>> בכל סגנון שתבחרו.

עולם של מוזיקה יהודית



רק בס"ד

- 2020 -

המנחה – המו' זהבה ברון

>> הודיה עזרן ואודיה אושרי <<

|  |  |
| --- | --- |
| שם הסטודנט: הודיה עזרן | ת.ז. 208097014 |
| שם הסטודנט: אודיה אושרי | ת.ז. 322297995 |

הצהרת סטודנט

שם המכללה בה לומד הסטודנט: סמינר "זיכרון צבי"

אני הח"מ, מצהיר בזאת כי פרויקט הגמר וספר הפרויקט המצ"ב נעשו על ידי בלבד.

פרויקט הגמר נעשה על סמך הנושאים שלמדתי במכללה ובאופן עצמאי.

פרויקט הגמר וספר הפרויקט נעשו על בסיס הנחייתו של המנחה האישי.

מקורות המידע בהם השתמשתי לביצוע פרויקט הגמר מצוינים ברשימת המקורות המצוינים בספר הפרויקט.

אני מודע לאחריות שהנני מקבל על עצמי על ידי חתימתי על הצהרה זו שכל הנאמר בה אמת ורק אמת.



חתימת הסטודנט: \_\_\_\_\_\_\_\_\_\_ תאריך: 01/12/2020

חתימת הסטודנט: \_\_\_\_\_\_\_\_\_\_ תאריך: 01/12/2020

אישור המנחה האישי

הריני מאשר שהפרויקט בוצע בהנחייתי, קראתי את ספר הפרויקט ומצאתי כי הוא מוכן לצורך הגשת הסטודנט להגנה על פרויקט גמר.

|  |  |  |
| --- | --- | --- |
| שם המנחה: זהבה ברון | חתימה: \_\_\_\_\_\_\_\_\_\_\_ | תאריך: 01/12/2020 |



לכבוד יחידת הפרויקטים מה"ט

הצעה לפרויקט גמר

1. פרטי הסטודנטים

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| תאריך סיום הלימודים | טלפון נייד | כתובת | ת.ז. | שם הסטודנט |
| 7.2020 | 054-8507273 | שמעיה 32 ב"ב | 322297995 | אודיה אושרי |
| 7.2020 | 050-4117455 | לוי יצחק 6 ב"ב | 208097014 | הודיה עזרן |

שם המכללה: זיכרון צבי סמל המכללה: 72439

מסלול ההכשרה: (הנדסאים/~~טכנאים מוסמכים~~ )מחק את המיותר.

מגמת לימוד: תכנות מחשבים מקום ביצוע פרויקט: בסמינר ובבית

1. פרטי המנחה האישי

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| מקום עבודה/תפקיד | תואר | טלפון נייד | כתובת | שם המנחה |
| המכללה למנהל- סמינרים | B.A. | 0504128709 | הרב לוין מרישא 17 ב"ב | זהבה ברון |

\* עבור מנחה אישי חדש יש לצרף קורות חיים, ניסיון מקצועי ותעודות השכלה לאישור מה"ט.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| חתימת הסטודנט | חתימת המנחה האישי | חתימת הגורם המקצועי מטעם מה"ט |

דרך מנחם בגין 86 תל אביב ת.ד .36049 מיקוד 67138 טלפון: 7347521-03 פקס: 7347644-03

1. שם הפרויקט – Music Online
2. רקע -

תיאור ורקע כללי

האתר הוקם במטרה להנגיש מוזיקה יהודית חרדית. האתר מאפשר אפשרויות חיפוש נרחבות, שמיעת שירים ברצף לפי סגנון, וצפייה בקליפים. אפשרות שמירה רשימת שירים באזור האישי. מאפשר להתעדכן בחדשות המוזיקה.

מטרות המערכת

להנגיש שמיעת מוזיקה יהודית חרדית בצורה נוחה וכן שמירת רשימת שירים באזור האישי.

1. סקירת מצב קיים בשוק, אילו בעיות קימות -

האתרים שקיימים המציעים מוזיקה יהודית אינם מאפשרים שמירת רשימת שירים באזור האישי.

1. מה הפרויקט אמור לחדש או לשפר -

הפרויקט בא לאפשר שמירת שרים באזור האישי. האתר יוצג בצורה מזמינה וידידותית למשתמש, ודרך השימוש בו תאה מובנת ונוחה.

1. דרישות מערכת ופונקציונאליות -
   1. דרישות מערכת, סביבת הטמעה ושימוש

המערכת תעבור קומפילציה והפצה בסביבת Visual Studio עם התקנת.Net Framework גרסה 4 ומעלה והיא אמורה לרוץ בסביבת שרת אשר מריץ IISExpress לקבלת בקשות לתצוגת דפי אינטרנט.

המשתמש יוכל להריץ את האתר בכל מכשיר אשר מותקן עליו דפדפן אינטרנט.

* 1. שרידות ביצועים והתמודדות עם עומסים

צד השרת מריץ IISExpress המסוגל להתמודד עם מספר קריאות של קריאות בו זמנית.

גם עומס על שרת ה-SQL אינו צפוי בסדר גודל כזה של אתר מכיוון שהוא בנוי להתמודדות בהצלחה עם עומסים כבדים בהרבה.

* 1. דרישות פונקציונאליות

רשימת דרישות המשתמש מהמערכת:

* + 1. המשתמש יכול להירשם לאתר
    2. המשתמש יכול להירשם לרשימת תפוצה

5.3.2.1 המערכת תשלח למשתמש הרשום ברשימת התפוצה עדכונים במייל  
5.3.2.2 באם המשתמש גם רשום לאתר ,המערכת תשלח לו עדכונים מותאמים לאופי החיפושים שלו באתר

* + 1. עם כניסת המשתמש לאתר הוא יצפה ברשימות שירים על פי חתכים  
       5.3.3.1 המשתמש יכול לדווח על הפרת זכויות יוצרים באם ימצא שיר שלו שאיננו מעוניין בהפצתו.
    2. המשתמש יכול לחפש שירים על פי חתכים
       1. בעת החיפוש המשתמש בוחר קריטריון לחיפוש
    3. המשתמש יכול לדווח על הפרת זכויות יוצרים באם ימצא שיר שלו שאיננו מעוניין בהפצתו.
    4. המשתמש יכול לשמור שיר לרשימת השמעה
       1. המשתמש מזדהה במערכת  
          5.3.5.1.1 המשתמש יכול לשחזר סיסמא במקרה ששכח את סיסמתו
       2. המשתמש בוחר רשימת השמעה שלתוכה יתווסף השיר.
    5. המשתמש יכול להיכנס לאזור האישי שלו. (כולל הזדהות עם שם וסיסמה)
       1. המשתמש יכול להוסיף רשימת השמעה
       2. המשתמש יכול להעביר שיר מרשימת השמעה אחת לאחרת
       3. המשתמש יכול למחק שיר
       4. המשתמש יכול למחק רשימת השמעה
       5. המשתמש יכול למיין שירים
       6. המשתמש יכול לעדכן הגדרות אישיות
    6. המשתמש יכול להפעיל שירים או קליפים
    7. המשתמש יכול לצפות במגזין
       1. המשתמש יכול לבחור כתבה לצפייה.
       2. המשתמש יכול להוסיף תגובה לכתבה.
    8. המנהל יכול להוסיף שיר לאתר  
       5.3.10.1 המנהל מקטלג שיר עפ"י תגיות המופיעות באתר.
    9. המנהל יכול להוסיף זמר
    10. המנהל יכול להוסיף תגית
    11. המנהל יכול למחוק שיר
    12. המנהל יכול לצפות בדיווחים על הפרת זכויות יוצרים  
        5.3.14.1 המנהל יכול לשנות את סטטוס הדיווח
    13. המנהל יכול לצפות בסטטיסטיקות
        1. המנהל בוחר אופי הניתוח הסטטיסטי
    14. המנהל יכול להוסיף כתבה למגזין
    15. המנהל יכול למחוק כתבה מהמגזין
    16. המנהל יכול למחוק תגובה לכתבה
    17. המנהל יכול להוסיף תגית לכתבה

1. בעיות צפויות במהלך הפיתוח ופתרונות תפעוליות, טכנולוגיות, עומס ועוד -

תיאור הבעיות- הללו כפועל יוצא של דרישות המשתמש מהתוכנה.

בעיה 1 – במידה ואדם בעל זכויות יוצרים נתקל בהפרת זכויות איך יוכל לדווח על זה.

בעיה 2 – בהוספת שיר לרשימת השמעה לא דרך האזור האישי לאיזו רשימה התווסף.

פתרונות אפשריים -

לבעיה 1

פתרון 1 – נוסיף אפשרות דווח ליד כל שיר.

פתרון 2 – המערכת תקצה אזור כללי לדיווח.

לבעיה 2

פתרון 1 – השיר התווסף לרשימת השמעה דיפולטיבית ובאזור האישי המשתמש יוכל להעביר לרשימת השמעה ספציפית יותר

פתרון 2 – המשתמש יבחר רשימת השמעה והשיר התווסף ישירות לרשימה המבוקשת

הפתרון הנבחר עבור כל אחת מהבעיות -

לבעיה 1 - נוסיף אפשרות דווח ליד כל שיר

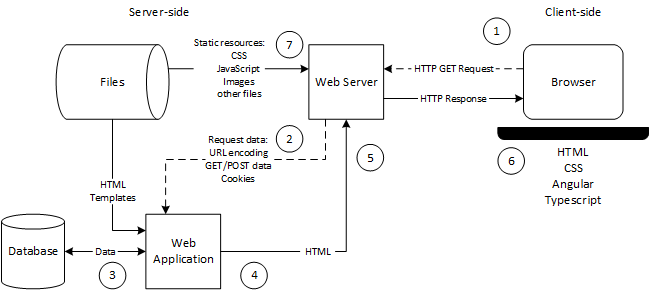
לבעיה 2 – המשתמש יבחר רשימת השמעה והשיר התווסף ישירות לרשימה המבוקשת

1. פתרון טכנולוגי נבחר -
   1. טופולוגית הפתרון

המערכת מורכבת מ**שרת** IIS המריץ את האתר בסביבת ה- Server,

**מסד נתונים** של SQL-Server

**ממשק משתמש** בצד הלקוח: דפדפן אינטרנט כלשהו כמו chromeעל מחשב שולחני או על מכשיר iOS/ Android



* 1. טכנולוגיות בשימוש

Angular- תשתית תוכנה ליישומי רשת בקוד פתוח המתוחזקת ע"י גוגל המשמשת בעיקר לפיתוח של אתרים דינמיים ומותאמת במיוחד לשימוש מקוון

Entity Framework- טכנולוגית גישה למסד נתונים בסביבת NET. המאפשרת גישה נוחה ל DB

WebApi– טכנולוגיה המקשרת בין צד שרת כלומר מסד נתונים לבין צד הלקוח - Angular

* 1. שפות הפיתוח:

שפות הפיתוח בצד השרת:

C# .NET - שפת פיתוח קוד שרת חזקה במשק בגלל העקרונות הפשוטים המאפשרים לנו לנהל מערכת חכמה והגיונית ומצד שני עשירה ביכולות מתקדמות C# הוא תנאי הכרחי לשימוש במגוון טכנולוגיות, מתוכן EDM

EDM (Entity Data Model) – הפיכת מסד הנתונים למודל ישויות ע"י הפיכת טבלה למחלקה ושדה ל- .property ובצורה זו ניתן יהיה לשלוף/למחוק/לעדכן נתונים ממסד הנתונים.

שפות הפיתוח בצד הלקוח:

TypeScript- שפת תכנות דינמית מונחית עצמים מבוססת מחלקות המותאמת לשילוב באתרי אינטרנט ורצה על ידי דפדפן האינטרנט בצד הלקוח. השפה מרחיבה את יכולות שפת התגיות הבסיסית HTML ומאפשרת בכך ליצור יישומי אינטרנט מתוחכמים יותר.

Angular- שפת פיתוח צד לקוח המוסיפה אבסטרקציות מעולם פיתוח Object Oriented, הכתיבה מתבצעת ע"י כתיבת HTML שבתוכו משולבים תוספות של Angular, כתיבת components כדי לשלוט על ה- ,templates כתיבה של

services כדי לשלוט על הלוגיקה, ושילוב של components ו- services ליצירת modules .

HTML5- היא שפת בניית אתרים לסימון טקסט, שפת תגיות, אשר נותנת הנחיות לדפדפנים בנוגע לאופן הצגתו של דף האינטרנט מבחינת טקסט, תמונות, טבלאות, עיצוב ועוד. זוהי שפת הקוד הבסיסית לסביבת ה-Web עבור בניית אתרים. HTML5 כוללת כלים חדשים שבאו להקל על עבודת המפתחים.

-Bootstrap היא סביבת עבודה בקוד פתוח לצד לקוח, שמכילה אוסף של כלים ליצירת אפליקציות Web ואתרים.

* 1. תיאור הארכיטקטורה הנבחרת- הסבר בכמה מילים מדוע.

הארכיטקטורה הנבחרת היא חלוקה ל-3 שכבות – 3 Tier Architecture.

בפרויקטים המבוססים על Database מומלץ להשתמש בגישה של מודל השכבות, בארכיטקטורה זו קיימת הפרדה בין השכבות השונות בפרויקט.

היתרון העיקרי בחלוקה לשכבות הוא תחזוקה קלה, במיוחד במערכות גדולות, כך שניתן יחסית בקלות להחליף כל שכבה בלי לגעת בשכבות האחרות כאשר נרצה לעשות שינויים בתוכנית.

* 1. חלוקה לתכניות ומודולים.

1. שכבת תצוגה (Presentation Layer)

שכבת ממשק המשתמש (**UI**- User Interface). שכבת התצוגה מדפי HTML באתר אינטרנט.  
השכבה מתקשרת בין המשתמש לבין שאר השכבות של המערכת.

1. שכבת הלוגיקה העסקית (**BL**- Business Logic)

השכבה שאמונה על הלוגיקה של המערכת ,עוסקת בעיבוד המידע, בחישובים שונים ושליחתו לשכבת התצוגה.

בשכבה זו נממש את הפונקציונאליות של המערכת.

1. שכבת הנתונים (**DAL**- Data Access Layer)

שכבה זו מורכבת ממקור נתונים –מסד הנתונים שלנו, וממערכת תוכנה Entity Framework אשר תפקידה לקרוא את המידע הנדרש למערכת, לשמור את העדכונים, ולהוסיף מידע חדש או למחוק פרטי מידע קיימים.

* 1. סביבת השרת

לצורך הפרויקט נשתמש בשרת מקומי–IISExpress ,

המסופק עם סביבת העבודה של Visual Studio. אם האתר יירכש על ידי לקוח, נעלה אותו לשרת אירוח כלשהו או – Microsoft Azure

* 1. ממשק המשתמש/לקוח – GUI.

שכבת ה GUI מורכבת מדפי HTML שמוצגים למשתמש דרך הדפדפן.

* 1. ממשקים למערכות אחרות/API - WebApi

* 1. שימוש בחבילות תוכנה.

JQuery, Bootstrap, CSS, Entity Framework, Rxjs

1. שימוש במבני נתונים וארגון קבצים -
   1. מבני הנתונים.

משתמשים: מזהה, שם, סיסמא, מייל, סוג, חזור, בסדר אקראי, ניוזלטר

שירים: מזהה, שם, קובץ, סוג, מזהה זמר, תגיות, תאריך הוספה, כמות צפיות

זמרים: מזהה, שם

רשימת השמעה: מזהה, מזהה משתמש, שם

שירים לרשימת השמעה: מזהה, מזהה רשימת השמעה, מזהה שיר

דיווח שירים: מזהה, שם, מייל, טלפון, מזהה שיר, הודעה, סטטוס

כתבות: מזהה, כותרת, קבצים, תוכן, תאריך, תגיות

תגובות: מזהה, מזהה כתבה, שם, כותרת, תוכן, תאריך

* 1. שיטת האחסון

שיטת האחסון היא ע"יSQL Server - מסד נתונים. הגישה למסד הנתונים היא ע"י Entities.

* 1. נפילה\קריסה\תמיכה

SQL Server בנוי לתמיכה במס' גדול של קריאות בו זמניות ואין חשש לקריסה ברמה של הפרויקט.

1. תרשימי מערכת מרכזיים



Use Case

****



1. תיאור המרכיב האלגוריתמי – חישובי

איסוף מידע וניתוחים סטטיסטיים (אנליטיות)

ללקוח יוצגו הזמרים הפופולאריים, נחשב זאת ע"פ מספר החיפושים לזמר.

ולמנהל יוצגו דיאגרמה המתארת את מידת הפופולאריות של כל ג'אנר לפי כמות הצפיות שלהם.

1. תיאור/התייחסות לנושאי אבטחת מידע

יש לדאוג לאבטחת השרת מעומסים מופרזים הנגרמים ע"י גורמים זדוניים,

ואת שרת ה- SQL Server מ-.SQL Injection

משתמש הגולש באתר מריץ אותה באופן אבסולוטי. קוד המשתמש ישלח לשרת ויצטרף לכל רשומה הנשמרת עבורו. לא יתכן מצב בו משתמש מריץ אפליקציה בתור משתמש אחר.

נא ציין מס׳ מקרים ותגובות להם ניתן מענה אבטחתי.

• כניסת משתמש על ידי כתובת מייל, המערכת מוודאת את תקינות כתובת המייל ושולחת קוד אימות לכתובת שהוזנה.

• במקרה שבעת כניסת משתמש הסיסמא אינה תואמת לכתובת המייל או לשם המשתמש שהוקש, תוצג הודעת שגיאה ולא תתאפשר התחברות למערכת.

.

1. משאבים הנדרשים לפרויקט:
   1. מספר שעות המוקדש לפרויקט, חלוקת עבודה בין חברי הצוות

מספר השעות המוקדשות לפרויקט הוא 700. חלוקה בין חברי הצוות הוא 350 שעות לכל אחת.

* 1. ציוד נדרש

CPU i5, RAM 8GB, HD SSDחיבור אינטרנט

* 1. תוכנות נדרשות

Visual Studio, SQL Server, IISExpress דפדפנים

* 1. ידע חדש שנדרש ללמוד לצורך ביצוע הפרויקט

הרחבת הידע בשפת Angular .

* 1. ספרות ומקורות מידע

<https://angular.io/>

<https://stackoverflow.com/>

<https://material.angular.io/>

<https://material.io/resources/icons>

1. תכנית עבודה ושלבים למימוש הפרויקט
2. ייזום רעיון, ניתוח מערכת, הכרת השטח, אפיון מסד נתונים
3. תכנון UI-UX , עיצוב גריד, זרימת מסכים
4. חקירת נושא עקומות גדילה, חישובים ויבוא טבלאות
5. כתיבת לוגיקה עסקית
6. כתיבת ממשק משתמש ועיצוב
7. בדיקות QA
8. התקנה והטמעה
9. תכנון הבדיקות שיבוצעו

Full flow

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| הערות | חשיבות | ידנית/ אוטומטית | מקרי הבדיקה | מס` דרישה במסמך אפיון | מספר בדיקה |
|  | בינונית | ידנית | בדיקת שדה כתובת מייל האם חוקי | 5.3.2 | 1 |
|  | גבוהה | ידנית | בדיקת התאמת סיסמא | 5.3.2 | 2 |
|  | בינונית | ידנית | בדיקה אם המשתמש סימן לייק לשיר או לכתבה | 5.3.9 | 3 |
|  | בינונית | ידנית | בדיקה אם המשתמש הצביע למצעד השירים | 5.3.3 | 4 |

1. בקרת גרסאות

יישמרו גרסאות במהלך הפיתוח (תלוי במידת ההתקדמות היומית), כל גרסה תיבדק בהשוואה לגרסה הקודמת. לכל גרסה נצרף תיאור המסביר את תוכן הגרסה והבדלים מגרסה קודמת.

|  |  |
| --- | --- |
|  |  |
| חתימת הסטודנט | חתימת המנחה האישי |

1. הערות ראש המגמה במכללה

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. C:\Users\טיירי\AppData\Local\Microsoft\Windows\INetCache\Content.Word\חתימה.pngC:\Users\טיירי\AppData\Local\Microsoft\Windows\INetCache\Content.Word\חתימה1.pngאישור ראש המגמה

|  |  |  |
| --- | --- | --- |
| שם: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | חתימה: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | תאריך: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

-----------------------------------------------------------------------------------------------------

1. הערות הגורם המקצועי מטעם מה"ט

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. אישור הגורם המקצועי מטעם מה"ט

|  |  |  |
| --- | --- | --- |
| שם: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | חתימה: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | תאריך: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |



תודות

בראשית כל – תודה לבורא עולם שהכל מידו ואלמלא רצונו לא היה הפרויקט יוצא לפועל!

תודה על הרעיונות, על הכלים והיכולת לממש אותם ולהוציא מהכוח אל הפועל.

תודה גם לרכזת תכנות – המורה זהבה ברון, לכל צוות מורות המסלול ולהנהלת הסמינר שעשו את המרב על מנת שנצליח ולהקנות לנו כלים.

למזכירה בת שבע על העדכונים והתזכורות,

למנחה - תמר שרעבי על העזרה בכל מהלך בניית הפרויקט

וכן למנחה רבקי דהאן – שעזרה לנו במאור פנים על אף שהיא לא המנחה שלנו.

תקציר

אתר 'מיוזיק אונליין' מנגיש מוזיקה יהודית בצורה הנוחה ביותר למשתמש ובנוי באופן ידידותי. 'מיוזיק אונליין' מאפשר למשתמש להתעדכן אונליין בכל חדשות המוזיקה בכלל ובסוג המוזיקה שהוא מעדיף בפרט.

בדף הראשי מציג האתר את כל החדשות והעדכונים החמים במוזיקה. שירים חדשים שיצאו לאחרונה, קליפים, כתבות חמות, הופעות מהתקופה האחרונה והפלייליסטים מבית 'מיוזיק אונליין' המושמעים ביותר. בנוסף האתר מעדכן את המשתמש בתגובות האחרונות שנכתבו ע"י משתמשים אחרים, בשירים ובכתבות שנצפו לאחרונה ובזמרים שאנשים חיפשו מספר פעמים רב ביותר.

בדף של 'חדש במוזיקה' יופיעו כל השירים עם אפשרויות חיפוש שיר מסוים או לפי זמר, מיון השירים לפי תאריך, שם שיר, כמות צפיות, כמות לייקים ועוד וכן לסננם לפי ג'אנר מסוים סגנון, נושא השירים כמו – שבת, חתונה, חגים וכו' וכן סינון ע"פ מילות מפתח כמו – קליפ, דואט, פיוט ועוד...

בדף ה'מגזין' מופיעות כתבות עדכניות על דברים חמים, טעימות מעולם המוזיקה, ראיונות של זמרים ואנשי מוזיקה ושידורים מאירועי חול המועד, ל"ג בעומר ועוד. עם אפשריות חיפוש מיון וסינון כמפורט לעייל.

בדף ה'פלייליסטים' יופיעו מגוון פלייליסטים מבית 'מיוזיק אונליין' בכל סגנון ולכל זמן.

בדף ה'מצעד' תהיה אפשרות להצביע ולדרג את עשרת השירים האהובים לשיר השנה וכן להצביע לזמר השנה, אפשרות ההצבעה תהיה חד פעמית ותתאפשר במידה והמצעד פעיל. כאשר המצעד יסתיים יוצגו בדף זה תוצאות המצעד לשנה זו.

בדף של שיר יופיע פרטי השיר עם הרייטינג שלו – מספר האנשים שצפו, כמות לייקים ומספר תגובות, בהמשך יופיעו תגיות הקשורות לשיר. בלחיצה עליהם האתר יציג את התוכן הרלוונטי לסינון שנבחר – שירים, קליפים, הופעות וכתבות. בהמשך יופיעו תגובות של משתמשים לשיר וכן אפשרות להוספת תגובה. בדף יופיעו גם מספר שירים דומים לשיר הנוכחי וכן סרגל כלים המאפשר לדווח על השיר במקרה של הפרת זכויות, הוספת מעקב לשיר וכן הוספת מנוי לזמר ע"י הזנת כתובת מייל במידה והמשתמש אינו מחובר לאתר. למשתמש מחובר תינתן אפשרות להוסיף את השיר לאחד מהפלייליסטים האישיים.

וכן בדף של הופעה וכתבה.

ובדף של פלייליסט יופיעו השירים של הפלייליסט עם אפשרויות השמעה נרחבות – השמעה בסדר אקראי, הפעלת השירים ברצף, הפעלה חוזרת בסיום השמעת כל השירים וכן אפשרות לקבוע סדר חדש להפעלתם.

באזור האישי האתר יציג למשתמש את הפלייליסטים האישיים עם אפשרות לערוך אותם, להעביר שיר בין פלייליסטים ואפשרויות נוספות. יופיעו גם השירים האחרונים שאהב, מידע אישי הכולל את השירים והכתבות במעקב והמנויים לזמרים וכן אפשרות עריכת הפרופיל.

באזור המנהל תינתן אפשרות לעריכת האתר. אפשרות הוספה, עריכה ומחיקה של שירים, כתבות, פלייליסטים. אפשרות למחיקת תגובות של משתמשים, הפעלת המצעד וסיומו. וכן יוצג אזור בו יופיעו הדיווחים להפרת זכויות יוצרים שנשלחו ע"י המשתמשים.

תוכן עניינים

1. הגדרת דרישות ותיאור כללי
   1. תיאור כללי
   2. תיאור חומרת המערכת
   3. תיאור תוכנת המערכת
   4. תיאור פונקציית המערכת
   5. זרימת המידע במערכת
2. ממשקים חיצוניים
3. ממשק אדם מכונה
   1. כללי
   2. תיאור מסכים
4. מבנה נתונים וארגון הקבצים
   1. ארגון הקבצים
   2. מבנה נתונים
5. תכנון
   1. כללי
   2. עקרונות תכנות
   3. תיאור אלגוריתמים
   4. בדיקת המערכת
6. מה הקנה הפרויקט
7. ביבליוגרפיה
8. הגדרת דרישות ותיאור כללי
   1. תיאור כללי

'מיוזיק אונליין' הוא אתר הנועד להנגיש מוזיקה יהודית בצורה נוחה וידידותית למשתמש ומאפשר להתעדכן בחדשות המוזיקה בכלל ובמוזיקה בהעדפה אישית בפרט.

* + 1. מטרות המערכת

לאפשר להתעדכן בחדשות המוזיקה

לאפשר להתעדכן בשירים, בקליפים ובהופעות החדשות

לאפשר לקבל עדכונים במייל על תוכן חדש לפי העדפת המשתמש

לאפשר יצירת פלייליסטים בהתאמה אישית

* + 1. היקף פרויקט

700 שעות

* + 1. מבנה הארגון

המערכת מתחלקת ל-3 חלקים

אזור כללי,

אזור אישי למשתמש רשום

ואזור מנהל לניהול ותפעול האתר.

* + 1. משימות המערכת
       1. משתמש באזור הכללי
* צפייה בשירים, קליפים, הופעות וכתבות חדשות
* הפעלת פלייליסטים
* דיווח על הפרת זכויות יוצרים לשיר
* הרשמה לאתר
* הרשמה לניוזלטר
* הוספת מעקב לשיר או כתבה
* הוספת מנוי לזמר
* הצבעה למצעד כאשר הוא פעיל
* צפייה בתוצאות המצעד
* צפייה בתגובות אחרונות
* צפייה בשירים וכתבות שנצפו לאחרונה
* צפייה בזמרים פופולאריים (לפי כמות החיפושים לזמר)
  + - 1. משתמש מחובר
* יצירה, עריכה ומחיקה של פלייליסט
* הוספה ומחיקה של שיר מהפלייליסטים האישיים
* העברת שיר בין פלייליסטים
* צפייה בשירים אהובים
* צפייה ברשימת מעקבים לשירים וכתבות וברשימת מנויים לזמרים
  + - 1. מנהל
* הוספה, עריכה ומחיקה של שירים כתבות ופלייליסטים
* מחיקת תגובות של משתמשים
* הפעלה, עריכה וסיום המצעד
* הוספת זמרים, אומנים ותגיות
* צפייה בדיווחים
* שינוי סטטוס לדיווח
  1. תיאור חומרה
     1. כללי

להפעלת המערכת נדרש מחשב עם מערכת הפעלה של windows החל מגרסה xp.

* + 1. מרכיבי המערכת

מחשב pc עם התאמה לדפדפן chrome.

* 1. תיאור תוכנת המערכת
     1. כללי
* מסד נתונים – הוא ב-SQL Server.
* שרת – מנוהל ע"י Web API. והגישה למסד הנתונים נעשית באמצעות DB First - Entity Framework. השליפות נעשות ע"י שאילתות LINQ.
* צד לקוח – נכתב בשפת Angular מתקדם ע"י שימוש ומיצוי יכולות של השפה.
  + 1. כלי התוכנה לפיתוח המערכת

Visual Studio, Visual Studio Code, SQL Server.

שפות תכנות - Angular, CSS, Html5, TS, C#.

* 1. תיאור פונקציות המערכת
     1. ניהול משתמשים
* כניסת משתמש ומנהל – הכנסת סיסמה, בדיקה אם קיים ושמירת הנתונים ב-Local Storage.
* הוספת חשבון משתמש – בדיקות תקינות הנתונים, בדיקה שלא קיים משתמש עם סיסמה או כתובת מייל זהים.
  + 1. צפייה בשירים חדשים

קריאה לפונקציה -public static List<songsDetails> GetSongs()

הפונקציה שולפת את השירים מטבלה מדומה המחזירה את פרטי השיר הכוללים את שם הזמר.

* + 1. חיפוש שירים, קליפים, הופעות וכתבות לפי פרמטר

קריאה לפונקציה –public List<ItemsByParameterDTO> GetItemsByParameter(string parameter)

הפונקציה מקבלת פרמטר לסינון – שם זמר, אומן או תגית ומחזירה את כל השירים, הקליפים, ההופעות והכתבות הרלוונטיות לפרמטר החיפוש מטבלה מדומה.

* + 1. צפייה בתגובות אחרונות של משתמשים

קריאה לפונקציה - public List<LastestResponsesDTO> GetLastResponsesToArticlesAndSongs()

הפונקציה שולפת ומחזירה תגובות אחרונות מטבלאות ResponsesToSongsTBL ו – ResponsesToArticlesTBL.

* + 1. צפייה בשירים, קליפים, הופעות וכתבות שנצפו לאחרונה

קריאה לפונקציה - public List<lastViews> GetLastViews()

הפונקציה שולפת ומחזירה את הפריטים שנצפו לאחרונה ע"י משתמשים.

* + 1. הוספת מנוי לזמר

קריאה לפונקציה - public static bool AddSubscription(SubscriptionTBL subscription)

הפונקציה מוסיפה מנוי למשתמש מחובר.

* + 1. הוספת מעקב לשיר או כתבה

קריאה לפונקציה - public static bool AddFollowUp(FollowUpTBL followUp)

מקבלת מזהה משתמש או כתובת מייל למשתמש שאינו מחובר וכן מזהה כתבה או שיר ומוסיפה.

* + 1. העברת שיר בין פלייליסטים

קריאה לפונקציה - public static bool MoveSongToOtherPlaylist(int songId, int prevPlaylistId, int currentPlaylistId)

מקבלת מזהה שיר, מזהה פלייליסט קודם ומזהה פלייליסט שאליו המשתמש רוצה להעביר.

* + 1. הוספת שיר
* קריאה לפונקציה - public static void AddSong(SongsTBL song, string singerName)

הפונקציה מוסיפה את השיר לטבלת השירים

* קריאה לפונקציה - public HttpResponseMessage UploadJsonFile(string folderName, string folderName2)

הפונקציה מקבלת את הקבצים (ע"י HttpResponseMessage) של השיר, התמונה והטקסט ושומרת בתיקיות המתאימות.

* קריאה לפונקציה - public static void AddTagsToSong(string[] tags, int songId)

הפונקציה מוסיפה את מערך התגיות לטבלת תגיות לשירים.

* קריאה לפונקציה - public static void AddArtistsToSong(ArtistWithJob[] artists, int songId)

הפונקציה מקבלת מערך של אובייקטים המכיל שם אומן ושם תפקיד. הפונקציה מוסיפה את האומנים לטבלת אומנים לשירים ומאתחלת מזהה תפקידים לפי המערך שקיבלה.

* קריאה לפונקציה - public static void AddArtistsToDuet(string[] singers, int songId)

הפונקציה מקבלת מערך של שמות הזמרים ומזהה השיר ומוסיפה לטבלת האומנים את הזמרים בתפקיד – דואט מלבד שם הזמר הראשון. את הזמר הראשון הפונקציה של הוספת שיר תקבל.

* קריאה לפונקציה - public static bool sendMailsToSubscriptionUsers(string image, string link)

הפונקציה מקבלת תמונה של השיר וקישור אליו ושולחת מיילים לכל המשתמשים המנויים לאחד מהזמרים המשתתפים בשיר.

* 1. זרימת המידע במערכת
     1. תרשים זרימה בסיסי על התנהגות המערכת –

שינוי סיסמה

דף שירים לפי חיפושים

דף כתבה

דף שיר

דף הבית

דף שירים

דף כתבות

דף פלייליסטים

הרשמה לאתר

הרשמה לניוזלטר

דף המצעד

אזור אישי

אזור מנהל

דף פלייליסט

* + 1. תרשים הטבלאות וקשרי הגומלין במודל –

**תמונה של הטבלאות**

1. ממשקים חיצוניים

האתר שולח הודעות Mail כל שבוע למשתמשים הרשומים לניוזלטר וכן בהעלאת שיר או כתבה למשתמשים מנויים לזמר המשתתף בשיר או הכתבה וכן בהוספת תגובה לשיר או כתבה שולח התראה למשתמשים במעקב.

1. ממשק אדם ומכונה
   1. כללי

האתר ידידותי למשתמש. הקפדנו על עיצוב נעים לעין ונוח לשימוש כדי לספק חווית משתמש מעולה

* 1. תיאור מסכים

**מסכים**

1. מבנה נתונים וארגון קבצים
   1. ארגון קבצים
      1. כללי

הנתונים נשמרים בצורה טבלאית בבסיס נתונים ב-SQL Server, כל המשתמשים שרשומים במערכת שמורים בטבלת משתמשים וכן כל השירים, הכתבות, הפלייליסטים, המצעד, התגובות, המעקבים, המנויים, הדיווחים, הזמרים, האומנים, התגיות, סוגי התגיות, חיפושים לזמרים, חיפושים של משתמשים שמורים בטבלאות ב-SQL.

* 1. מבנה נתונים

הנתונים נשמרים בתוך הטבלאות, השדות נשמרים בצורה טבלאית (לכל שדה יש שם שנשמר במערכת SQL Server) ולפי השמות של השדות נשלפים הנתונים.

**שירים:**

**כתבות:**

1. תכנון
   1. כללי

האתר מחולק לצד שרת וצד לקוח.

צד שרת נבנה ב-Web API בשפת C#.

צד לקוח נבנה בשפת TypeScript בארכיטקטורת אנגולר.

הפרויקט נבנה לפי מבנה הארכיטקטורות הנ"ל כדי להשתמש במירב היכולות ולהפיק את התועלת המרבית.

* + 1. עקרונות תכנות

הפרויקט נבנה בגישת OOP (Object Oriented Programming) – תכנות מונחה עצמים. זוהי פרדיגמה תכנות המשתמשת בעצמים (אובייקטים) לשם תכנון תכניות מחשב ויישומים. הפרדיגמה מספקת מספר כלי הפשטה וטכניקות ובהן הורשה, מודולריות, פולימורפיזם וכימוס.

* 1. תיאור אלגוריתמים

להלן מספר קטעי קוד במערכת –

* + 1. פונקציה המקבלת פרמטר – שם זמר, שם אומן או שם תגית ומחזירה את כל הפריטים – השירים, הקליפים, ההופעות והכתבות הקשורות לפרמטר ומוסיפה עמודה בשם type המכילה את סוג הפריט.

CREATE FUNCTION [dbo].[itemsByParameter] (@parameter varchar(50))

RETURNS table

AS

RETURN

(

select so.id, title, subtitle, date, null as dateHebrew, image\_location as image,

CASE WHEN isPerformance = 'true' THEN 'performance'

WHEN EXISTS (SELECT \* FROM SongsTBL as song join TagsToSongsTBL as tagts

on song.id = tagts.songId join TagsTBL as tag on tag.id = tagts.tagId

Where song.id = so.id and tag.name = 'קליפ') THEN 'clip'

ELSE 'song' END AS type

from SongsTBL as so join SingersTBL as si

on so.singerId = si.id where si.name = @parameter

union

select so.id, title, subtitle, date, null as dateHebrew, image\_location as image,

CASE WHEN isPerformance = 'true' THEN 'performance'

WHEN EXISTS (SELECT \* FROM SongsTBL as song join TagsToSongsTBL as tagts

on song.id = tagts.songId join TagsTBL as tag on tag.id = tagts.tagId

Where song.id = so.id and tag.name = 'קליפ') THEN 'clip'

ELSE 'song' END AS type

from SongsTBL as so join TagsToSongsTBL as tts

on so.id = tts.songId join TagsTBL as t on tts.tagId = t.id where t.name = @parameter

union

select so.id, title, subtitle, date, null as dateHebrew, image\_location as image,

CASE WHEN isPerformance = 'true' THEN 'performance'

WHEN EXISTS (SELECT \* FROM SongsTBL as song join TagsToSongsTBL as tagts

on song.id = tagts.songId join TagsTBL as tag on tag.id = tagts.tagId

Where song.id = so.id and tag.name = 'קליפ') THEN 'clip'

ELSE 'song' END AS type

from SongsTBL as so join ArtistsToSongsTBL as ats

on so.id = ats.songId join ArtistsTBL as a on ats.artistId = a.id where a.name = @parameter

union

select a.id, title, subtitle, date, dateHebrew, main\_image as image, 'article' as type

from ArticlesTBL as a

join TagsToArticlesTBL as tta on a.id = tta.articleId join TagsTBL as t on tta.tagId = t.id

where t.name = @parameter

)

* + 1. פונקציה המקבלת שם פריט – זמר, אומן או תגית ומחזירה את הפרטים שלו מהטבלה המתאימה.

create FUNCTION [dbo].[itemByName] (@item varchar(50))

RETURNS table

AS

RETURN

(

select top 1 id, name, image from (

select id, name, image from SingersTBL where name = @item

union

select id, name, null as image from ArtistsTBL where name = @item

union

select id, name, null as image from TagsTBL where name = @item) as X)

* + 1. טבלה מדומה השולפת את 20 התגובות האחרונות שנוספו לטבלאות – תגובות לשירים ותגובות לכתבות.

CREATE VIEW [dbo].[latestResponses]

AS

select top 20 id, type, name, date, topic, topicId, title\_res, content\_res from (

select r.id, 'song' AS type, r.name, r.date, s.title as topic, s.id as topicId, r.title as title\_res, r.content as content\_res

from ResponsesToSongsTBL as r join SongsTBL as s on r.songId = s.id

union

select r.id, 'article' AS type, name, r.date, a.title as topic, a.id as topicId, r.title as title\_res, r.content as content\_res

from ResponsesToArticlesTBL as r join ArticlesTBL as a on r.articleId = a.id) as X order by date desc

* + 1. טבלה מדומה השולפת את 20 השירים, הקליפים, ההופעות והכתבות שנצפו לאחרונה

CREATE VIEW [dbo].[lastViews]

AS

select top 20 \* from (

select id, title, subtitle, date, lastViewingDate, image\_location as image,

CASE WHEN isPerformance = 'true' THEN 'performance'

WHEN EXISTS (SELECT \* FROM SongsTBL as song join TagsToSongsTBL as tagts

on song.id = tagts.songId join TagsTBL as tag on tag.id = tagts.tagId

Where song.id = so.id and tag.name = 'קליפ') THEN 'clip'

ELSE 'song' END AS type

from SongsTBL so

union

select id, title, subtitle, date, lastViewingDate, main\_image as image, 'article' AS type from ArticlesTBL

) as X order by lastViewingDate desc, date desc

* + 1. מחלקה המטפלת בשמירת הקבצים לדוגמה - בהעלאת שיר.

public class UploadController : ApiController

{

[System.Web.Http.HttpPost]

[System.Web.Http.ActionName("UploadImage")]

public HttpResponseMessage UploadJsonFile(string folderName, string folderName2)

{

HttpResponseMessage response = new HttpResponseMessage();

var httpRequest = HttpContext.Current.Request;

if (httpRequest.Files.Count > 0)

{

int i = 0;

foreach (string file in httpRequest.Files)

{

++i;

var postedFile = httpRequest.Files[file];

var filePath = HttpContext.Current.Server.MapPath("~/UploadFile/" + postedFile.FileName);

filePath = AppDomain.CurrentDomain.BaseDirectory.Substring(0,

AppDomain.CurrentDomain.BaseDirectory.LastIndexOf("Server") - 1) + "\\DAL\\src\\"

+ (i == 1 ? folderName : folderName2) + "\\" + postedFile.FileName;

postedFile.SaveAs(filePath);

}

}

return response;

}

}

* + 1. פונקציה המקבלת מזהה שיר ומחזירה תוצאות דומות לפי תגיות

public static List<songsDetails> GetSimilarSongs(int songId)

{

MusicOnlineEntities et = new MusicOnlineEntities();

List<TagsDTO> tags = Casts.ToTagsDTO.GetTagsFromTagsToSong(TagsToSongsBL.GetTagsToSong(songId));

List<songsDetails> possibleSongs = new List<songsDetails>();

List<songsDetails> songs = new List<songsDetails>();

List<TagsDTO>[] hardTags = new List<TagsDTO>[necessity.Count];

List<TagsDTO> flexibleTags = new List<TagsDTO>();

int index = 0;

foreach (Necessity nec in necessity)

{

TagsDTO tagType = new TagsDTO();

List<TagsDTO> tagsType = new List<TagsDTO>();

if (nec.level == eNecessityLevel.HARD)

{

tagsType = TagsTypesBL.GetTagByType(tags, nec.tag);

if (tagType != null)

{

hardTags[index] = new List<TagsDTO>();

hardTags[index++].AddRange(tagsType);

}

}

else if (nec.level == eNecessityLevel.FLEXIBLE)

{

tagType = TagsTypesBL.GetTagByType(tags, nec.tag).FirstOrDefault();

if (tagType != null) { flexibleTags.Add(tagType); }

}

}

possibleSongs.AddRange(GetSongsByAllTypes(hardTags));

if (tags.Where(tag => tag.name == "ווקאלי" || tag.name == "אקפלה").FirstOrDefault() == null)

{

possibleSongs = clearVocalSongs(possibleSongs);

}

songs.AddRange(OrderSongsByTags(possibleSongs, flexibleTags.Select(tag => tag.name).ToList()));

bool isFound = songs.Remove(songs.Where(song => song.id == songId).FirstOrDefault());

return songs;

}

* 1. בדיקת מערכת
* מעבר על מקרי קצה אפשריים, תיקון בעיות, שיפורים ובדיקה חוזרת.
* ערכנו בדיקות תקינות על שדות הקלט ע"מ למנוע שגיאות אפשריות והכנסת ערכים שגויים.

במהלך בניית האתר ופיתוחו ערכנו שיפורים ותיקונים כדי לייעל את האתר ולמנוע אפשרות לשגיאות ונפילות.

1. מה הקנה הפרויקט

* ניסיון בתכנון ואפיון מערכת מהבסיס ועד שלב הגמר.
* הבנת התקשורת בין צד שרת לצד לקוח.
* ניסיון מעשי בבניית שרת Web API.
* ניסיון מעשי בעבודה עם Entity Framework.
* ניסיון מעשי בבניית מסד נתונים ב-SQL Server.
* ניסיון מעשי בבניית פרויקט Angular.
* ניסיון מעשי בעבודה ובשימוש בספריות.
* הכרות מעמיקה בכתיבת דפי html ו-CSS.
* פיתוח יכולת חזקה של לימוד עצמי
* פיתוח יכולת חשיבה לוגית.
* ניסיון חיפוש בגוגל ויכולת מציאת פתרונות לבעיות באופן עצמאי.

1. ביבליוגרפיה

<https://angular.io/>

<https://stackoverflow.com/>

<https://material.angular.io/>

<https://material.io/resources/icons>