בס"ד

מסמך אפיון פרקטיקום

אפרת אביגיל ישראלי

Magical music

1תיאור המערכת

1.1מטרת 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 חיפוש שירים באופן נגיש ומאובטח.

2.1קהל 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.

3פונקציונליות של המערכת

המערכת תכלול את הפונקציות הבאות:

1. העלאת שירים חדשים לענן.
2. יצירת קטגוריות אלבומים וז'אנרים ע"מ להקל על החיפוש..
3. שיתוף שירים עם משתמשים אחרים.
4. חיפוש שירים לפי זמר או מילות השיר.
5. גיבוי אוטומטי של אלבומים.
6. יצירת רינגטון משיר
7. יצירת פלייליסט, תיוג שירים דירוג והמלצות.
8. חיפוש שירים לפי ז'אנר , זמר , מילות שיר
9. יצירת תווים לשיר
10. זהוי ז'אנר אוטומטי
11. מנגנון המלצות

1.4בעיות שהמערכת פותרת

המערכת פותרת את הבעיה של ניהול מאגר מוזיקה ומפוזר בין תקיות שונות שונים, בכך שהיא מאפשרת חיפוש נח ומהיר . המערכת מבטיחה גיבוי אוטומטי, כך שהשירים לא יאבדו, ומאפשרת שיתוף קל עם אחרים, מה שמפשט את התהליך עבור משתמשים. בנוסף, המערכת מציעה פיצ'רם חדשניים הממקסים את חווית המשתמש, כמו יצירת תווים לשיר, מנגנון המלצות.

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 ומוזיקה.
- אחסון קבצים: שימוש ב Amazon S3-לאחסון שירים.
- מערכת ליצירת תווים לשיר ע"י 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 (שיתוף, חיפוש, ניהול, הוספה).
- **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 למשק הראשי.
2. **מסך ניהול קבצים:**
   - רשימת שירים עם כפתורי מידע מחיקה.והורדה
   - כפתור להוספת שיר למאגר
   - תיאור זרימה: כל שיר יוכל להיבחר לצפיה במידע או למחיקה המאפשרת הוספת תגיות וארגון לפי קטגוריות.
3. **מסך אדמין:**
   - דוחות וסטטיסטיקות פעילות.
   - ניהול משתמשים ועריכת הרשאות.
   - שירים מועדפים.
   - רשימת השירים שנוספו לאחרונה.

---

3. איפיון תמציתי

3.1 רשימת פונקציות המערכת

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

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

תיאור	מאפיינים	סוג נתונים	שם העמודה
מזהה ייחודי למשתמש	PK, AUTO_INCREMENT	INT	id
שם פרטי	NOT NULL	VARCHAR(50)	first_name
שם משפחה	NOT NULL	VARCHAR(50)	last_name
כתובת אימייל	UNIQUE, NOT NULL	VARCHAR(100)	email
סיסמה (מאובטחת)	NOT NULL	VARCHAR(255)	password
תאריך יצירה	DEFAULT NOW()	DATETIME	created_at
תאריך עדכון	NULL	DATETIME	updated_at
תפקיד המשתמש(user/admin)	NOT NULL, DEFAULT 'user'	VARCHAR(20)	role

טבלת: Songs

תיאור	מאפיינים	סוג נתונים	שם העמודה
מזהה ייחודי לahr	PK, AUTO_INCREMENT	INT	id
מזהה המשתמש שהעלה את שיר	FK	INT	user_id
מזהה האלבום (אם יש)	FK, NULL	INT	album_id
יוצר השיר- זמר	NOT NULL	VARCHAR(255)	creator
סוג הקובץ (mp3, wav)	NOT NULL	VARCHAR(20)	file_type
תאריך יצירה	DEFAULT NOW()	DATETIME	created_at
תיוגים נוספים לשיר	NULL	VARCHAR(255)	tags

טבלת: category

תיאור	מאפיינים	סוג נתונים	שם העמודה
מזהה ייחודי לקטגוריה	PK, AUTO_INCREMENT	INT	id
מזהה המשתמש שיצר את הקטגוריה	FK	INT	user_id
סוג הקטגוריה(זמר, ז'אנר, סגנון)	NOT NULL	VARCHAR(100)	name
תיאור הקטגוריה	NULL	TEXT	description

תיאור	מאפיינים	סוג נתונים	שם העמודה
תאריך יצירה	DEFAULT NOW()	DATETIME	created_at

טבלת) Logs יומן פעולות:(

תיאור	מאפיינים	סוג נתונים	שם העמודה
מזהה ייחודי ליומן	PK, AUTO_INCREMENT	INT	id
מזהה המשתמש שעשה את הפעולה	FK	INT	user_id
סוג הפעולה (העלאה, מחיקה וכו')	NOT NULL	VARCHAR(100)	action
תיאור הפעולה	NULL	TEXT	description
תאריך ושעה של הפעולה	DEFAULT NOW()	DATETIME	created_at

קשרים בין טבלאות:

1. **Users ↔ Songs**:

- קשר Many-to-Many. לכל משתמש יכולות להיות שירים רבים, וכל שיר משתייך למספר משתמשים בו זמנית.

2. **Users ↔ **category**:

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

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 השונות

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

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

- אינטגרציה עם APIליצירת תווים לשיר -Mubert / SUNO AI
- פיתוח חיפוש לפי תגיות, תאריכים ואנשים.

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

- אפשרות לשיתוף שירים 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

שיהיה ההמון בהצלחה 👍 👍