**מסמך אפיון פרויקט פרקטיקום**

**חני ליברמן**

**✨🎵 Magical music 🎵✨**

**.1 תיאור המערכת**

**1.1 מטרת הפרויקט**

המערכת תאפשר למשתמשים לנהל את מאגר המוזיקה שלהם בצורה אינטואיטיבית, תוך שמירה על סדר ונגישות גבוהה. האפליקציה תאפשר למשתמשים לחפש שירים, לארגן אותן לפי אלבומים\ז׳אנרים\זמרים \מילות שיר וכדומה, וכן ליצור רשימות השמעה מועדפות, המלצות ודירוגי שירים, ליצור רינגטון משיר, ליצור תווים ואקורדים לשיר, לזהות ז'אנר בצורה אוטומטית, ועוד.

המטרה היא להקל על המשתמשים בניהול, ארגון וחיפוש שירים באופן נגיש ומאובטח.

**1.2 קהל היעד**

הקהל היעד של האפליקציה הוא אנשים פרטיים ועסקיים הזקוקים לפתרון לניהול מאגר המוזיקה בצורה מסודרת, כולל:

* משתמשים פרטיים חובבי מוזיקה.
* נגנים, בעלי תזמורות ואולפני סאונד שצריכים שירים או תווים או את שאר הדברים שהאפליקציה מציעה עבור עבודתם.
* עסקים קטנים או חברות שמשתמשות במוזיקה ובאפשרויותיה, לדוגמא מסעדות עבור מנגינות רקע.

**1.3 פונקציונליות של המערכת**

המערכת תכלול את הפונקציות הבאות:

1. העלאת מוזיקה לענן.
2. יצירת קטגוריות וז'אנרים על מנת להקל על החיפוש.
3. חיפוש שירים לפי ז׳אנר, זמר, מילות שיר וכדומה.
4. יצירת פלייליסט, תיוג שירים, דירוג והמלצות
5. יצירת תווים לשיר, זיהוי ז׳אנר אוטומטי, מנגנון המלצות.
6. יצירת רינגטון משיר.

**1.4 בעיות שהמערכת פותרת**

המערכת פותרת את הבעיה של ניהול מאגר מוזיקה מפוזר בתיקיות שנות, וכך תאפשר חיפוש נוח ומהיר לכל שיר אפשרי. המערכת מבטיחה גיבוי אוטומטי, ומאפשרת הורדת מוזיקה נוחה, בנוסף. המערכת מאפשרת פיצ'רים חדשניים הממקסמים את חווית המשתמש.

**1.5 טכנולוגיות בשימוש**

המערכת תפותח באמצעות הטכנולוגיות הבאות:

* צד שרת (Backend): .NET 9, לבניית ה-API.
* צד לקוח (Frontend): React.js, לבניית ממשק משתמש אינטראקטיבי + ממשק ניהול ב-Angular.
* מסד נתונים: PostgreSQL לאחסון נתונים של משתמשים ושירים.
* אחסון קבצים: שימוש ב-Amazon S3 לאחסון שירים.
* מערכת ליצירת תווים באמצעות טכנולוגיית AI.

**1.6 מתודולוגיית פיתוח (Agile)**

הפרויקט יפותח במתודולוגיית Agile עם חלוקה ל-5 ספרינטים בני שבועיים כל אחד. כל ספרינט יתמקד במשימות ליבה עיקריות. להלן פירוט הספרינטים הצפויים:

* **ספרינט 1** – בניית תשתיות וניהול משתמשים.
* **ספרינט 2** – ממשק משתמש ראשוני.
* **ספרינט 3** – פיצ'רים ליבה (הוספה, חיפוש, ניהול).
* **ספרינט 4** – שיפורים ואבטחה.
* **ספרינט 5** – פריסה ופרסום.

**2. עיצוב ממשק משתמש ויזואלי**

**שרטוטים (Wireframes)**

1. **מסך כניסה/רישום**:
   * שדות: שם משתמש, סיסמא, כפתור רישום/כניסה.
   * תיאור זרימה: לחיצה על "כניסה" תביא לממשק הראשי.
2. **מסך ניהול קבצים**:
   * רשימת שירים עם כפתורי מידע הורדה ומחיקה.
   * כפתור הוספת שיר למאגר.
   * תיאור זרימה: כל שיר יכול להיבחר לצפייה במידע, למחיקה או להורדה, וכן לשיוך לקטגוריה המתאימה.
3. **מסך אדמין**:
   * דוחות וסטטיסטיקות פעילות.
   * ניהול משתמשים ועריכת הרשאות.
   * תרשים שירים מועדפים.
   * רשימת השירים שנוספו לאחרונה.

**3. איפיון תמציתי**

**3.1 רשימת פונקציות המערכת**

| **פונקציה** | **Route** | **פרמטרים** | **פלט** | **לוגיקה עסקית** |
| --- | --- | --- | --- | --- |
| העלאת קובץ | POST /api/upload | Headers: Authorization: Bearer , Body: { file: File, tags: string[] } | { "fileId": "12345", "url": "https://storage.example.com/song.mp3" } | הגבלת גודל קובץ ל-50MB. סינון סוגי קבצים (mp3, wave). הוספת חתימת זמן לקובץ למניעת כפילויות. |
| הורדת קובץ | GET /api/files/{fileId} | Headers: Authorization: Bearer | קובץ השיר בפורמט המתאים | בדיקת הרשאות גישה לפני שליחת הקובץ. יומן הורדות לכל קובץ. |
| מחיקת קובץ | DELETE /api/files/{fileId} | Headers: Authorization: Bearer | 204 No Content | הרשאה למחיקה ניתנת רק למשתמש שהעלה את הקובץ או למנהל. מחיקת הרשומה מהמסד נתונים ואחסון בענן. |
| יצירת קטגוריה | POST /api/albums | Headers: Authorization: Bearer , Body: { name: string, description: string, files: string[] } | { "albumId": "67890", "name": "כאייל תערוג", "fileCount": 10 } | אימות שם ייחודי לכל קטגוריה. שיוך קבצים לקטגוריה בהתאם להרשאות משתמש. |
| הוספת קובץ לאלבום | PUT /api/albums/{albumId}/add | Headers: Authorization: Bearer , Body: { files: string[] } | 200 OK | אימות בעלות על השיר. ווידוא שקבצים שייכים למשתמש או שיש לו גישה. |
| שיתוף אלבום עם משתמשים אחרים | POST /api/albums/{albumId}/share | Headers: Authorization: Bearer , Body: { users: string[], permissions: string } | 200 OK | הרשאות גישה נקבעות בהתאם לסוג המשתמש (צפייה/עריכה/מחיקה). |
| חיפוש תמונות לפי תגיות או תאריך | GET /api/search | Query Params: tags: string[], dateRange: { from: string, to: string } | רשימת שירים מתאימים | שימוש במנוע חיפוש עם אינדוקס מהיר. הצגת תמונות לפי סדר רלוונטיות. |

**3.2 אימות והרשאות**

האפליקציה מבצעת אימות באמצעות JWT Authentication. המשתמשים יקבלו גישה לפונקציות מסוימות בהתאם להרשאותיהם:

* **משתמש רגיל**: יכול להעלות שירים, לערוך קטגוריות ולחפש שירים.
* **מנהל**: יכול לערוך משתמשים, להפיק דוחות ולנהל את ההגדרות הגלובליות של המערכת.

**3.3 אפליקציית ניהול**

מנהל המערכת יוכל לבצע את הפעולות הבאות:

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

**4. תרשים מבנה הטבלאות (Database Schema)**

**טבלאות נתונים:**

**טבלת Users:**

| **שם העמודה** | **סוג נתונים** | **מאפיינים** | **תיאור** |
| --- | --- | --- | --- |
| id | INT | PK, AUTO\_INCREMENT | מזהה ייחודי למשתמש |
| first\_name | VARCHAR(50) | NOT NULL | שם פרטי |
| last\_name | VARCHAR(50) | NOT NULL | שם משפחה |
| email | VARCHAR(100) | UNIQUE, NOT NULL | כתובת אימייל |
| password | VARCHAR(255) | NOT NULL | סיסמה (מאובטחת) |
| created\_at | DATETIME | DEFAULT NOW() | תאריך יצירה |
| updated\_at | DATETIME | NULL | תאריך עדכון |
| role | VARCHAR(20) | NOT NULL, DEFAULT 'user' | תפקיד המשתמש (user/admin) |

**טבלת Songs:**

| **שם העמודה** | **סוג נתונים** | **מאפיינים** | **תיאור** |
| --- | --- | --- | --- |
| id | INT | PK, AUTO\_INCREMENT | מזהה ייחודי לשיר |
| user\_id | INT | FK | מזהה המשתמש שהעלה את השיר |
| album\_id | INT | FK, NULL | מזהה האלבום (אם יש) |
| Creator | VARCHAR(20) | NOT NULL | יוצר השיר |
| file\_type | VARCHAR(20) | NOT NULL | סוג הקובץ (mp3,wave) |
| created\_at | DATETIME | DEFAULT NOW() | תאריך יצירה |
| tags | VARCHAR(255) | NULL | תיוגים נוספים לשיר |

**טבלת Categories:**

| **שם העמודה** | **סוג נתונים** | **מאפיינים** | **תיאור** |
| --- | --- | --- | --- |
| id | INT | PK, AUTO\_INCREMENT | מזהה ייחודי לקטגוריה |
| user\_id | INT | FK | מזהה המשתמש שיצר את הקטגוריה |
| name | VARCHAR(100) | NOT NULL | שם הקטגוריה |
| description | TEXT | NULL | סוג הקטגוריה (זמר, תאריך, ז'אנר) |
| created\_at | DATETIME | DEFAULT NOW() | תאריך יצירה |

**טבלת Logs (יומן פעולות):**

| **שם העמודה** | **סוג נתונים** | **מאפיינים** | **תיאור** |
| --- | --- | --- | --- |
| id | INT | PK, AUTO\_INCREMENT | מזהה ייחודי ליומן |
| user\_id | INT | FK | מזהה המשתמש שעשה את הפעולה |
| action | VARCHAR(100) | NOT NULL | סוג הפעולה (העלאה, מחיקה וכו') |
| description | TEXT | NULL | תיאור הפעולה |
| created\_at | DATETIME | DEFAULT NOW() | תאריך ושעה של הפעולה |

**קשרים בין טבלאות:**

1. \*\*Users ↔ Songs\*\*:

- קשר Many-to-Many. לכל משתמש יכולות להיות שירים רבים, ו כל שיר שייך לכולם.

2. \*\*Users ↔ Categories \*\*:

- קשר One-to-Many. לכל משתמש יכולות להיות קטגוריות רבות, אך כל קטגוריה שייכת למשתמש אחד.

3. \*\* Categories ↔ Songs (דרך Categories Files)\*\*:

- קשר Many-to-Many. אלבום אחד יכול להכיל תמונות רבות, ותמונה אחת יכולה להימצא בכמה אלבומים.

4. \*\*Users ↔ Logs\*\*:

- קשר One-to-Many. לכל משתמש יכולות להיות מספר פעולות ביומן (Logs).

**5. הגדרת סבבי פיתוח (ספרינטים)**.

**ספרינט 1 (שבוע 1-2): תשתית וניהול משתמשים**

* הקמת מסד נתונים PostgreSQL.
* פיתוח REST API לניהול משתמשים.
* אימות משתמשים עם Firebase Authentication.

**ספרינט 2 (שבוע 3-4): העלאת שירים וארגון בקטגוריות**

* בניית ממשק העלאת שירים וארגון בקטגוריות השונות ב-React.
* שמירת שירים ב-AWS S3 והוספת מטא-דאטה למסד הנתונים.

**ספרינט 3 (שבוע 5-6): תיוג וחיפוש מתקדם**

* אינטגרציה עם Suno Ai\Mubert API ליצירת תווים לשיר.
* פיתוח חיפוש לפי תגיות, קטגוריות, תאריכים ואנשים.

**ספרינט 4 (שבוע 7-8): שיתוף שירים וניהול הרשאות**

* אפשרות לשיתוף קטגוריות עם משתמשים אחרים.
* הגדרת הרשאות גישה לכל משתמש.

**ספרינט 5 (שבוע 9-10): פריסה, בדיקות ואופטימיזציה**

* פריסת השרת ב-AWS Lambda ו-Frontend ב-Vercel.
* בדיקות אבטחה ו-Performance Testing.
* דף נחיתה שיווקי והכנת דמו