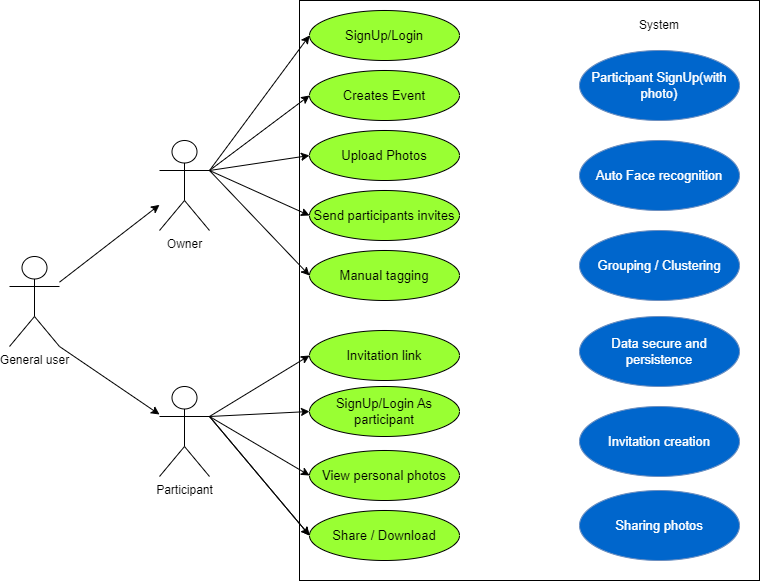
משתתפים :

* דורון בן זקן
* דור מור
* איתי רזון

**Use case Diagram**



* **סוגי משתמשים:**
  + בעלי האירוע (מארגנים).
  + משתתפים באירוע
  + לקוחות צד שלישי (בעלי עניין בתמונות).
* **שלבי התהליך:**
  + **בעל האירוע:**
    - פתיחת אירוע חדש
    - העלאת תמונות מהאירוע.
    - סימון משתתפים בתמונות באופן ידני (לא חובה – לאחר הזיהוי האוטומטי במערכת).
    - הגדרת הרשאות שיתוף.
  + **משתתף באירוע:**
    - קבלת גישה אישית לאירוע(קישור למערכת).
    - התחברות או הרשמה למערכת.
    - צפייה בתמונות המיועדות אלי.
    - אפשרות להורדה או שיתוף התמונות עם אחרים. (ממשק עם פלטפורמות נוספות)
  + **מערכת:**
    - זיהוי פנים בתמונות מתבצע מאחורי הקלעים בצורה אוטומטית לאחר העלאת התמונות.
    - מיון תמונות לפי משתתפים וקטלוגן לקבוצות רלוונטיות (אפשרות) .
    - שמירה ואבטחה של נתונים.
    - שליחת הודעה למשתתף עם לינק לגישה לתמונות.
    - אפשרות לשיתוף התמונות .

**Flow Diagram**

תמונה שמכילה טקסט, תרשים, צילום מסך, עיצוב

התיאור נוצר באופן אוטומטי

התרשים מתאר תהליך זרימה של שתי פעולות עיקריות: זרימת פעולות של בעל אירוע וזרימת פעולות של משתתף באירוע.

* **בעל אירוע**:
  + אם לבעל האירוע אין חשבון, הוא מתבקש להירשם, להזין פרטים, ולאמת את נכונותם.
  + במידה והפרטים תקינים, הוא נכנס למערכת. אם יש לו כבר חשבון, הוא עובר ישירות להתחברות.
  + לאחר ההתחברות, בעל האירוע יוצר אירוע חדש, מעלה תמונות, שולח הזמנות למשתתפים, והמערכת מתחילה בעיבוד.
  + שלבי העיבוד כוללים זיהוי פנים, סידור אוטומטי של התמונות לפי אנשים, ומתן אפשרות לתיוג ידני.
* **משתתף באירוע**:
  + המשתתף מקבל הזמנה לאירוע. אם אין לו חשבון, עליו להירשם, להזין פרטים, ולאמת אותם.
  + אם הפרטים תקינים, המשתתף מתחבר למערכת, ואם יש לו חשבון, הוא מתחבר ישירות.
  + לאחר ההתחברות, המשתתף יכול לצפות בתמונות, לבחור להוריד אותן או לשתף אותן עם אחרים.

**ERD (Entity–relationship diagram)**

**תמונה שמכילה טקסט, תרשים, תוכנית, קו

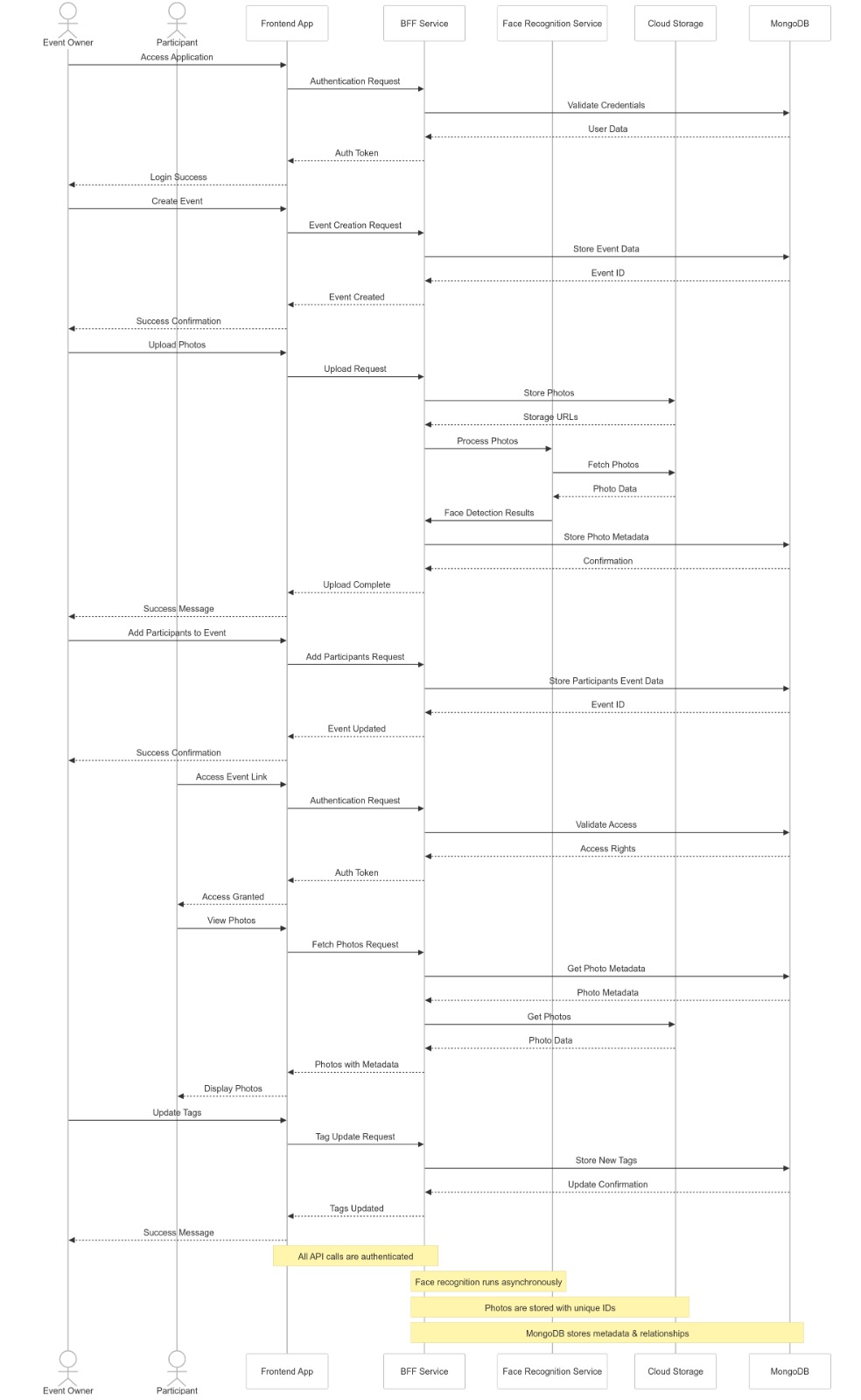
התיאור נוצר באופן אוטומטי**

התרשים מציג את המערכת באמצעות ישויות שונות ויחסים ביניהם. ישויות מרכזיות במערכת כוללות את Users (משתמשים), Events (אירועים), Photos (תמונות), ו-PhotoGroups (קבוצות תמונות).

* Users מייצגים את המשתמשים במערכת, יכולים להיות Owners (בעלי אירועים) או Participants (משתתפים), עם מידע אישי כמו שם, דוא"ל, סיסמה ומספר טלפון.
* Events מייצגים את האירועים השונים במערכת, וכל אירוע מקושר למספר Owners ו-Participants. האירועים גם כוללים Photos ו-PhotoGroups, כאשר כל תמונה שייכת לאירוע ולקבוצת תמונות מסוימת.
* Photos מייצגות את התמונות הקשורות לאירועים, וכל תמונה שייכת לקבוצת תמונות (PhotoGroup) שיכולה לכלול מספר Participants.
* PhotoGroups מייצגות קבוצות של תמונות הקשורות לאירועים ספציפיים, עם תיאור, שם ומספר משתתפים.

הקשרים בין הישויות מייצגים את האינטראקציות וההשתייכויות ביניהם, כמו למשל, שאירועים יכולים לכלול מספר תמונות וקבוצות תמונות, שמשתתפים יכולים להיות מקושרים אליהם, ושהתמונות שייכות לקבוצות אלו.

**Sequence Diagrams**



התרשים מתאר את תהליך העבודה של מערכת לניהול אירועים ותמונות:

1.אימות והתחברות: בעל האירוע מתחבר למערכת דרך ה-Frontend App, המוודא את זהותו מול שירות ה-BFF ו-MongoDB.

2.יצירת אירועים: בעל האירוע יוצר אירוע חדש דרך הממשק, והמידע נשמר ב-MongoDB.

3.העלאת תמונות: בעל האירוע מעלה תמונות שנשמרות בענן (Cloud Storage).

4.הזמנת משתתפים: בעל האירוע מזמין משתתפים, והמערכת מעדכנת את פרטי המשתתפים לאירוע.

5. זיהוי פנים מופעל על התמונות והנתונים נשמרים במסד הנתונים.

6.אימות והתחברות: משתתף באירוע מתחבר למערכת דרך ה-Frontend App, המוודא את זהותו מול שירות ה-BFF ו-MongoDB.

7.גישה וצפייה בתמונות: המשתתפים מתחברים לאירוע, צופים בתמונות דרך הממשק, שמציג את המידע והתמונות מהענן.

7.תיוג ידני: בעל האירוע יכול לעדכן תגים לתמונות, והמידע מתעדכן במסד הנתונים.

התרשים כולל תיאור ל תהליך ההתחברות מול ה- API, זיהוי פנים אסינכרוני, ושמירת מידע עם מזהים ייחודיים.

**Architecture Diagrams**

תמונה שמכילה טקסט, תרשים, צילום מסך, תוכנית

התיאור נוצר באופן אוטומטי

התרשים מציג מערכת המחולקת לשכבות:

1. משתמשים: בעל האירוע ומשתתפים מתקשרים דרך ממשק המשתמש.
2. צד לקוח: אפליקציית React.js לניהול אירועים, העלאת תמונות וגלריה.
3. צד שרת: שכבת BFF מנהלת API, אימות וניהול נתונים; שירות זיהוי פנים לקיבוץ תמונות.
4. אחסון: MongoDB שומר נתונים, וענן מאחסן תמונות גולמיות ומעובדות.

המערכת מחברת משתמשים עם שירותי צד לקוח, צד שרת ואחסון.

**Risks Management**

* סיכוני פרטיות ואבטחת מידע: חשיפה של תמונות אישיות או דליפת נתונים רגישים.  
  פתרון: הגבלת גישה לנתונים רגישים לפי הרשאות (שימוש ב Token).
* אי-דיוק בזיהוי פנים: אלגוריתם זיהוי הפנים עלול לזהות אנשים בצורה שגויה (False Positive/Negative).  
  פתרון: מתן אפשרות למשתמשים מסוג בעלי אירוע לערוך תיוגים ידניים במקרה של טעויות.
* עומסים ותקלות בביצועים: עיבוד כמות גדולה של תמונות עלול לגרום לזמני תגובה ארוכים או לקריסת המערכת.  
  פתרון: תכנון ארכיטקטורה תומכת עומסים (Load Balancing).
* סיכוני חוויית משתמש (UX/UI): ממשק מסובך או לא אינטואיטיבי עלול להרחיק משתמשים.  
  פתרון: עיצוב ממשק פשוט ואינטואיטיבי שמתאים לכל רמות המשתמשים ומתן תמיכה טכנית למשתמשים במידת הצורך בנוסף, יועלה סרטון הדרכה שמדגים את התהליך.

**Engineering Challenges**

1. אתגר:  
   התמודדות עם עיבוד כמויות גדולות של תמונות, תוך שמירה על זמני תגובה קצרים, מניעת קריסות מערכת, ותמיכה במספר גבוה של משתמשים במקביל באמצעות Load Balancing.  
   מענה:  
   עיבוד תמונות:

נדרש לחלק את עומס עיבוד התמונות בין שרתים או תהליכים בצורה יעילה.

על המערכת לתמוך ב-100+ תמונות בפעולה אחת(Batching), תוך שמירה על זמן עיבוד ≤ 5 דקות.  
ריבוי משתמשים:

המערכת צריכה לתמוך במספר גבוה של משתמשים בו-זמנית.

1. אתגר:  
   שמירה על פרטיות המשתמשים באמצעות מנגנון מאובטח לניהול גישה מבוסס הרשאות, תוך שימוש ב-Tokens לאימות וזיהוי משתמשים.  
   מענה:

שימוש ב-Tokens: JSON Web Tokens (JWT) הוא פתרון נפוץ לניהול גישה, אך דורש מנגנוני יצירה, אימות, וניהול מחזור חיים (Lifecycle) מאובטחים.

יש להגן על ה-Tokens מפני זיוף, גניבה או שימוש חוזר.

1. אתגר:  
   פיתוח מערכת WEB שתפעל בצורה תקינה ואופטימלית לפחות בשלושת הדפדפנים המובילים (Google Chrome, Firefox, Safari), תוך שמירה על תאימות ויכולת ביצוע גבוה, עם אפשרות להרחבה לאפליקציות ניידות (iOS ו-Android).  
   מענה:  
   דפדפנים שונים פועלים על מנועי עיבוד שונים מה שגורם להבדלים ביישום תקני HTML, CSS ו-JavaScript.

התאמת ממשק ה-WEB למסכים קטנים דורשת עיצוב רספונסיבי.

1. אתגר:  
   הטמעת מנגנון שיאפשר למשתמשים מסוג בעלי אירוע לערוך תיוגים ידניים של תמונות במערכת, במקרים שבהם זיהוי הפנים נכשל או ביצע טעויות (False Positives/Negatives), תוך שמירה על חוויית משתמש נוחה.

מענה:  
ממשק משתמש לעריכת תיוגים:

דרוש עיצוב ממשק ידידותי ויעיל, שיאפשר עריכת תיוגים ידניים במהירות ובנוחות.

יש לכלול אפשרויות כמו בחירה או מחיקה של משתתף מזוהה, או הוספת משתתף חדש לתמונה.

**Limitations**

מגבולות טכניות:

* ידע בטכנולוגיות זיהוי פנים:

הטמעת טכנולוגיות זיהוי פנים מתקדמות (כגון שימוש ב-OpenCV או מודלים מבוססי למידת מכונה) דורשת מומחיות. נדרש להשקיע זמן בלמידת הספריה ויכולותיה על מנת למקסם את השימוש בה.  
פתרון: לימוד הדרגתי של OpenCV ורכישת מיומנויות שימוש גבוהות בספריה.

* תאימות לפלטפורמות:

תמיכה מלאה בדפדפנים המובילים ובאפליקציות ניידות (iOS, Android) מצריכה ידע בפיתוח חוצה פלטפורמות (Cross-Platform), עיצוב רספונסיבי, ואופטימיזציה של ביצועים.  
פתרון: עיצוב רספונסיבי בצד הלקוח, כהתחלה נתמקד בפיתוח Web בלבד.

מגבלות זמן ועבודת צוות:

* תלות בין תחומים:

ה-Frontend תלוי בזמינות ה-Backend והבינה המלאכותית, מה שעלול לעכב את הפיתוח אם אחד החלקים אינו מוכן בזמן.  
פתרון: נתמקד במאפיינים הקריטיים ביותר קודם, ונבצע את הפיתוח לפי סדר חשיבות ומורכבות הפיתוח.

* זמן עבודה מוגבל:

אם חברי הצוות אינם פנויים לעבודה מלאה על הפרויקט (למשל, בשל לימודים או התחייבויות אחרות), התקדמות הפרויקט עלולה להתעכב.  
פתרון: נתכנן את העבודה בצורה מסודרת באמצעות מערכת לניהול משימות אותן נחלק בצורה ברורה ופשוטה, כך שכל חבר צוות יוכל לעבוד על חלק ספציפי בפרויקט בזמן המתאים לו.  
בנוסף נקיים ישיבות הערכת מצב בהן נבחן את עמידתנו ביעדים ונקבע יעדים חדשים.

**Work Plan (Gant)**

תמונה שמכילה טקסט, צילום מסך, צבעוני, מספר

התיאור נוצר באופן אוטומטי

סיכום משימות ותיאורים:

1. Define project infrastructure (Backend)

הגדרת התשתית לפרויקט ובחירת מסגרת עבודה מתאימה.

1. Create basic API (Backend)

פיתוח API בסיסי לניהול משתמשים ואירועים.

1. Set up database (Backend)

הקמת בסיס נתונים, יצירת טבלאות וחיבור ל Backend-.

1. Develop UI for login and registration (Frontend)

פיתוח ממשקי משתמש עבור התחברות והרשמה.

1. Develop UI for event creation (Frontend)

יצירת ממשק משתמש עבור יצירת אירועים.

1. Initial integration and testing (Integration)

חיבור ראשוני של המערכת וביצוע בדיקות בסיסיות.

1. Develop image upload system (Backend)

פיתוח מערכת להעלאת תמונות וניהול אחסון.

1. Email invitation services (Backend)

פיתוח שירותים לשליחת הזמנות לאירועים בדוא"ל.

1. Build image upload screen (Frontend)

יצירת מסך העלאת תמונות עם תצוגה מקדימה.

1. Event list screen (Frontend)

פיתוח מסך להצגת רשימת אירועים.

1. AI - Facial recognition integration (Backend)

שילוב זיהוי פנים למיון תמונות לפי משתתפים.

1. Testing and optimization (Integration)

ביצוע בדיקות למערכת ואופטימיזציה לביצועים.

1. Manual tagging and sharing features (Backend)

פיתוח תיוג ידני ושיתוף מאובטח של תמונות.

1. Sharing screen (Frontend)

יצירת מסך לניהול שיתוף תמונות.

1. Optimize system performance (Backend)

שיפור ביצועים ואופטימיזציה למסד הנתונים.

1. Enhance UI usability (Frontend)

שיפור חוויית המשתמש והוספת חיווי בזמן אמת.

1. Functional and security testing (Integration)

בדיקות פונקציונליות ובדיקות אבטחה.

1. Beta testing and feedback (Integration)

ביצוע בדיקות בטא ואיסוף פידבק ממשתמשים.

1. Deployment and documentation (Finalization)

פריסת המערכת וכתיבת תיעוד למשתמשים ולמפתחים.