פרויקט גמר/ מסמך אפיון

**מאגר תמונות לצלמים**

# תיאור המערכת

## מטרת הפרויקט:

הפרויקט נועד לספק לצלמים מערכת אינטואיטיבית ויעילה לניהול תמונות.

המערכת מאפשרת העלאת תמונות, ארגון באלבומים וגיבוי אוטומטי בענן, הורדה ומחיקה. בנוסף היא מאפשרת סינון חכם תוך שימוש בטכנולוגיות AI לזיהוי אוטומטי של מאפיינים כמו איכות התמונה/ מס' האנשים בתמונה/ מי האנשים בתמונה/ תמונות חוץ-פנים וכד'.

המערכת מאפשרת גם עיצוב קולאז'ים (מתוך תבניות מוכנות? / ע"י AI) והורדה שלהם בפורמט PDF

מטרת האפליקציה היא לייעל את תהליך העבודה של צלמים ולאפשר להם לנהל את התמונות שלהם בצורה מסודרת ויעלה.

## קהל היעד:

המערכת מיועדת לצלמים מקצועיים וחובבים המחפשים דרך נוחה לנהל את התמונות שלהם. (היא מתאימה גם למעצבים גרפיים, סטודיו לצילום, ואפילו לאנשים פרטיים המעוניינים בארגון חכם של אלבומי התמונות האישיים שלהם.)

## פונקציונליות עיקרית:

* **העלאת תמונות**: אפשרות להעלאה ישירה מהמחשב או באמצעות גרירה (Drag & Drop).
* **הורדה מאורגנת**: הורדת כל התמונות בתיקייה כקובץ ZIP.
* **מיון חכם**:
  + זיהוי תמונות מטושטשות. (מחיקה אוטומטית?)
  + זיהוי אוטומטי של תמונות חוץ/פנים.
  + זיהוי מספר האנשים בתמונה.
  + חיפוש תמונות לפי תגים?
  + לבדוק אם אדם מסוים מופיע בתמונה (בהשוואה לתמונה קודמת)
  + שיפור איכות תמונה?
* עיצוב קולאז'ים??? אם יהיה זמן:
  + או-יצירת קולאז' חכם- ע"י AI.
  + או- עיצוב קולאז' מתוך תבניות
  + הורדת הקולאז' כקובץ PDF.
* ניהול תיקיות: יצירה, עריכה ומחיקה של תיקיות ואלבומים.

## בעיות שהמערכת פותרת:

האפליקציה פותרת את הבעיה של ניהול תמונות מבולגן ומפוזר בין מכשירים שונים ומאירועים שונים. היא מאפשרת גישה נוחה לכל התמונות ממקום אחד ומבטיחה גיבוי אוטומטי כדי למנוע אובדן של תמונות יקרות.

המערכת גם פותרת את הבעיה של סינון תמונות ובכך חוסכת עבודה של שעות לצלמים.

## טכנולוגיות בשימוש:

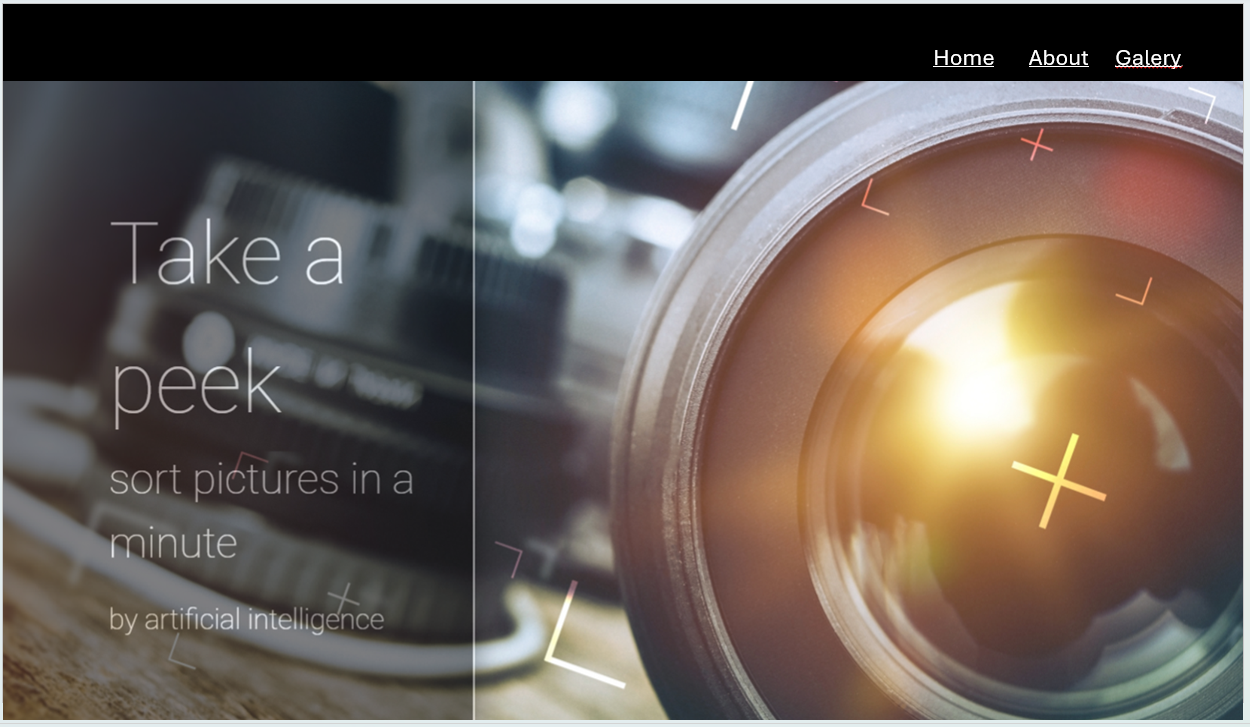
הפרויקט יפותח באמצעות הטכנולוגיות הבאות:

* **צד שרת (Backend):** .NET 9לבניית ה-API.
* **צד לקוח (Frontend):** React.js לבניית ממשק משתמש אינטראקטיבי.
* **מסד נתונים:** PosgreSQL לאחסון נתונים של משתמשים ותמונות.
* **אחסון קבצים:** שימוש ב-Amazon S3 לאחסון תמונות.
* **AI לעיבוד תמונה**: שימוש ב-Library מתאים כמו OpenCV לזיהוי מאפיינים בתמונות.

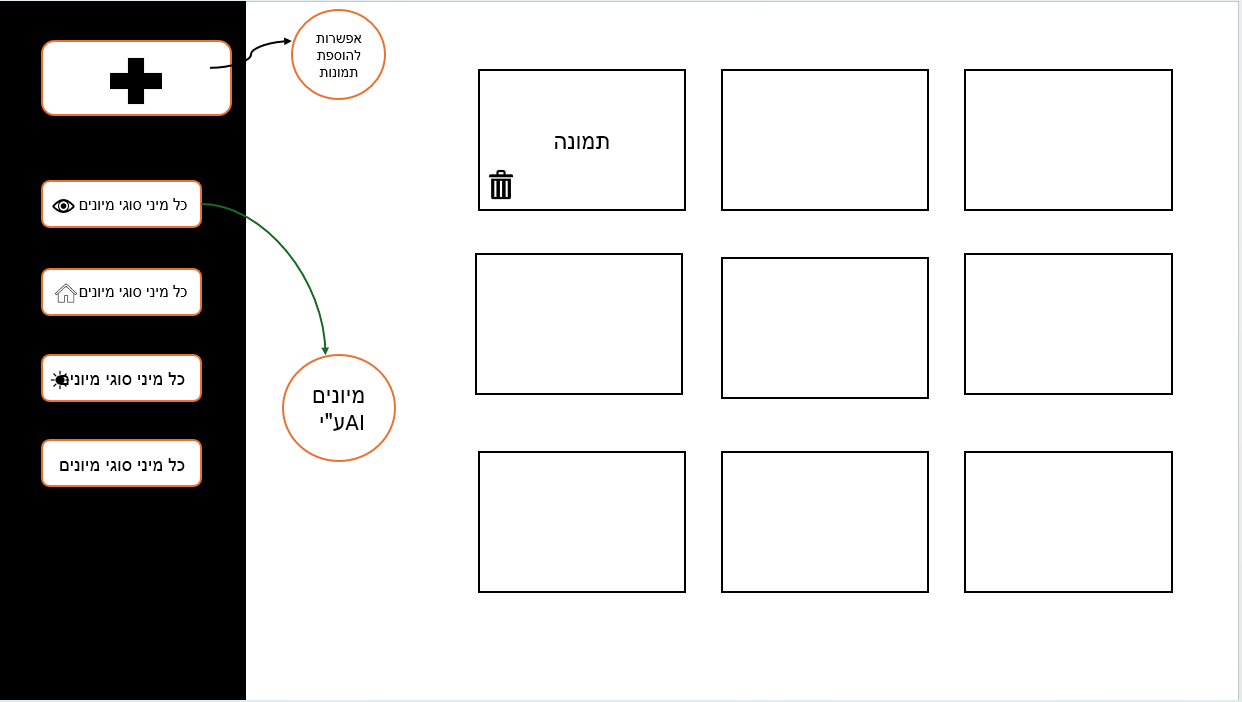
## מתודולוגיית פיתוח (Agile):

# עיצוב ממשק משתמש ויזואלי

דף הבית- כניסה:



דף הגלריה- מבפנים/ בתוך תיקיה:



# אפיון תמציתי

## רשימת פונקציות המערכת

**העלאת קובץ**

* **Route:** POST /api/upload
* **פרמטרים:**
  + **Body:** { file: File, tags: string[] }
* **פלט:**
  + data: { fileId: string, url: string }
* **לוגיקה עסקית:**
  + הגבלת גודל קובץ ל-50MB
  + הוספת חתימת זמן לקובץ
  + שמירה ב-S3

**הורדת קבצים**

* **Route:** GET /api/download/{folderId}
* **פרמטרים:**
  + **Path:** { folderId: string }
* **פלט:**
  + קובץ ZIP להורדה
* **לוגיקה עסקית:**
  + יצירת ZIP של כל התמונות בתיקייה
  + שליחת לינק להורדה

**סינון תמונות**

* **Route:** POST /api/filter
* **פרמטרים:**
  + **Body:** { filterType: string, fileIds: string[] }
* **פלט:**
  + data: { filteredFiles: File[] }
* **לוגיקה עסקית:**
  + שימוש ב-AI לזיהוי תמונות מטושטשות, פנים/חוץ, מספר אנשים

**יצירת תיקייה חדשה**

* **Route:** POST /api/folders
* **פרמטרים:**
  + **Body:** { name: string, parentId?: string }
* **פלט:**
  + data: { folderId: string, name: string }
* **לוגיקה עסקית:**
  + בדיקה אם שם התיקייה כבר קיים
  + יצירה במסד הנתונים

**מחיקת תמונה**

* **Route:** DELETE /api/files/{fileId}
* **פרמטרים:**
  + **Path:** { fileId: string }
* **פלט:**
  + status: 204 No Content
* **לוגיקה עסקית:**
  + בדיקה אם הקובץ קיים
  + מחיקה מ-S3 וממסד הנתונים

**יצירת קולאז’**

* **Route:** POST /api/collage
* **פרמטרים:**
  + **Body:** { fileIds: string[], template: string }
* **פלט:**
  + data: { collageUrl: string }
* **לוגיקה עסקית:**
  + שילוב תמונות לפי תבנית
  + ייצוא כ-PDF

## אימות והרשאות

* **מנגנון אימות:**
  + המערכת משתמשת ב-JWT Authentication.
  + המשתמש מתחבר עם מייל/סיסמה ומקבל JWT Token.
* **תפקידים:**
  + **צלם רגיל:** יכול להעלות, להוריד ולנהל את הקבצים שלו בלבד.
  + **מנהל:** יכול לנהל משתמשים, לראות דוחות, להגדיר הרשאות ומגבלות אחסון.

## אפליקציית ניהול

**מסכים עיקריים**

1. **מסך התחברות** – אימות באמצעות JWT
2. **ניהול משתמשים** – הוספה, עדכון, מחיקה, שינוי הרשאות
3. **דוחות פעילות** – נתונים סטטיסטיים על שימוש במערכת
4. **ניהול הגדרות מערכת** – ניהול מכסות אחסון, הרשאות גישה, מגבלות מערכת
5. **צפייה בלוגים** – מעקב אחרי פעולות במערכת (העלאות, מחיקות, כניסות וכו’)

**פונקציות עיקריות**

* **ניהול משתמשים:**
  + GET /api/admin/users – שליפת רשימת משתמשים
  + POST /api/admin/users – יצירת משתמש חדש
  + PUT /api/admin/users/{id} – עדכון משתמש
  + DELETE /api/admin/users/{id} – מחיקת משתמש
* **דוחות מערכת:**
  + GET /api/admin/reports – שליפת נתונים סטטיסטיים
* **ניהול הרשאות:**
  + PUT /api/admin/users/{id}/permissions – שינוי תפקידים והרשאות
* **ניהול הגדרות מערכת:**
  + PUT /api/admin/settings – עדכון פרמטרים כמו מגבלת אחסון או פורמט ברירת מחדל לקולאז’ים

# **(Database Schema)תרשים מבנה הטבלאות**

**טבלת Users**

|  |  |  |  |
| --- | --- | --- | --- |
| שם עמודה | סוג נתונים | מאפיינים | תיאור |
| id | INT | PRIMARY KEY | מזהה ייחודי |
| name | VARCHAR(50) | NOT NULL | שם |
| email | VARCHAR(100) | UNIQUE, NOT NULL | אימייל |
| password | VARCHAR(255) | NOT NULL | סיסמה מוצפנת |
| created\_at | DATETIME | DEFAULT NOW() | תאריך יצירה |
| updated\_at | DATETIME | ON UPDATE NOW() | תאריך עדכון |

**טבלת Files- img**

|  |  |  |  |
| --- | --- | --- | --- |
| שם עמודה | סוג נתונים | מאפיינים | תיאור |
| id | INT | PRIMARY KEY | מזהה ייחודי של הקובץ |
| user\_id | INT | FOREIGN KEY → Users(id) | בעל הקובץ |
| folder\_id | INT | FOREIGN KEY → Folders(id) | התיקייה שבה נמצא הקובץ |
| filename | VARCHAR(255) | NOT NULL | שם הקובץ |
| url | TEXT | NOT NULL | קישור לקובץ ב-S3 |
| size | INT | NOT NULL | גודל הקובץ (בבתים) |
| created\_at | DATETIME | DEFAULT NOW() | תאריך יצירה |
| updated\_at | DATETIME | ON UPDATE NOW() | תאריך עדכון |

**טבלת Folders**

|  |  |  |  |
| --- | --- | --- | --- |
| שם עמודה | סוג נתונים | מאפיינים | תיאור |
| id | INT | PRIMARY KEY | מזהה תיקייה |
| user\_id | INT | FOREIGN KEY → Users(id) | משתמש שיצר את התיקייה |
| parent\_id | INT | FOREIGN KEY → Folders(id), NULLABLE | תיקיית אב (אם יש) |
| name | VARCHAR(100) | NOT NULL | שם התיקייה |
| created\_at | DATETIME | DEFAULT NOW() | תאריך יצירה |
| updated\_at | DATETIME | ON UPDATE NOW() | תאריך עדכון |

**טבלת FileTags (טבלת חיבור Many-to-Many**

|  |  |  |  |
| --- | --- | --- | --- |
| שם עמודה | סוג נתונים | מאפיינים | תיאור |
| file\_id | INT | FOREIGN KEY → Files(id) | מזהה קובץ |
| tag\_id | INT | FOREIGN KEY → Tags(id) | מזהה תגית |

**טבלת Tags----**)--- אולי. כי עדיף מenum כי משתנה ומתווסף כל הזמן

|  |  |  |  |
| --- | --- | --- | --- |
| שם עמודה | סוג נתונים | מאפיינים | תיאור |
| id | INT | PRIMARY KEY | מזהה תגית |
| name | VARCHAR(50) | UNIQUE, NOT NULL | שם תגית |

**טבלת Collages אם צריך לשמור קולאז'ים במערכת**- כדאי לשמור או שישר אחרי שמוריד ימחק?

|  |  |  |  |
| --- | --- | --- | --- |
| שם עמודה | סוג נתונים | מאפיינים | תיאור |
| id | INT | PRIMARY KEY | מזהה קולאז' |
| user\_id | INT | FOREIGN KEY → Users(id) | בעל הקולאז' |
| template | VARCHAR(100) | NOT NULL | תבנית עיצוב שנבחרה |
| created\_at | DATETIME | DEFAULT NOW() | תאריך יצירה |

# הגדרת סבבי פיתוח (ספרינטים)

## ספרינט 1 (שבועות 1-2)

**מטרה:** הקמת סביבת עבודה ובסיס נתונים + תהליך התחברות והרשמה

**משימות:**

* פתיחת פרויקטים חדשים ב-.NET, React, Angular
* יצירת טבלאות במסד הנתונים (SQL Server/MySQL)
* פיתוח תהליך התחברות והרשמה ב-API
* פיתוח מסך התחברות והרשמה ב-React

## ספרינט 2 (שבועות 3-4)

**מטרה:** ניהול משתמשים וקבצים

**משימות:**

* פיתוח API לניהול משתמשים (CRUD)
* הוספת הרשאות ותפקידים (משתמש רגיל / מנהל)
* פיתוח API להעלאת קבצים ושמירתם ב-S3
* יצירת מסך ניהול קבצים (העלאה, מחיקה, הצגת רשימת קבצים)

## ספרינט 3 (שבועות 5-6)

**מטרה:** הוספת תיקיות וסינון תמונות

**משימות:**

* פיתוח API לניהול תיקיות (יצירה, מחיקה, ארגון היררכי)
* הוספת תמיכה בסינון תמונות לפי מאפיינים (פנים/חוץ, מספר אנשים)
* פיתוח מסך לניהול תיקיות
* שילוב אלגוריתם AI לזיהוי תמונות מטושטשות

## ספרינט 4 (שבועות 7-8)

**מטרה:** יצירת קולאז'ים והורדת ZIP

**משימות:**

* פיתוח API ליצירת קולאז'ים מתבניות מוכנות
* הוספת אפשרות להורדת תיקיות שלמות כ-ZIP
* פיתוח ממשק גרפי לבחירת תבנית קולאז'
* בדיקות וטיוב ביצועים בתהליכי ההורדה

## ספרינט 5 (שבועות 9-10)

**מטרה:** אבטחה, ניהול הרשאות ודוחות מערכת

**משימות:**

* הוספת JWT לאימות משתמשים
* הגבלת הרשאות לפי סוג משתמש (Admin / User)
* יצירת API להצגת דוחות פעילות (העלאות, מחיקות, שימוש באחסון)
* מסך דוחות למנהל עם נתונים סטטיסטיים

## ספרינט 6 (שבועות 11-12)

**מטרה:** בדיקות, שיפורים ופרסום מערכת

**משימות:**

* ביצוע בדיקות (Unit Tests + Integration Tests)
* שיפור חוויית משתמש (UI/UX)
* פרסום המערכת בסביבה חיה (Production)
* כתיבת תיעוד והדרכה לשימוש במערכת