**אפיון**

### 1. תיאור המערכת

**שם האפליקציה:**   
PhotoSmart

**מטרת הפרויקט:**  
האפליקציה נועדה לאפשר לבעלי האירוע לשלוח לאורחים את כל התמונות מהאירוע שבהן האורח המסוים מופיע. האפליקציה תאפשר העלאת תמונות, זיהוי פנים, תיוג, חיפוש מתקדם ושיתוף עם אורחים.

**קהל היעד:**  
קהל היעד כולל בעלי אירועים, צלמים וכל מי שמעוניין לשתף תמונות עם אורחים בצורה נוחה ויעילה.

**פונקציונליות של המערכת:**

1. העלאת תמונות לענן.
2. זיהוי פנים והקצאת תמונות לאורחים.
3. תיוג אוטומטי של תמונות.
4. חיפוש תמונות לפי תוויות ותאריכים.
5. שליחת תמונות לאורחים לפי זיהוי פנים.

**בעיות שהמערכת פותרת:**  
האפליקציה פותרת את הבעיה של שיתוף תמונות בצורה מסודרת, ומציעה גישה נוחה לכל התמונות שבהן האורח מופיע, מה שמפשט את תהליך השיתוף.

**טכנולוגיות בשימוש:**

1. **צד שרת (Backend):** .NET 9 לבניית ה-API.
2. **צד לקוח (Frontend):** React.js לבניית ממשק משתמש אינטראקטיבי.
3. **אפליקציית ניהול (Frontend):** Angular לניהול משתמשים ודוחות.
4. **מסד נתונים:** PostgreSQL לאחסון נתונים.
5. **אחסון קבצים:** Amazon S3 לאחסון תמונות.

**1.6 מתודולוגיית פיתוח (Agile)**

הפרויקט יפותח במתודולוגיית Agile עם חלוקה ל-3 ספרינטים בני שבוע כל אחד. כל ספרינט יתמקד במשימות ליבה עיקריות.

**2. עיצוב ממשק משתמש ויזואלי**

שרטוטים (wireframes) לכל מסך מרכזי, כולל מסך כניסה, מסך ניהול קבצים ומסך שליחת תמונות לאורחים.

**3. איפיון תמציתי**

**3.1 רשימת פונקציות המערכת**

1. **העלאת תמונות**
   * **Route:** POST /api/photos/upload
   * **פרמטרים:**
     + Headers: Authorization (Token)
     + Body: { images: [imageData], eventId: string }
   * **פלט:**
     + { success: boolean, message: string, photoIds: [string] }
   * **לוגיקה עסקית:**
     + אימות משתמש, שמירה של התמונות ב-Cloud Storage.
2. **זיהוי פנים והקצאת תמונות לאורחים**
   * **Route:** POST /api/photos/organize
   * **פרמטרים:**
     + Headers: Authorization (Token)
     + Body: { eventId: string, photoIds: [string], guestIds: [string] }
   * **פלט:**
     + { success: boolean, message: string }
   * **לוגיקה עסקית:**
     + זיהוי פנים והקצאת התמונות לאורחים.
3. **תיוג תמונות**
   * **Route:** POST /api/photos/tag
   * **פרמטרים:**
     + Headers: Authorization (Token)
     + Body: { photoId: string, tags: [string] }
   * **פלט:**
     + { success: boolean, message: string }
   * **לוגיקה עסקית:**
     + עדכון מסד הנתונים עם התיוגים.
4. **חיפוש תמונות**
   * **Route:** GET /api/photos/search
   * **פרמטרים:**
     + Query Params: { tags: [string], dateRange: { start: string, end: string } }
   * **פלט:**
     + { success: boolean, photos: [photoData] }
   * **לוגיקה עסקית:**
     + חיפוש במסד הנתונים.
5. **שליחת תמונות לאורחים**
   * **Route:** POST /api/photos/send
   * **פרמטרים:**
     + Headers: Authorization (Token)
     + Body: { personId: string, eventId: string }
   * **פלט:**
     + { success: boolean, message: string }
   * **לוגיקה עסקית:**
     + שליחת התמונות לאימייל או למערכת הודעות.

**3.2 אימות והרשאות**

האפליקציה מבצעת אימות באמצעות JWT Authentication. תפקידים כוללים:

* **משתמש רגיל:** גישה להעלאת תמונות, חיפוש ושיתוף.
* **מנהל:** גישה לניהול משתמשים, דוחות והגדרות מערכת.

**3.3 אפליקציית ניהול**

יכולות המנהל במערכת כוללות:

* **דוחות:** הפקת דוחות פעילות משתמשים וסטטיסטיקות שימוש.
* **ניהול משתמשים:** CRUD למשתמשים + שיוך תפקידים.
* **הרשאות:** עדכון הרשאות למשתמשים.
* **הגדרות מערכת:** ניהול פרמטרים גלובליים.

### 4. מבנה הטבלאות (Database Schema)

**טבלת Users**

| **שם העמודה** | **סוג נתונים** | **מאפיינים** | **תיאור** |
| --- | --- | --- | --- |
| id | SERIAL | PK | מזהה ייחודי |
| first\_name | VARCHAR(50) | NOT NULL | שם פרטי |
| last\_name | VARCHAR(50) | NOT NULL | שם משפחה |
| email | VARCHAR(100) | UNIQUE, NOT NULL | אימייל |
| password\_hash | VARCHAR(255) | NOT NULL | סיסמה מוצפנת |
| created\_at | TIMESTAMP | DEFAULT NOW() | תאריך יצירה |
| updated\_at | TIMESTAMP | DEFAULT NOW() | תאריך עדכון |
| role | VARCHAR(50) | DEFAULT 'user' | תפקיד (user/admin) |

**טבלת Photos**

| **שם העמודה** | **סוג נתונים** | **מאפיינים** | **תיאור** |
| --- | --- | --- | --- |
| id | SERIAL | PK | מזהה ייחודי |
| user\_id | INT | FK (Users.id) | מזהה המשתמש שהעלה |
| url | VARCHAR(255) | NOT NULL | כתובת התמונה |
| event\_id | INT | FK (Events.id) | מזהה האירוע |
| created\_at | TIMESTAMP | DEFAULT NOW() | תאריך יצירה |
| updated\_at | TIMESTAMP | DEFAULT NOW() | תאריך עדכון |

**טבלת Events**

| **שם העמודה** | **סוג נתונים** | **מאפיינים** | **תיאור** |
| --- | --- | --- | --- |
| id | SERIAL | PK | מזהה ייחודי |
| title | VARCHAR(100) | NOT NULL | שם האירוע |
| date | TIMESTAMP | NOT NULL | תאריך האירוע |
| location | VARCHAR(255) | NOT NULL | מיקום האירוע |
| created\_at | TIMESTAMP | DEFAULT NOW() | תאריך יצירה |
| updated\_at | TIMESTAMP | DEFAULT NOW() | תאריך עדכון |

**טבלת Albums**

| **שם העמודה** | **סוג נתונים** | **מאפיינים** | **תיאור** |
| --- | --- | --- | --- |
| id | SERIAL | PK | מזהה ייחודי |
| user\_id | INT | FK (Users.id) | מזהה המשתמש שיצר |
| title | VARCHAR(100) | NOT NULL | שם האלבום |
| description | TEXT |  | תיאור האלבום |
| created\_at | TIMESTAMP | DEFAULT NOW() | תאריך יצירה |
| updated\_at | TIMESTAMP | DEFAULT NOW() | תאריך עדכון |

**טבלת AlbumPhotos (טבלת חיבור)**

| **שם העמודה** | **סוג נתונים** | **מאפיינים** | **תיאור** |
| --- | --- | --- | --- |
| album\_id | INT | FK (Albums.id) | מזהה האלבום |
| photo\_id | INT | FK (Photos.id) | מזהה התמונה |

**קשרים בין הטבלאות**

* **Users (1) ↔ Photos (Many)**: משתמש אחד יכול להעלות מספר תמונות.
* **Users (1) ↔ Events (Many)**: משתמש אחד יכול ליצור מספר אירועים.
* **Photos (Many) ↔ Events (1)**: תמונה שייכת לאירוע אחד.
* **Users (1) ↔ Albums (Many)**: משתמש אחד יכול ליצור מספר אלבומים.
* **Albums (Many) ↔ Photos (Many)**: אלבום יכול להכיל מספר תמונות, ותמונה יכולה להיות בכמה אלבומים (רבים לרבים).

**5. הגדרת סבבי פיתוח (ספרינטים)**

**ספרינט 1 (שבוע 1)**

**מטרה:** הקמת סביבת עבודה ובסיס נתונים  
**משימות:**

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

**ספרינט 2 (שבוע 2)**

**מטרה:** פיתוח פונקציות ניהול תמונות ואלבומים  
**משימות:**

* פיתוח פונקציית העלאת תמונות ב-API.
* פיתוח פונקציה לזיהוי פנים והקצאת תמונות לאורחים ב-API.
* פיתוח אפליקציית ניהול ב-Angular שתכלול:
  + ניהול משתמשים (CRUD).
  + ניהול דוחות פעילות משתמשים.
* פיתוח מסך ניהול תמונות ואלבומים ב-React.

**ספרינט 3 (שבוע 3)**

**מטרה:** שיפוט ושיפור חוויית המשתמש  
**משימות:**

* הטמעת חיפוש תמונות לפי תוויות ב-API.
* ביצוע בדיקות אבטחה ואופטימיזציות ב-API.
* שיפור ממשק המשתמש ב-React וב-Angular .
* פיתוח פונקציה לשליחת תמונות לאורחים ב-API.