה על התמונות שהם פומביות ולא חסומות |
| /{image\_id} | GET | המשתמש | קבלת כל הפרטים על תמונה מסוימת לפי id אם התמונה שלך |
| /images/search | GET | - | קבלת תמונות לפי חיפוש |
| /{image\_id}/rating | GET | - | קבלת ממוצע תגובות של תמונה |
| /{image\_id}/rename | PUT | המשתמש | ערכית שם התמונה |
| /{image\_id}/block | PUT | מנהל | חסימת תמונה |
| /{image\_id}/public | PUT | המשתמש | שינוי סטטוס האם פומבי |
| /{image\_id}/assign | PUT | המשתמש | הכנסת תמונה לאלבום |
| /{image\_id} | DELETE | המשתמש | מחיקת תמונה |
| /images/blocked | GET | מנהל | מקבל את התמונות הפומביות החסומות |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| ***Base URL/commits*** |  |  |  |
| / | GET | מנהל | קבלת כל התגובות על כל התמונות |
| /{image\_id} | GET | - | קבלת כל התגובות על תמונה מסוימת |
| /{commit\_id} | GET | המשתמש | קבלת פרטי תגובה |
| /{image\_id} | POST | המשתמש | הוספת תגובה ודירוג על תמונה מסוימת |
| /{commit\_id} | PUT | המשתמש | ערכית תגובה או דירוג של המשתמש עצמו |
| /{commit\_id} | DELETE | המשתמש או מנהל | מחיקת תגובה ודירוג של המשתמש |
|  |  |  |  |
| ***Base URL/albums*** |  |  |  |
| / | POST | המשתמש | יצירת אלבום |
| /public | GET | - | קבלת האלבומים הפומביים |
| /search | GET | - | קבלת אלבומים לפי חיפוש |
| /me | GRT | המשתמש | קבלת כל האלבומים של המשתמש |
| /{album\_id}/images/public | GET | - | קבלת כל התמונות הפומביות של האלבום |
| /{album\_id}/images | GET | המשתמש | קבלת כל התמונות באלבום של המשתמש |
| /{album\_id} | PUT | המשתמש | עדכון שם אלבום או האם פומבי |
| /{album\_id}/block | PUT | מנהל | חסימת אלבום |
| /{album\_id} | DELETE | המשתמש | מחיקת אלבום |

## אתר אינטרנט/מובייל:

### תיאור:

### עמודים כללים וapi:

app : web – מנהל את כל המסלולי גישה לעמודים בתוך האפליקציה

מובייל : \_layout.tsx – משמש כמנתב בין העמודים

api – תיקיה אשר הקבצים בה אחראים על התקשורת עם השרת (המסלולים גישה לשרת הכנסת header או data אם נצרך וכו') (באפליקציית המובייל כיוון שאין גישה לLS אנחנו משתמשים ב AsyncStorage ואנו דואגים להכניס את הפרטים ל AsyncStorage גם כן דרך הapi כאשר משתמש מתנתק פונקציה מתוך התיקייה מוחקת את הנתונים מהזיכרון)

### העמודים/המסכים:

אם אין משתמש מחובר נותנים גישה רק לעמודים :

- - - עמוד כניסה – שדות עבור הזנת מייל וסיסמה וכפתור התחברות (אם התחברנו בהצלחה אנו מועברים לעמוד הבית)

- - - עמוד רישום - שדות עבור הזנת שם משתמש, מייל וסיסמה וכפתור הרשמה (אם נרשמנו בהצלחה מועברים לעמוד כניסה)

- - - עמוד הבית – מציג את על התמונות הפומביות וכפתור חיפוש לפי שם (אם יש משתמש מחובר לחיצה על תמונה תעביר את המשתמש לעמוד פרטי תמונה אם המשתמש הוא מנהל יש לו גם כפתור חסימת תמונה)

- - - עמוד אלבומים פומביים – מציג את כל האלבומים הפומביים אם לוחצים על אחד האלבומים מועברים לעמוד פרטי אלבום (אם יש מנהל מחובר יש גם כפתור לחסימת אלבום) וכפתור חיפוש לפי שם אלבום

- - - עמוד פרטי אלבום – אם האלבום שייך למשתמש מוצגים על התמונות של המשתמש שקשורות לאלבום אם האלבום לא שייך למשתמש מוצגות רק התמונות הפומביות ששייכות לאלבום לחיצה על תמונה תציג את התמונה בתצוגה גדולה יותר

אם יש משתמש מחובר שאינו מנהל נתונים גישה ל:

- - - עמוד הבית – מציג את על התמונות הפומביות וכפתור חיפוש לפי שם (אם יש משתמש מחובר לחיצה על תמונה תעביר את המשתמש לעמוד פרטי תמונה אם המשתמש הוא מנהל יש לו גם כפתור חסימת תמונה)

- עמוד הודעות – מציג את כל ההודעת של המשתמש או רק הודעות שלא נקראה ואם לוחצים על הודעה היא הופכת לנקראה

- - -עמוד אלבומים פומביים – מציג את כל האלבומים הפומביים אם לוחצים על אחד האלבומים מועברים לעמוד פרטי אלבום (אם יש מנהל מחובר יש גם כפתור לחסימת אלבום) וכפתור חיפוש לפי שם אלבום

- - -עמוד פרטי אלבום – אם האלבום שייך למשתמש מוצגים על התמונות של המשתמש שקשורות לאלבום אם האלבום לא שייך למשתמש מוצגות רק התמונות הפומביות ששייכות לאלבום לחיצה על תמונה תציג את התמונה בתצוגה גדולה יותר

- - -עמוד פרטי תמונה – מציג את פרטי התמונה ,את התגובות על התמונה וכפתור להוספת תגובה (אם לחצים על הוספת תגובה מועברים לעמוד הוספת תגובה) בנוסף עבור תגובה של המשתמש יש 2 כפתורים : כפתור מחק שמוחק את התגובה וכפתור עריכה (לחיצה על כפתור עריכה מעביר את המשתמש לעמוד עריכת תגובה)  
אם התמונה חסומה מתחת לכפתור של הוספת תגובה תופיע הודעה שהתמונה חסומה

- - -עמוד הוספת תגובה – שדה בו מכניסים את התגובה ורשימה נפתחת של דירוג (1-5 כוכבים) כפתור ביטול שמחזיר את המשתמש לעמוד פרטי התמונה וכפתור הוספה שמוסיף את התגובה ומחזיר את המשתמש לעמוד פרטי תמונה

- - - עמוד עריכת תגובה - שדה בו מכניסים את התגובה (כברירת מחדל מוצגת התגובה שהייתה) ורשימה נפתחת של דירוג (1-5 כוכבים) (כברירת מחדל מוצג הדירוג שהיה) כפתור ביטול שמחזיר את המשתמש לעמוד פרטי התמונה וכפתור עדכון שמעדכן את התגובה ומחזיר את המשתמש לעמוד פרטי תמונה

- - - עמוד הפרופיל שלי – מציג את פרטי המשתמש שדה להזנת שם משתמש חדש כפתור לעדכון שם המשתמש וכפתור מחיקת משתמש

- - - עמוד התמונות שלי – מציג את כל התמונות של המשתמש לחיצה על תמונה מעבירה את המשתמש לעמוד פרטי התמונה בנוסף לכל תמונה יש כפתור מחיקה שמוחק את התמונה כפתור פומבי שמשנה את הסטטוס של התמונה (פומבי או פרטי) רשימה נפתחת בה ניתן לבחור לאיזה אלבום רוצים לשייך את התמונה (ברירת מחדל האלבום אליו שייכת התמונה או אלבום כללים) וכפתור שינוי שם לחיצה עליו מקפיצה התראה בה המשתמש מתבקש להזין שם חדש לתמונה  
מתחת לכל תמונה חסומה מופיעה הודעה שהתמונה חסומה

- - - עמוד הוספת תמונה – שדה להזנת של התמונה   
אפשרות להעלות קובץ תמונה וכפתור העלאת קובץ שמוסיף את התמונה למשתמש   
אפשרות להזין קישור URL של תמונה וכפתור הוספה מקישור  
אפשרות להזין תיאור של התמונה(באנגלית) וכפתור יצירת תמונה בעזרת AI כאשר התמונה נוצרת התמונה מוצגת למשתמש

- - - עמוד האלבומים שלי – מציג את כל האלבומים של המשתמש לחציה על אלבום תעביר את המשתמש לעמוד פרטי האלבום לחיצה על כפתור ערכיה של אלבום תעביר את המשתמש לעמוד עריכת אלבום וכפתור מחיקת אלבום מוחק את האלבום בנוסף יש כפתור יצירת אלבום לחיצה עליו תעביר את המשתמש לעמוד יצירת אלבום  
ליד כל אלבום חסום מופיע הודעה שהאלבום חסום

- - - עמוד יצירת אלבום – שדה להזנת שם האלבום כפתור לקביעה אם האלבום יהיה פומבי כפתור חזרה שמחזיר את המשתמש לעמוד האלבומים שלי וכפתור יצירת אלבום כאשר את הלוחץ עליו מוצגות התמונות של המשתמש וניתן לבחור תמונות להוסיף לאלבום וכפתור סיום שמעביר את המשתמש לעמוד האלבומים שלי

- - - עמוד עריכת אלבום – שדה להזנת שם האלבום (כברירת מחדל מוצג השם הקיים) כפתור שינוי סטטוס פומבי (כברירת מחדל הסטטוס הקיים) כפתור ביטול שמעביר את המשתמש לעמוד האלבומים שלי וכפתור עדכון שמעדכן את פרטי האלבום ומעביר את המשתמש לעמוד האלבומים שלי

מנהל מקבל גישה גם לעמודים:

- - - ניהול משתמשים (עבור מנהל) – מציר טבלה ובה כל הפרטים על המשתמשים וכן כפתור לחסימת משתמש

- - - ניהול תגובות (עבור מנהל) – מציג טבלה ובה כל התגובות של המשתמשים וכן כפתור מחיקת תגובה

- - - ניהול תמונות חסומות (עבור מנהל) – מציג את כל התמונות הפומביות החסומות וכן כפתור לשחרר אותם

עבור web :

- - - סרגל ניווט אורח – הרשמה, התחברות, אלבומים, בית

- - - סרגל ניווט משתמש – בית, התמונות שלי, הוספת תמונה, הפרופיל שלי, כל האלבומים, האלבומים שלי, יצירת אלבום,(עבור מנהל ניהול משתמשים וניהול תגובות) וכפתור התנתקות

משתמש לא מחובר:

כאשר אנחנו פותחים את אתר האינטרנט

אנו מגעים לעמוד הבית ויש סרגל ניווט לאורח

משתמש מחובר:

כאשר המשתמש מתחבר מוצג סרגל ניווט למשתמש והוא מועבר לעמוד הבית

כאשר המשתמש מתנתק הסרגל ניווט משתנה לסרגל ניווט לאורח והוא מעבר לעמוד הבית

## תוספות:

קבצי util

auth\_utils.py – משתמש של bcrypt ו JWT לצורך האימות של הסיסמה שהמשתמש הזין מול הhash של הסיסמה ששמור בבסיס הנתונים

generate\_image.py – משתמש ב StableDiffusionPipeline מתוך diffuser לצור יצירת תמונה מטקטס

permissions.py – בודק הרשאות גישה של משתמשים לפונקציות בשרת

background\_jobs.py – בודק פעם ביום או כאשר השרת עולה האם יש משתמש שחסום יותר מ7 ימים ואם כן משחרר אותו

toxicity\_model.py – בודק פעם ביום האם יש הודעות פוגעניות ואם יש מוחק אותם (הבדיקה נעשית מול רשימה של מילים בעברית ומול מודל שבודק האם תגובה באנגלית היא פוגענית)

## עץ הקבצים:

AI-IMAGE/

├── MobileApp/ # אפליקציית מובייל - Expo + React Native

│ ├── api/

│ │ ├── albums.js

│ │ ├── api.js

│ │ ├── auth.js

│ │ ├── comments.js

│ │ ├── images.js

│ │ └── users.js

│ ├── app/

│ │ ├── (tabs-authenticated)/

│ │ │ ├── \_layout.tsx

│ │ │ ├── add-image.tsx

│ │ │ ├── admin-comments.tsx

│ │ │ ├── admin-users.tsx

│ │ │ ├── albums.tsx

│ │ │ ├── blocked-images.tsx

│ │ │ ├── create-album.tsx

│ │ │ ├── index.tsx

│ │ │ ├── my-albums.tsx

│ │ │ ├── my-images.tsx

│ │ │ └── profile.tsx

│ │ ├── (tabs-guest)/

│ │ │ ├── \_layout.tsx

│ │ │ ├── albums.tsx

│ │ │ ├── index.tsx

│ │ │ ├── login.tsx

│ │ │ └── register.tsx

│ │ ├── albums/

│ │ │ ├── edit/

│ │ │ │ └── [id].tsx

│ │ │ └── [id].tsx

│ │ ├── comment/

│ │ │ ├── add/

│ │ │ │ └── [id].tsx

│ │ │ └── edit/

│ │ │ └── [id].tsx

│ │ ├── image/

│ │ │ └── [id].tsx

│ │ ├── \_layout.tsx

│ │ └── +not-found.tsx

│ └── README

│ └── ...

│

├── net-app/ # אפליקציית Web - React + Vite

│ ├── src/

│ │ ├── api/

│ │ │ ├── api.js

│ │ │ ├── auth.js

│ │ │ ├── users.js

│ │ │ ├── images.js

│ │ │ ├── comments.js

│ │ │ └── albums.js

│ │ ├── components/

│ │ │ ├── NavbarGuest.jsx

│ │ │ └── NavbarUser.jsx

│ │ ├── pages/

│ │ │ ├── albums/

│ │ │ │ ├── AlbumDetails.jsx

│ │ │ │ ├── CreateAlbum.jsx

│ │ │ │ ├── EditAlbum.jsx

│ │ │ │ ├── MyAlbums.jsx

│ │ │ │ └── PublicAlbums.jsx

│ │ │ ├── auth/

│ │ │ │ ├── Login.jsx

│ │ │ │ └── Register.jsx

│ │ │ ├── commits/

│ │ │ │ ├── AddComment.jsx

│ │ │ │ ├── AdminComments.jsx

│ │ │ │ └── EditComment.jsx

│ │ │ ├── images/

│ │ │ │ ├── AddImage.jsx

│ │ │ │ ├── Home.jsx

│ │ │ │ ├── ImageDetails.jsx

│ │ │ │ └── MyImages.jsx

│ │ │ └── users/

│ │ │ ├── AdminUsers.jsx

│ │ │ └── MyProfile.jsx

│ │ ├── App.jsx

│ │ └── main.jsx

│ │── index.html

│ │── README

│ └── ...

│

├── server/ # צד שרת - FastAPI

│ ├── controllers/

│ │ ├── albums.py

│ │ ├── auth.py

│ │ ├── commits.py

│ │ ├── images.py

│ │ └── users.py

│ ├── data/

│ │ └── toxic\_words.txt

│ ├── static/

│ ├── utils/

│ │ ├── auth\_utils.py

│ │ ├── background\_jobs.py

│ │ ├── generate-image.py

│ │ ├── hebrew\_toxic\_words.py

│ │ ├── toxicity\_model.py

│ │ └── promissions.py

│ ├── .env

│ ├── database.py

│ ├── main.py

│ ├── README

│ └── requirements.txt

│

├── create-tables.sql # קובץ יצירת טבלאות למסד הנתונים

├── README

## קצת על המודלים:

יצירת תמונה:

אנו משתמשים ב StableDiffusionPipeline.from\_pretrained("runwayml/stable-diffusion-v1-5") לצורך יצרת תמונה

מה זה עושה:

מודל Stable Diffusion מקבל תיאור טקסטואלי (prompt) כמו "a cat in space" ויוצר תמונה שתואמת לתיאור הזה.

איך זה עובד:

1. Text Encoder מבוסס CLIP או (T5)

הטקסט עובר קידוד ע"י מודל NLP כמו CLIP) או (T5 ויוצרת וקטור (embedding) שמייצג את המשמעות של הטקסט.

1. Latent Diffusion Model

* המודל מתחיל מרעש גאוסי (כמו תמונה מלאה ברעש) ומנסה "למחוק את הרעש" כדי לחשוף את התמונה שמתאימה ל־prompt.
* התהליך נקרא denoising diffusion: בכל שלב הוא משפר מעט את התמונה לפי כיוון המשמעות של הטקסט.

1. Decoder (VAE):

* המודל יוצר את התמונה בתור latent space (מרחב חבוי – לא RGB רגיל)
* ב־סיום הוא ממיר את זה חזרה לתמונה נראית לעין (RGB) בעזרת Autoencoder .

🖼️ התוצאה – תמונה באיכות גבוהה עם פרטים שמבוססים על הטקסט שהוזן.

זיהוי תגובות פוגעניות:

אנחנו משתמשים בpipeline("text-classification", model="unitary/toxic-bert") לצורך הזיהוי

מה זה עושה:

מקבל טקסט כמו "You're so stupid and useless"ומחזיר

[ { {"label": "TOXIC", "score": 0.98 ]

איך זה עובד מאחורי הקלעים?

1. BERT:

המודל מתבסס על BERT (Bidirectional Encoder Representations from Transformers).

BERT קורא את המשפט מכל הכיוונים ומפיק ייצוג עשיר של המשמעות של כל מילה בהקשר שלה.

2. Fine-Tuning לסיווג טקסטים פוגעניים:

המודל אומן מחדש (fine-tuned) על דאטה סטים של טקסטים מתויגים לפי רמות טוקסיות (למשל Jigsaw Toxic Comments).

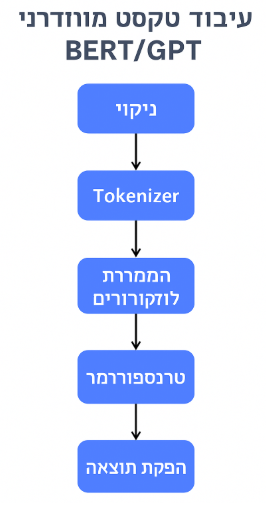
הוא למד להבדיל בין טקסטים רגילים לפוגעניים, כולל קללות, גזענות, מיזוגניה, שנאה וכו'.

3. Classifier:

לוקטור הסופי של המשפט (ה־[CLS] token) מחוברת שכבה קטנה של סיווג.

היא מחזירה תוצאה של TOXIC / NON\_TOXIC + score (רמת הביטחון).

צירוף של תמונה ויזואלית לצורך הבנה של איך עובד מודל NLP :



* ניקוי (Preprocessing) – מנקים את הטקסט מאימוג'ים וסימנים שלא מועילים
* פיצול למילים (Tokenization) –
* ב־NLP קלאסי: פיצול לפי רווחים או Regex.
* ב־BERT: טוקניזציה תת-מילית (WordPiece או Byte-Pair Encoding) – לדוגמה: "walking" → "walk", "##ing"
* (החזרה למילה בסיסית (Stemming/Lemmatization) –
* משתמשים בזה רק ב־NLP קלאסי.
* מודלים כמו BERT לא צריכים סטמינג , כי הם לומדים את הקשרים מהמון הקשרים אפשריים של המילים עצמן.)
* הפיכת מילים לווקטורים (Embedding) –
* כל מילה או תת־מילה מומרת לוקטור מספרי באמצעות embedding matrix.
* במודלים כמו GPT/BERT, זה חלק פנימי מהמודל – נקרא input embeddings.
* שימוש במודל עצמו (Transformer ) –

תלוי במשימה:

* לסיווג → Classification head
* לזיהוי ישויות → Tagging head
* ליצירת טקסט/תמונה → Decoder או StableDiffusion
* (השוואת טקסטים / זיהוי משמעות / החלטה)
* הפקת תוצאות