# מסמך אפיון Tune Your Mood

## תיאור המערכת

**1.1 מטרת הפרויקט:**

הפרויקט נועד לספק למשתמשים חוויית האזנה מותאמת אישית, עם דגש על נוחות ושירותים חדשניים בתחום המוזיקה. המערכת כוללת נגן מוזיקה חכם שמאפשר למשתמשים גישה קלה ונוחה לשירים אהובים, תוך מתן אפשרות לחיפוש מהיר של שירים לפי שם, אמן או ז'אנר. בנוסף, המשתמשים יכולים ליצור ולנהל פלייליסטים אישיים.

אחד הפיצ'רים הייחודיים של המערכת הוא היכולת להתאים את המוזיקה למצב הרוח של המשתמש. הנגן יכול לזהות את המצב רוח הנוכחי של המשתמש, ולהציע מוזיקה שתשדרג את החוויה, בין אם מדובר במוזיקה רגועה ומרגיעה בזמן מנוחה, או במוזיקה אנרגטית ומרעננת לפעילויות ספורטיביות. כל אלו יחד יוצרים חוויית האזנה אישית ומעמיקה יותר, המותאמת בצורה מושלמת לצרכים וההעדפות של המשתמשים.

**1**.**2 קהל היעד:**

* אנשים שמחפשים פתרון פשוט ונגיש לשמיעת מוזיקה מכל מכשיר.
* משתמשים המעוניינים לארגן את אוספי המוזיקה האישיים שלהם בצורה מסודרת.
* אנשים שמחפשים אפשרויות נוחות לגיבוי המוזיקה שלהם ולאיבוד שירים חשובים.
* **אנשים שרוצים למצוא שירים לפי מצב רוח או תחושה שהם חווים**:

**1.3 פונקציונליות של המערכת:**

* **חיפוש שירים** - לפי ז'אנר, שם שיר, שם אמן, או מילות שיר.
* **יצירת פלייליסטים מותאמים** - המשתמש יכול ליצור, לשמור ולשתף רשימות השמעה.
* **תיוג שירים ודירוג** - כל שיר ניתן לתיוג לפי קטגוריות, דירוג אישי או מצב רוח.
* **יצירת רינגטונים** - המערכת מאפשרת חיתוך קטעים משירים ושמירתם כרינגטון.
* **התאמת שירים לפי מצב רוח** - משתמש יכול להקליד תחושה או משפט, והמערכת תמליץ על שירים מתאימים.
* **ניהול ספריית מוזיקה אישית** - הוספת שירים לרשימת מועדפים, סינון וסידור מוזיקה.
* **שירותי ענן** - גיבוי ושמירת רשימות השמעה ושירים מועדפים בענן.

**1.4 בעיות שהמערכת פותרת:**

* מקלה על חיפוש שירים וריכוזם במקום אחד.
* היא מאפשרת גישה נוחה לכל השירים ממקום אחד מאורגן.
* מאפשרת המלצות מותאמות אישית לפי מצב רוח.
* מסייעת ביצירת פלייליסטים חכמים ומנוהלים בצורה טובה.
* משפרת את חוויית השימוש עם אפשרות שיתוף אחרים בקלות.

**1.5 טכנולוגיות בחיפוש:**

* **:Backend** .NET לפיתוח צד שרת.
* **:Frontend** React לבניית ממשק משתמש דינמי, Angular לתמיכה במשימות משניות.
* **:Database** sqlServe בסיס נתונים מותאם לאחסון שירים, משתמשים ותיוגים.
* **:AI** מודל למידת מכונה לסיווג שירים לפי קטגוריות מצב רוח.
* **:Cloud Storage**פתרון לאחסון שירים ופלייליסטים.

## **עיצוב ממשק משתמש ויזואלי**

* **מסך התחברות / הרשמה** – משתמשים יכולים ליצור חשבון ולהתחבר.
* **דף חיפוש שירים** – חיפוש לפי קריטריונים שונים.
* **מסך התאמת שיר לפי מצב רוח** – תיבת טקסט להקלדת תחושה וקבלת שיר מותאם.
* **מסך ניהול פלייליסטים** – יצירה, עריכה ומחיקת רשימות השמעה.
* **דף הגדרות משתמש** – שינוי העדפות והתאמות אישיות.

## **אפיון תמציתי**

**3.1 רשימת פונקציות המערכת**

#### **.1 חיפוש שירים**

מאפשר למשתמשים לחפש שירים לפי שם, אמן, ז'אנר או מילים.

* **Route:** GET /api/search
* **פרמטרים:**
  + query: string – מחרוזת חיפוש (שם שיר, שם אמן, מילים)
  + filter: enum (genre/artist/lyrics) – סוג הסינון (ז'אנר, אמן או מילים)
* **פלט:**
  + results:
    - songId: int
    - title: string
    - artist: string
    - genre: string

#### **.2 יצירת וניהול פלייליסטים**

מאפשר למשתמשים ליצור, לערוך ולנהל פלייליסטים אישיים.

* **יצירת פלייליסט חדש**
  + **Route:** POST /api/playlists
  + **פרמטרים:**
    - name: string – שם הפלייליסט
    - songs: array[int] – מזהי השירים בפלייליסט
  + **פלט:**
    - playlistId: int
    - name: string
    - songs: array[int]
* **עדכון פלייליסט**
  + **Route:** PUT /api/playlists/{playlistId}
  + **פרמטרים:**
    - name: string – שם מעודכן
    - songs: array[int] – רשימת שירים מעודכנת
* **מחיקת פלייליסט**
  + **Route:** DELETE /api/playlists/{playlistId}
  + **פלט:**
    - message: string

#### **3. המלצות שירים לפי מצב רוח**

מנוע AI מנתח את הטקסט שהמשתמש מקליד ומציע שירים מתאימים.

* **Route:** POST /api/mood
* **פרמטרים:**
  + text: string – תיאור מצב רוח
* **פלט:**
  + songs:
    - songId: int
    - title: string
    - artist: string
    - genre: string

#### **4. יצירת רינגטון משיר**

מאפשר למשתמשים לבחור חלק משיר ולהורידו כרינגטון.

* **Route:** POST /api/ringtone
* **פרמטרים:**
  + songId: int – מזהה השיר
  + startTime: string – נקודת התחלה (בפורמט mm:ss)
  + duration: int – אורך הרינגטון בשניות
* **פלט:**
  + downloadUrl: string

#### **. 5 ניהול משתמשים**

מאפשר למנהלים לנהל משתמשים, תפקידים והרשאות.

* **רישום משתמש חדש**
  + **Route:** POST /api/users/register
  + **פרמטרים:**
    - email: string – כתובת אימייל
    - password: string – סיסמה
  + **פלט:**
    - userId: int
    - message: string
* **כניסה למערכת (Login)**
  + **Route:** POST /api/users/login
  + **פרמטרים:**
    - email: string
    - password: string
  + **פלט:**
    - token: string
    - userId: int
    - role: string
* **עדכון תפקיד משתמש**
  + **Route:** PUT /api/users/{userId}/role
  + **פרמטרים:**
    - role: string – תפקיד חדש
  + **פלט:**
    - message: string

**3.2 אימות והרשאות**

#### **מערכת אימות (Authentication)**

* **JWT Authentication**: כל הבקשות למערכת ידרשו אסימון (Token) מבוסס JWT (JSON Web Token) כדי להבטיח גישה מאובטחת.
* **תהליך ההתחברות**:
  + המשתמש מזין שם משתמש וסיסמה (או משתמש בגישה חיצונית כמו Google/Facebook OAuth).
  + השרת מאמת את הנתונים ושולח אסימון JWT ללקוח.
  + האסימון מאוחסן בצד הלקוח (localStorage/sessionStorage) ומשמש לאימות בכל בקשה עתידית.
  + האסימון פג תוקף לאחר זמן מוגדר ויש צורך לרענן אותו (Refresh Token).

#### **תפקידי משתמשים (User Roles)**

1. **משתמש** (User)
   * גישה לנגן המוזיקה ולכלי החיפוש.
   * יצירת וניהול פלייליסטים אישיים.
   * דירוג שירים וקבלת המלצות AI מותאמות אישית.
   * גישה חופשית לשירים מותאמים למצב רוח.
   * יכולות מתקדמות ביצירת רינגטונים ועריכת שירים.
2. **מנהל מערכת** (Admin)
   * גישה מלאה לכל נתוני המשתמשים וניהול הרשאות.
   * הפקת דוחות וסטטיסטיקות על השימוש במערכת.
   * ניטור מערכת והתראות על תקלות.

#### **ניהול הרשאות (Access Control)**

* לכל תפקיד יוגדרו הרשאות ספציפיות.
* ההרשאות ייבדקו בכל בקשה ל-API באמצעות JWT Middleware.
* ממשק ניהול יאפשר למנהלים לשנות הרשאות משתמשים בקלות.
  1. **אפליקציות ניהול**

#### **יכולות עיקריות:**

##### **1. דוחות וסטטיסטיקות שימוש**

* מעקב אחרי מספר המשתמשים הפעילים ביום/שבוע/חודש.
* הצגת נתונים על השירים הכי מושמעים והז'אנרים הפופולריים.
* סטטיסטיקות חיפוש: אילו חיפושים נפוצים ביותר? אילו מצבי רוח הכי נבחרים?

##### **2. ניהול משתמשים**

* **פעולות CRUD (Create, Read, Update, Delete)** למשתמשים.
* חיפוש משתמשים לפי שם, אימייל, סטטוס מנוי או תאריך הרשמה.
* שינוי תפקידי משתמשים (שדרוג לפרימיום, השעיית משתמש, חסימה).
* הצגת היסטוריית פעילות של משתמשים:
  + שירים שהושמעו לאחרונה.
  + חיפושי שירים לפי מצב רוח.
  + דירוגי שירים ופלייליסטים שנוצרו.

##### **3. הגדרות מערכת (System Settings)**

* ניהול מגבלות אחסון בענן לכל משתמש.
* קביעת פרמטרים עבור מנוע ה-AI (דיוק, רגישות לקטגוריות מצב רוח).
* התאמת פרמטרי ניתוח מוזיקלי (כגון התאמה לפי BPM, סגנון, מילים).
* הפעלת מצב תחזוקה במקרה של עדכונים קריטיים.

## **תרשים מבנה טבלאות**

**טבלת Users**

|  |  |  |
| --- | --- | --- |
| **מאפיינים** | **סוג נתונים** | **עמודה** |
| Primary key | int | Id |
| Not null | Varchar(255) | Name |
| Unique | Varchar(255) | Email |
| Default now() | Timestamp | Created\_at |

**טבלת Songs**

|  |  |  |
| --- | --- | --- |
| **מאפיינים** | **סוג נתונים** | **עמודה** |
| Primary key | int | Id |
| Not null | Varchar(255) | Title |
| Not null | Varchar(255) | Artist |
| Not null | Varchar(100) | Genre |
|  | Varchar(100) | mood\_category |

## **הגדרות סבבי פיתוח**

### **ספרינט 1 - הקמת תשתיות ופיתוח API בסיסי**

* תכנון מסד הנתונים והקשרים בין הישויות.
* פיתוח API לניהול משתמשים ושירים.
* הטמעת אימות משתמשים עם JWT.

### **ספרינט 2 - בניית ממשק משתמש ראשוני**

* עיצוב דפי החיפוש, השמעת שירים וניהול פלייליסטים.
* חיבור ה-Frontend ל-API.

### **ספרינט 3 - פיתוח פיצ'רים מרכזיים**

* הטמעת מערכת חיפוש שירים לפי מילות שיר.
* פיתוח יכולות דירוג ושמירת העדפות משתמשים.

### **ספרינט 4 - פיצ'ר התאמת שירים למצב רוח**

* פיתוח אלגוריתם AI לניתוח טקסט והתאמת שירים.
* אינטגרציה עם המערכת לניהול מצב רוח.

### **ספרינט 5 - בדיקות, שיפורים ופריסה**

* בדיקות אבטחה ושיפור ביצועים.
* פריסת האפליקציה לשרתים.
* הכנת דוקומנטציה סופית.

### **כלי ניהול**

* **GitHub** לניהול גרסאות הקוד.