

המחלקה להנדסת תוכנה

דמיון שירים על פי תגים

MUSIC SIMILARITY

חיבור זה מהווה חלק מהדרישות לקבלת תואר ראשון
בהנדסה

מאת

אבי כהן

שגיא מרסיאנו

המחלקה להנדסת תוכנה

פרויקט גמר – תשע"ח

דמיון שירים על פי תגים

MUSIC SIMILARITY

חיבור זה מהווה חלק מהדרישות לקבלת תואר ראשון

בהנדסה

מאת

אבי כהן

שגיא מרסיאנו

תאריך:

אישור:

מנחה אקדמי: דר' מרים אללוף

תאריך:

אישור:

רכז הפרויקטים: דר' אסף שפיינר

תקציר

פרויקט גמר זה נעשה בשיתוף עם מיזם תמרינגה. תמרינגה הינה פלטפורמה היוצרת גירוי ושמע מותאמים אישית לחולי אלצהיימר ובכך משפרת את מצבם. במהלך הפרויקט, תכננו ובנינו מערכת המלצת שירים על פי קריטריונים ודמיון בין משתמשים עבור תמרינגה. הרחבנו את מאגר השירים ע"י שליפות של תכני מדיה לפי שנה וארץ מקור, ממאגרים חיצוניים כגון YouTube ואחסנו את המידע ב-Databases יעודי שנבנה כתשתית למערכת ההמלצה. מערכת ההמלצה מתבססת על מאגרי מידע של מוזיקה שיתופיים ברשת ותציג עבור המשתמש את השירים המתאימים לו וקישורי השמעה בתוך המערכת, ללא יציאה לאתר חיצוני. המערכת תחשב התאמה בין משתמשים על סמך מאפיינים והעדפות השמעה ובכך תמליץ עבורו על שירים בעתיד. בחיבור זה, נציג את מערכת ההמלצה שמומשה, את תכנונה ואת המסקנות שקיבלנו לאחר מחקרים רבים שהיו על נושאים שונים.

קישורים למערכות ניהול הפרויקט ובקרת תצורה :

#	מערכת	מיקום
1	מאגר קוד	https://github.com/avicochen89/Music-Similarity
2	יומן	https://calendar.google.com/calendar/embed?src=pnu92pav3s91nku9u35gks3m0c%40group.calendar.google.com&ctz=Asia%2FJerusalem
3	סרטון גרסת אלפא	https://youtu.be/Kuv0vKDEa68
4	סרטון פרויקט גמר	https://youtu.be/N-pBX8gyUw4

הצהרה

**הפרויקט נעשה בהנחיית ד"ר מרים אללוף במחלקה להנדסת תוכנה, עזריאלי -
המכללה האקדמית להנדסה ירושלים.**

החיבור מציג את עבודתנו האישית ומהווה חלק מהדרישות לקבלת תואר ראשון

בהנדסה.

היקף הפרויקט גדול ומיועד לשני סטודנטים באישור המנחה האקדמי.

תודות

נבקש להודות לאנשים שעזרו לנו במהלך ביצוע פרויקט הגמר :
לד"ר מרים אללוף, על הנחיית הפרויקט. תודה על תמיכה, עזרה, הקשבה וסבלנות
בלתי נדלת. ידעת לתכנן את הפרויקט עוד בראשית ימיו ולכוון אותנו בהתאם. היית
אוזן קשבת וידעת לעזור ברגעים שהיינו צריכים הכוונה.

לאה כהן סבן וסטלה מליצר, מייסדות מיזם תמרינגה. תודה על שיתוף פעולה מדהים,
פרגון ואוזן קשבת. ידעתן להזכיר לנו כל הזמן על חשיבות ותרומת הפרויקט לחולי
האלצהיימר. הפכתן פרויקט גמר של סטודנטים לתרומה גדולה ועל כך תודה.

לד"ר אסף שפנייר, רכז הפרויקטים. תודה על הסדר והארגון. היית זמין לכל שאלה
וקשוב לצרכינו.

פרויקט גמר של התואר היה דבר שחששנו ממנו, מפאת מעמדו והיקפו. בזכות
השותפים לעשייה, ד"ר מרים וד"ר אסף, לאה וסטלה, הפך הפרויקט שממנו חששנו
כל כך לחוויה מדהימה. החל מבחירתנו, בהמשך כעשייתנו ולבסוף הגשתו. למדנו הרבה
מכל הנוגעים בדבר והרגשנו שאנחנו עושים דבר משמעותי שיגרום נחת והקלה לחולי
אלצהיימר.

תודה רבה

תוכן עניינים

8	1. מבוא
8	2. תיאור הבעיה
10	2.1 אפיון המערכת ומסכים
12	2.2 הבעיה מבחינת הנדסת תוכנה
13	3. תיאור הפתרון
14	3.1 מבנה DATABASE של MONGODB
17	3.2 גישה למאגרים חיצוניים
17	3.3 אלגוריתם ההמלצה
20	3.4 תיאור הכלים המשמשים לפתרון
20	4. תכנית בדיקות
24	5. סקירת עבודות דומות בספרות והשוואה
23	6. סיכום ומסקנות
25	7. נספחים
25	7.1 תרשימים וטבלאות
26	7.2 תכנון הפרויקט
27	7.3 טבלת סיכונים
28	7.4 רשימת טבלות דרישות
28	7.5 רשימת ספרות וביבלוגרפיה

מילון מונחים, סימנים וקיצורים

- YouTube הוא אתר אינטרנט לאחסון ושיתוף של סרטוני וידאו.
- MusicBrainz הוא מאגר מוזיקלי אינטרנטי שמבוסס על קוד פתוח. המאגר מכיל מידע על מוזיקה, ומאפשר להשתמש במידע זה באופן ציבורי. מופיע גם בראשי תיבות MB.
- Last.fm הוא אתר שמספר רדיו אינטרנטי ורשת חברתית שעיקרה הוא מוזיקה. האתר מכיל מידע רב על מוזיקה ומאפשר להשתמש במידע זה באופן ציבורי.
- Discogs הוא אתר ומסד נתונים המכיל מידע על הקלטות אודיו. האתר מתמחה במוזיקה אלקטרונית. האתר מאפשר קניה ומכירה של מוצרי מוזיקה שונים (דיסקים, תקליטים וכדומה).
- MongoDB בסיס הנתונים בו נשתמש בפרויקט. מבוסס NoSQL.
 1. Collection טבלה בבסיס הנתונים
 2. Document שדה בתוך הטבלה בבסיס הנתונים
 3. Indexing פעולה אשר מסדרת את הנתונים בטבלה בסדר עולה או יורד על פי פרמטר מסוים. פעולה זו גורמת לשליפה מהירה של נתונים מהטבלה.
 4. Clustering פעולה אשר מחלקת את הנתונים הטבלה לפי קבוצות על פי פרמטר מסוים.
- Cosine Similarity [8] הינו מדד של דמיון בין שני וקטורים. חישוב זה משמש למדידת דמיון בין משתמשים על סמך הדירוג שלהם במערכות המלצה. בפרויקט נעשה שימוש בחישוב זה על מנת לחשב דמיון בין משתמשים שהצביעו עבור אותם שירים. מידע מפורט יותר נמצא בחלק 5.1 ספרות וביבילוגרפיה עם קישור לספר ופרק מתאים.
- CF [10] הינו מנגנון בו מעריכים את הדמיון בין משתמשים על סמך ההיסטוריה הקשבה שלהם בעבר ולהמליץ על השירים למשתמש דרך העדפתם של משתמשים דומים

1. מבוא

מחלת האלצהיימר היא מחלה קשה וחשוכת מרפא, המתבטאת בניוון איטי ומתמשך עד מוות של תאי מוח ומשפיעה על מערכת העצבים המרכזית. המחלה נפוצה בייחוד בקרב קשישים. המחלה מתאפיינת בירידה קוגניטיבית ובייחוד הפגיעה בזיכרון.

למוזיקה יש כוח מיוחד המשפיע על אנשים החולים באלצהיימר ודמנטיות קשורות. השפעת המוזיקה מראה תוצאות משכנעות גם בשלבים מאוחרים ביותר של המחלה. כשמבוצע בטיפול זה שימוש ראוי, מוזיקה יכולה לשפר את מצב הרוח, לטפל בתסיסת מתח מושרה, לעורר מגעים חברתיים חיוביים, להקל על התפקוד הקוגניטיבי ולשפר תנועות מוטוריות [12].

טיפול בקשישים החולים באלצהיימר, הביאה את לאה כהן סבן וסטלה מלינצר לפתח את תמרינגה. תמרינגה הינה פלטפורמה היוצרת גירוי ושמע מותאמים אישית לחולי אלצהיימר ובכך משפרת את מצבם.

מטרת הפרויקט הייתה לבנות מערכת המלצה עבור תמרינגה, שתתאים עבור כל משתמש את התכנים המתאימים על פי מאפייניו שהם גיל וארץ מוצא. המערכת ריכזה, מתוך מאגר אינטרנטי שיתופי, שירים על פי פרמטרים כגון שם אמן, שם השיר, שנת הוצאה, איזור גיאוגרפי, פופולאריות ותגים המאפיינים את השיר וסגנונו, ומציגה את השירים המתאימים ביותר לכל משתמש על פי גילו וארץ מוצאו, ובעלי דירוג פופולאריות גבוה. בנוסף המערכת מקשרת בין משתמשים דומים שאהבו את אותם שירים.

ישנם הרבה מאגרים המכילים מטה-דטה של שירים ומידע עליהם, אך אין מאגר שמרכז את כל המידע שנחוץ לפרויקט כגון שם אמן, שם השיר, שנת הוצאה, איזור גיאוגרפי, פופולאריות ותגים המאפיינים את השיר וסגנונו ובצורה מסודרת ומקוטלגת. ולכן, נתקלנו בקושי רב לאסוף מידע על תכני מוזיקה וביצענו פניות למספר מאגרי מידע שונים בעלי ממשקים ותוכנה שונים ובנינו את Databases הממשש תשתית למערכת ההמלצה.

2. תיאור הבעיה

נרצה ליצור מערכת המלצה נוחה וידידותית למשתמש, כזאת המיועדת להשמעת תכני מוזיקה לאוכלוסיה ספציפית. מטרת הפרויקט הינה שלילת שירים למשתמש על פי גילו וארץ מוצאו, בעלי דירוג פופולאריות גבוה. תחילת העבודה על הפרויקט החלה במחקר מעמיק על מאגרים המכילים מידע על תכני מוזיקה, אשר במטרה לשמש תשתית למערכת ההמלצה. הצורך שלנו במידע מגוון, המכיל מידע מפורט כגון שם המבצע, שם השיר, שנת הוצאה, תגים המאפיינים את השיר, דירוג פופולאריות, החל מאמצע שנות ה-50 התגלה כבעייתי ונמצאו סתירות וחוסרים במידע רלוונטי. לצערנו אין תיעוד מספק מהו המידע הניתן ע"י המאגר ולכן יש ראשית ללמוד את ממשק הגישה התכניתי (API) הייחודי לכל מאגר.

מאגרי המידע הקיימים שאותם חקרנו הם:

- YouTube
- MusicBrainz
- Last.fm
- Discogs

פירוט למחקר מתואר בפסקאות הבאות:

מחקר ראשוני החל בפלטפורמה המוכרת והשימושית ביותר כיום, YouTube. מחקר מעמיק ב- YouTube API [3] מראה כי הוא לא מספק שדות מטאדאטה אשר יכולים לסייע באפיון תוכן המדיה ובניית מערכת ההמלצה. ראינו כי התכנים אינם מאופיינים בסגנון מוזיקלי, או שנת הוצאה ומכילים בעיקר שם מבצע ושם השיר ותוכן מילולי שהכניס המשתמש שהעלה את השיר לאתר. התכנים בעיקר מסווגים על פי סוג התוכן ברמת סרט או שיר. הבנו כי YouTube לא נותן מענה לדרישתנו והרחבנו את המחקר והחיפוש לפלטפורמות פחות מוכרות וחדשות לנו.

דרישה נוספת שלנו בפרויקט היא היכולת להציג ולנגן, תוך כדי הישארות במערכת שלנו ולא לצאת למערכת חיצונית, כל שיר שעלה כתוצאה מהרצת המערכת, וברור כי YouTube היא הפלטפורמה הראשית לכך, מכוון שהיא המערכת הגדולה ביותר ומכילה תכנים רבים. בתחילת המחקר לא נמצא הדרך לקבלת לינק לשיר לפי שם השיר, מאחר והשירים מאופיינים ב-ID ייחודי ל-YouTube. במהלך חודש אוקטובר יצא לאוויר 3 YouTube API שאחד מחידושיהו הינו קבלת רשימת ID של שירים על פי חיפוש מסוים. חידוש נוסף הוא קבלת לינק להצגת השיר לפי ה-ID מסוים. ובכך בעצם התגברנו על הקושי והתאפשרה לנו היכולת לנגן כל שיר שיעלה בחכנתו.

התחנה הבאה הייתה מאגר המוזיקה והמידע MusicBrainz (MB). מחקר מעמיק ב-MusicBrainz מראה כי הוא מכיל מידע רב אודות תוכן מוזיקה, אך לאחר חקירה ממושכת ב-MusicBrainz API [6] התגלו קשיים רבים בחיפוש מידע ספציפי (חיפוש לפני שנה או איזור גיאוגרפי) ונמצא מידע כפול וסותר כגון שנת הוצאה ומיקום. חקירה אינטנסיבית הביאה אותנו למסקנות הבאות. ראשית כל, תוכן מדיה (שיר) מכיל שנת הוצאה ומיקום גיאוגרפי על פי שנת הוצאת האלבום המכיל את השיר והמיקום הגיאוגרפי. הסיבה העיקרית לכפילות שנת הוצאה ומיקום היא הוצאות אלבומי להיטים או אלבומים המכילים שירים מתקופות שונות, ולכן לכל שיר קיימים מספר רב של שנות הוצאה ומיקום. ולכן, על פי בדיקות רבות ואימות הנתונים בגוגל על כל שירים רבים, הבנו כי הדרך הטובה ביותר להתמודד עם הבעיה ולקבל את המידע המדויק ביותר היא לקחת את שנת ההוצאה הראשון שמופיע ברשימה. בנוסף גלינו כי MusicBrainz אינו מכיל דירוג פופולאריות אודות תוכן מדיה, מידע שחשוב מאוד לפרויקט. הבנו כי לא ניתן להסתפק אך ורק ב-MusicBrainz במקור ראשי למידע ונצטרך לדעת לשלב בין מאגרים שונים.

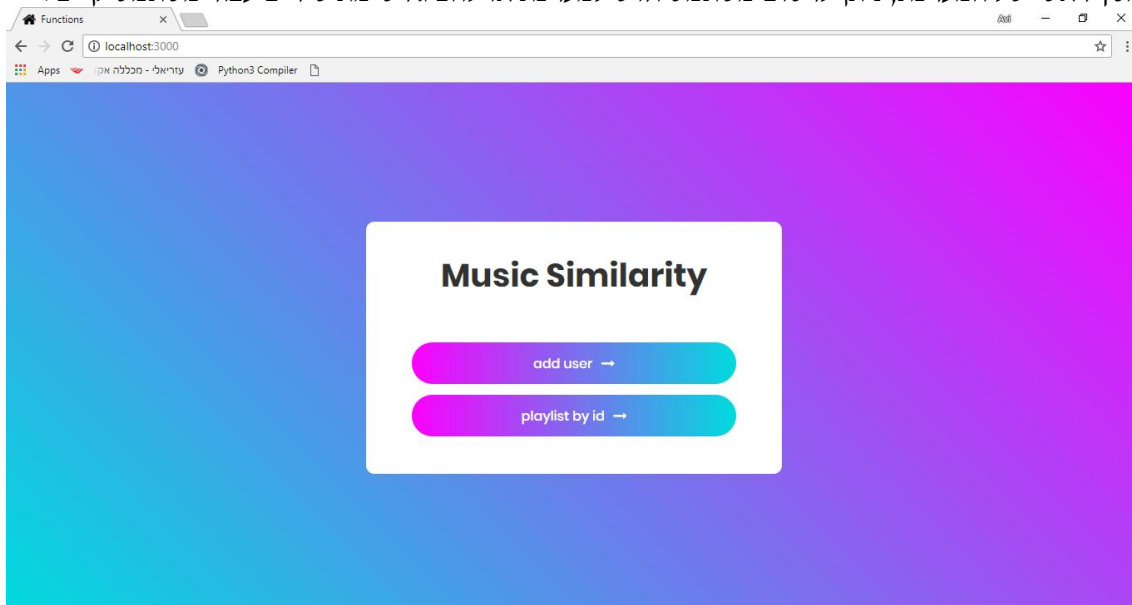
בעיה נוספת שנתקלנו איתה במהלך העבודה עם MusicBrainz, הייתה שלאחר 16 בקשות של השרת לקבלת מידע, החיבור היה מסתיים והעברת המידע נפסקה. בכל בקשה היה ניתן לקבל 100 שירים, ולאחר 16 בקשות היה ניתן לקבל 1600 שירים. המטרה שלנו הייתה לקבל מספר גבוה של שירים שיכול להגיע גם ל-300,000 שירים בכל פעם שאנו נגשים ל-MusicBrainz (שירים שיצאו שנה מסוימת/מיקום גיאוגרפי). על מנת להתמודד עם הבעיה הורדנו ושמרנו בקבצים את כל המידע על השירים מהשנים 1880 עד 2018. הנתונים נשמרו בקבצי json המכילים 100 שירים בכל קובץ, לאחר מכן העברנו את כל המידע לתוך שרת המונגו עם המידע שהיה רלוונטי לנו. עדיין היה חסר מידע מסויים (כמו דירוג השיר או מספר צפיות, והיכולת להשמיע אותו) והשלמנו את המידע מה-YouTube שהוא מס מזהה של הוידאו על מנת לנגן אותו במערכת, מספר צפיות ותגים.

מחקר נוסף ביצענו במאגר המוזיקה והמידע Last.fm. בדומה ל-MusicBrainz, מציג Last.fm מידע על שירים מתקופות שונות. לשלב הזה במחקר ידענו כי עלינו לחפש מידע כגון שם מבצע, שם שיר, שנה, תגים אותו השיר, איזור גיאוגרפי ודירוג פופולאריות. בשונה מ-MusicBrainz, מאגר המידע Last.fm מציג מצעד שירים באמצעות API לפי תקופה זמן מסוימת, אך בדיקה ידנית עבור מצעד השירים שקיבלנו התברר כי השירים לא עדכניים ולא מציגים את הלהיטים עבור אותה תקופה. בדיקה מעמיקה בפורומים שונים הביאה לידיעתנו כי מאגר המידע לא מעודכן מאז 2015. הוחלט לזנוח אותו ולהמשיך במחקר.

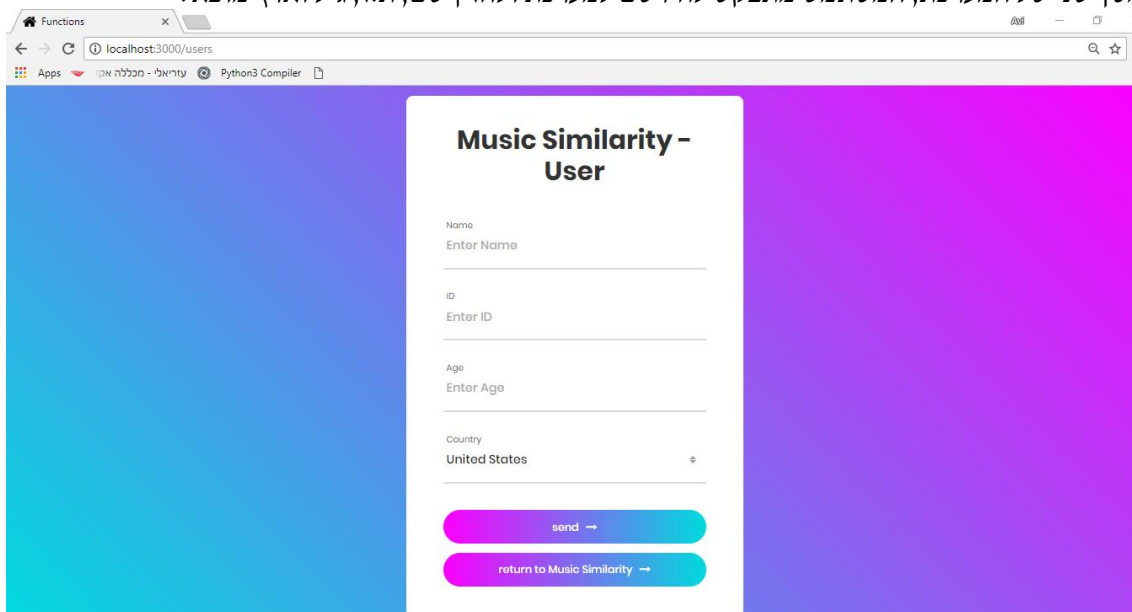
מחקר נוסף ביצענו במאגר המידע והמוזיקה Discogs. בדומה לקודמיו, ה-Discogs מספק מידע רב על שירים, כגון שנות הוצאה ומידע על חברות תקליטים שונות. האתר מתחב בעיקר במוזיקה אלקטרונית, ואנו נרצה מערכת המלצה שתשלט במגוון רחב של מוזיקה. בנוסף, ראינו כי המידע מקוטלג על פי אלבומים שלמים, וכי אנו נרצה לקבל מידע אודות שיר ספציפי. דבר נוסף שפסל את השימוש בפלטפורמה זאת במערכת שלנו היה כי המידע אינו מקוטלג על פי מיקום גיאוגרפי, דבר שמאוד קריטי למטרת הפרויקט, היות והקריטריון הראשי הוא מיקום גיאוגרפי, בנוסף לשנת הוצאת השיר.

2.1 אפיון המערכת ומסכים

בנינו מערכת בעלת ממשק web. המערכת מאפשרת קבלת נתונים מהמשתמש ומציגה עבורו רשימת שירים מומלצת בעקבות אלגוריתם המלצה המתבסס על פי שליפת תכני מוזיקה על פי גיל וארץ מוצא בעלי מדד פופולאריות גבוה ביותר ועל פי העדפות משתמשים בעלי מאפיינים דומים. המשתמש יכול להאזין לתכנים, לקבוע אם אהב או לא ועל פי מידע זה, המערכת מתאימה למשתמש שירים נוספים בעתיד. מסך ראשי של המערכת, ניתן לרשום משתמש חדש למערכת או להציג רשימת שירים עבור משתמש קיים:



מסך שני של המערכת, המשתמש מתבקש להירשם למערכת ולהזין שם, ת.ז, גיל וארץ מוצא:

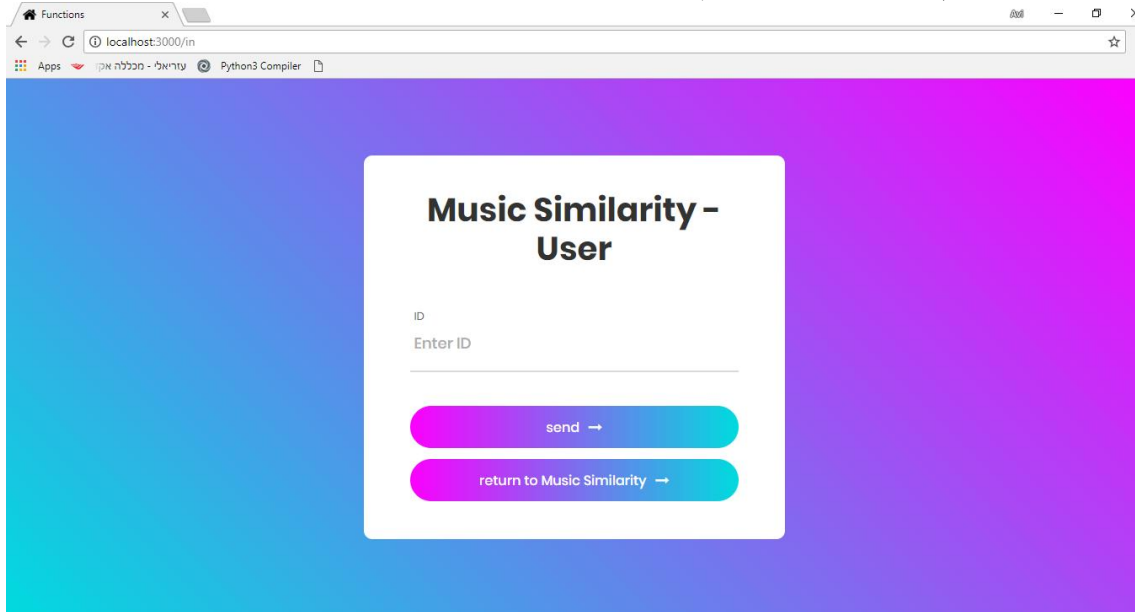


לאחר לחיצה על כפתור Send, מתבצע מאחורי הקלעים:

המשתמש ירשם ב-database. התכנית מבצעת clustering של המשתמשים הרשומים במערכת ומחלקת אותם לקבוצות על פי גיל וארץ מוצא.

עבור כל קבוצה, אלגוריתם ההמלצה ישלוח מה Databases של 25 שירים (להלן : Playlist) בעלי דירוג ההשמעה הגבוה ביותר לפי שנה וארץ מוצא. השליפה מתבצעת על סמך המידע המאוכסן ב-DATABASE ושהובא מ-MusicBrainz ומכיל מידע על כל שיר כולל שנה וארץ מוצא. דירוג ההשמעה נקבע עפ"י מידע רלוונטי ב-YouTube.

בשלב הבא המשתמש ייכנס למערכת כדי לקבל את רשימת ההשמעה המתאימה לו (כפי שנראה במסך הבא בו המשתמש מתבקש להזדהות על ידי ת.ז, בלחיצה על כפתור Send).



בפעם הראשונה שהמשתמש נכנס למערכת ישנה ההצגה הבאה :

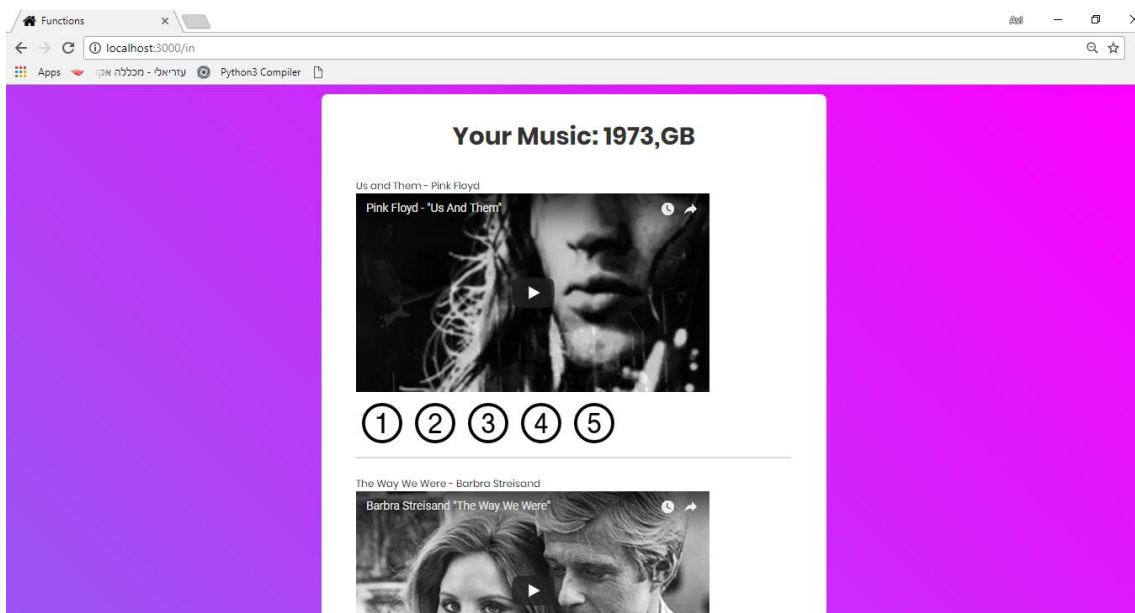
בראש העמוד ניתן לראות את השנה בה המשתמש היה בן 20 (בדוגמא זו 1973) ובשנה זאת התבצע חיפוש של שירים, כמו כן מוצג גם ארץ המוצא (Great Britain –GB).

המערכת תציג למשתמש בצורה רנדומאלית 10 שירים מתוך playlist (המכיל 25 שירים), בצירוף קישור לצפייה/השמעה ב-YouTube, בלי לצאת מהמערכת. בנוסף, יוצג שם השיר ושם האמן.

האופן שבו השירים מוצגים במערכת היא בעזרת שימוש ב-YouTube API ועל ידי כך שאנו מחזיקים ב-DATABASE את ה-ID של השיר ב-YouTube ושולחים בקשה בכל פעם לצפות/להאזין לשיר מסויים. המשתמש יוכל לדרג את השירים בסקלה של 1-5. 5 שיר שאהב/מכיר, 1 לא אהב/לא מכיר. הדירוג נכנס ל-DATABASE ונשמר עבור כל משתמש ועבור כל שיר.

כאשר המשתמש מדרג את רשימת השירים שהוצגה בפניו, מתבצע חישוב cosine similarity שמטרתו היא לחשב את הדמיון בינו לבין שאר המשתמשים בקבוצה. Cosine similarity [8] הינו מדד של דמיון בין שני וקטורים שונים מאפס ומודד את גודל הזווית ביניהם. חישוב זה משמש למדידת דמיון בין משתמשים על סמך הדירוג שלהם במערכות המלצה שונות. בפרויקט נעשה שימוש בחישוב זה על מנת לחשב דמיון בין משתמשים שהצביעו עבור אותם שירים.

ב-DATABASE נשמר החישוב והוא משמש את אלגוריתם ההמלצה כאשר המשתמש ישוב למערכת בפעם השניה והלאה :



בפעמים הבאות שהשתמש נכנס למערכת:
למשתמש יוצגו שוב רשימה של 10 שירים מתוך playlist, אך בשונה מהכניסה הראשונה, השירים כעת הנשלפים מאלגוריתם ההמלצה הם: 3 שירים ראשונים הם שירים שהשתמש אהב.
4 שירים הבאים הינם שירים שהומלצו לו על ידי משתמשים נוספים בקבוצה. כלומר, לאחר חישוב \cosine בין המשתמש לשאר הקבוצה, האלגוריתם מסתכל על השירים שמשתמשים בעלי ציון ה-similarity הגבוה ביותר אהבו ועל כן ממליץ את אותם שירים עבור המשתמש הספציפי.
3 שירים נוספים יהיו שירים שהשתמש לא צפה עוד או לא דירג מתוך Playlist.

2.2 הבעיה מבחינת הנדסת תוכנה

הפרויקט מספק אתגרים בתחום מערכת המלצה, המבוססות על קשר בין משתמשים דומים ושלילת תכנים רלוונטיים על פי פרמטרים. על מנת לבנות מערכת המלצה שתעבוד בצורה מדויקת ויעילה, קיים צורך להבין איך לקשר בין משתמשים ותכנים דומים, ועולות השאלות הבאות:
כיצד נשלף מידע רלוונטי ומדויק?
על פי אילו מאפיינים ניתן יהיה לבסס קשר בין משתמש אחד לאחר?
כיצד לזהות קשר בין אישיות שונות על פי תכנים דומים?

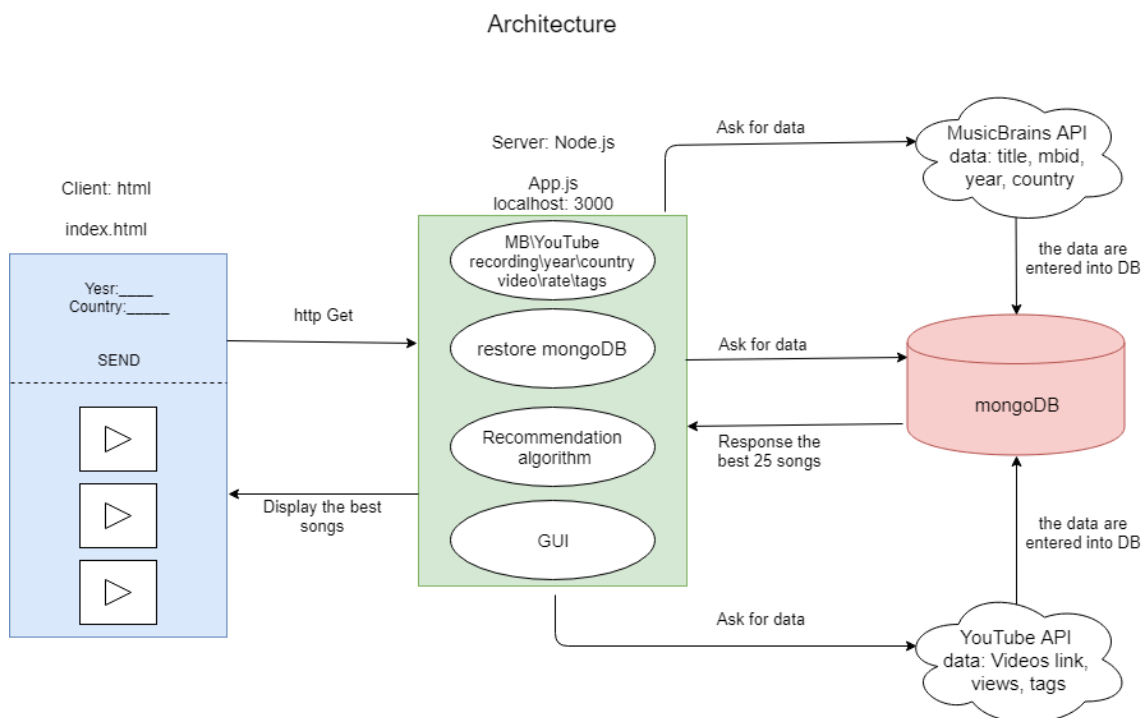
לאחר חקירה מעמיקה בארבעת מאגרי מידע, הבנו כי ה-data source הקיים באינטרנט אינו מוצג בצורה ברורה ומ"מוכנה לשימוש", המלווה במידע כפול וסותר. הבנו כי אין ברירה, ועל מנת להכניס מידע מדויק ככל האפשר עבור תוכן מדיה database שלנו, יש צורך לברור מתוך כל המידע שקיבלנו ולהבין: האם יש תבנית קבועה ששם נמצא שנת ההוצאה והמיקום הראשוני של השיר? איך להתעלם מהוצאות שונות בשנים מאוחרות יותר? הבדיקה נעשתה לעיתים על ידי בדיקה ידנית וחיפוש באינטרנט, חיפוש תקופות ואזורים שנכונים לתקופתנו, ואף נעזרנו באנשים קרובים בגילאים שונים, ממוצאים שונים על מנת לאמת את המידע.

לאחר מחקר מעמיק ב-API של הפלטפורמות ה"ל, הוחלט לשלב ולהשתמש ב-MusicBrainz ו-YouTube על מנת לבנות Database שמכיל מידע רלוונטי עבור מערכת ההמלצה. נותר לנו לאפיין את מבנה ה-Database ולקבוע את ה-data model כך שנאכסן ונשלף את המידע מתוך Databases בצורה מהירה ויעילה. כמו כן, נרצה לדעת באיזה אופן נקשר בין תכנים המזהים כדומים על פי תגים מסוימים, איך לשמור את המידע ה"ל ב-Database ולנצל את הדמיון בניהם כדי לקבל המלצות טובות עבור המשתמשים. חשיבה רבה על data model, והתייעצות עם אנשים מנוסים הוחלט על מבנה databasen ומבנו מוצג בהמשך. נשתמש ב-database של mongoDB [1].

הבאנו כי עיקרה של המערכת היא לנסות להבין מה היו הלהיטים הפופולאריים עבור המשתמש בשנות ה-20 לחיו בארץ מוצאו. מחשבה זאת גרמה לנו להבין שהמאפיינים של המשתמש, גיל וארץ מוצא הם המאפיינים שלפיהם ניתן לקשר בין משתמשים מאחר וחיו באותה תקופה והיו חשופים לאותה מוזיקה. בנוסף על כך, ראינו שניתן לזהות קשר בין שני משתמשים זהים אם הם אהבו את אותם שירים ובכך להסיק כי לא רק הקשר התקופתי מקשר בניהם, אלא גם הסגנון המוזיקלי. מנגנון זה נקרא סינון שיתופי CF [10] אשר מחשב דמיון בין משתמשים ומציע למשתמש המלצות על פי משתמשים שדמו לו.

3. תיאור הפתרון

בפרויקט בנינו מערכת המלצה בעלת 3 שכבות פיזיות, המורכבת ממשק web, שרת פיזי המריץ את אלגוריתמי ההמלצה ושרת וירטואלי שמכיל את Databases ב-MongoDB. להלן תרשים הארכיטקטורה של המערכת:

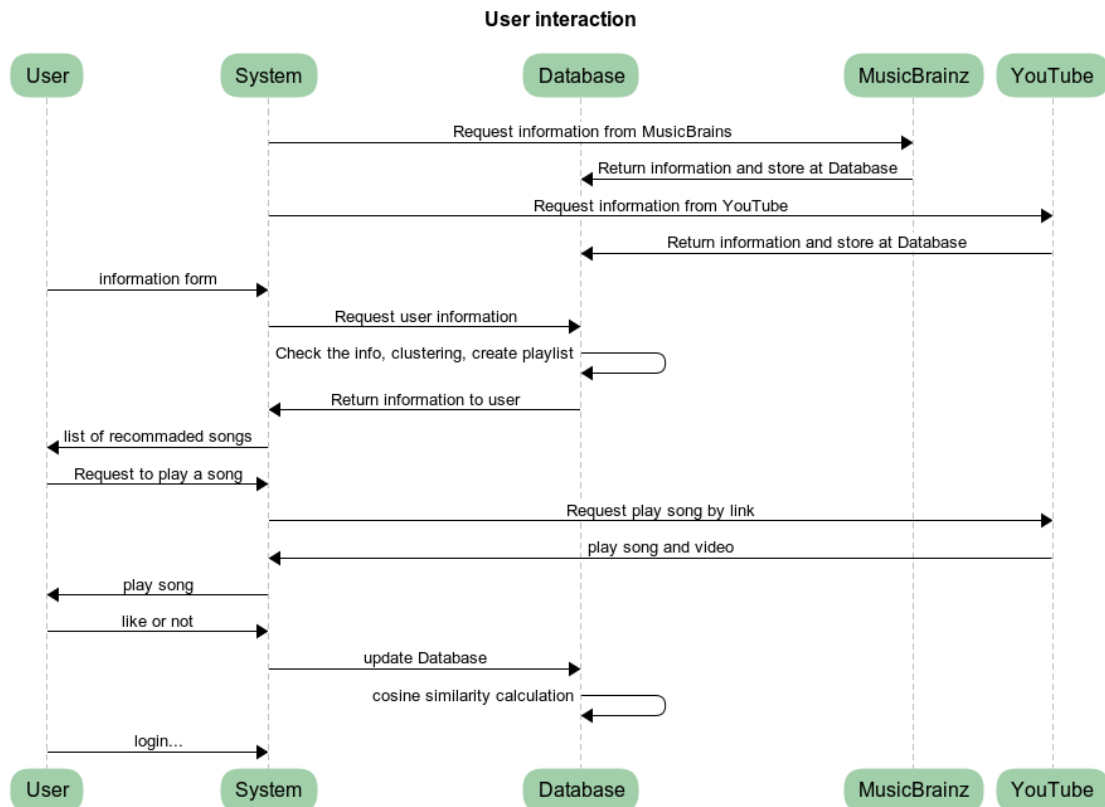


בצבע האדום ניתן לראות Databases של הפרויקט המבוסס mongoDB שמכיל מידע שאנו שואבים מ-YouTube ו-MusicBrainz. הוא מכיל מידע על כל שיר כגון שם השיר, אמן, ארץ ושנת הוצאה, מספר צפיות, תגים ולינק ל-YouTube. כחול, צד לקוח המבוסס html, javascript ו-jquery והוא מבצע פניות לשרת, שמתקשר ב-mongo ו-node.js עם ה-Database. צד השרת, בירוק, האחראי על התקשורת בין הצד לקוח ל-Database, מכיל בתוכו 4 פונקציות עיקריות:

- גישות למאגרים חיצוניים- קבלת המידע מ-YouTube ו-MusicBrainz
- אחסון הנתונים וניהול התוכן בשרת Mongon
- אלגוריתם ההמלצה- שליפת השירים מהדטה בייס וחישוב cosine similarity.
- GUI

חלוקת העבודה:
החלק של הגישה למאגרי המידע החיצוניים ומערכת ההמלצה הייתה בתחום האחריות של אבי, בניית Databases, צד לקוח, צד שרת ובניית המונגו בתחום האחריות של שגיא.
העבודה נעשתה בתיאום מושלם וחלוקת משימות בנינו.

התרשים הבא מתאר את תהליכי המערכת ומסביר איך היא נבנתה. כמו כן ניתן לראות מהן הפעולות העיקריות במערכת ומה תפקיד כל שחקן המשתמש בה:



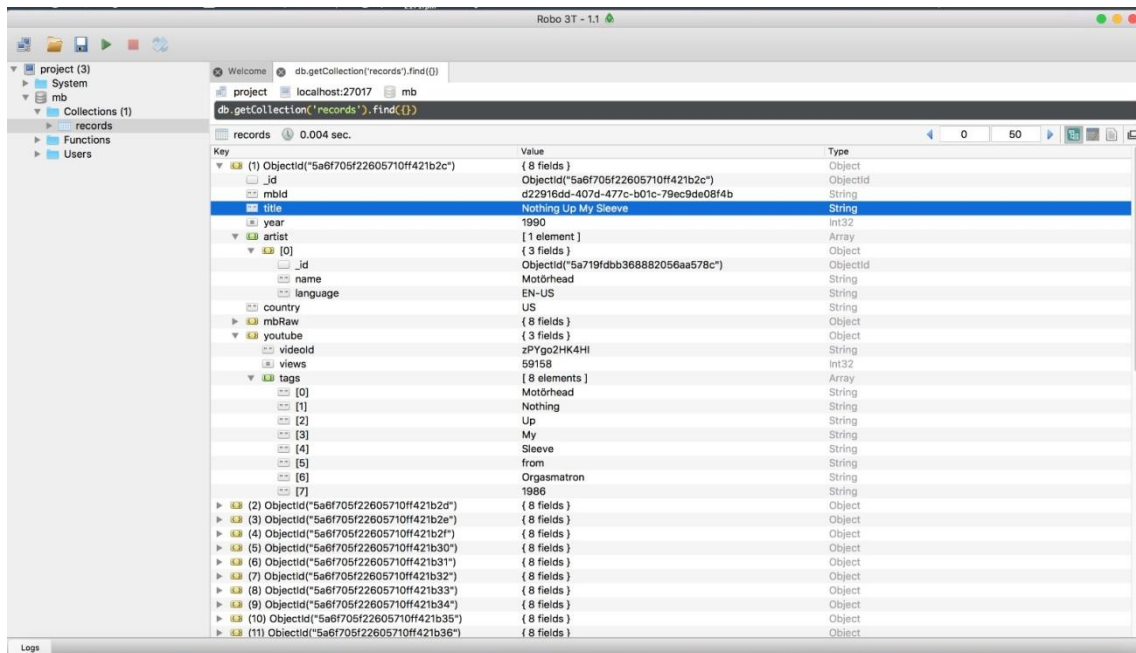
ראשית כל, אנו שואבים את המידע הנדרש ממאגרי מידע חיצוניים, ומאחסנים אותם ב-Databases מקומי. Databases המקומי מסודר כך ששליפת המידע תהיה יעילה ומהירה ובהתאם לשאלות המרכזיות של המערכת. לאחר מכן המשתמש יוכל להיכנס למערכת ולהזין את פרטיו. בכניסה של המשתמש ל-Databases, מתבצע clustering והוא ושאר המשתמשים מחולקים לקבוצות המאפיינות את שנת הלידה וארץ המוצא על פי פרטים אלה יקבל רשימת שירים מומלצת עבורו על פי אלגוריתם המלצה. רשימת השירים מוצגת עבורו בצירוף לינקים להשמעה. בלחיצה על לינק, נשלחת בקשת השמעה מ-YouTube. לאחר השמעת השיר, המשתמש יכול לקבוע אם אהב/הכיר את השיר או לא, וזאת על ידי דירוג השיר בסקלה של 1-5. לאחר דירוג השירים מתבצע חישוב התאמה בין המשתמשים. חישוב זה ישמש את אלגוריתם ההמלצה כאשר המשתמש יחזור לשימוש חוזר במערכת. האלגוריתם יציג למשתמש רשימת שירים שמורכבת משירים שהוא אהב, שירים שמשתמשים דומים לו (על פי החישוב) אהבו, ושירים נוספים שמתאימים לו על פי נתוני היבשים (גיל וארץ מוצא).

3.1 מבנה Databases של mongoDB.

את המידע שקיבלנו מ-YouTube ו-MusicBrainz, הכנסנו ל-Databases מקומי, המבוסס MongoDB. Databases מסודר בצורה כזאת שתתאים למבנה השאלות. מאחר והשאלות הראשיות של המערכת היא קבלת שירים, המדורגים גבוהה ביותר, על פי ארץ מוצא ושנת הוצאה, כך גם מסודר ה-collection (טבלה) ב-Databases. כל השירים נמצאים ב-collection אחד, המאופיינים באינדקסים של שנת הוצאה, ארץ מוצא, ודירוג. Indexing ב-mongo הוא כלי המאפשר לסדר את האובייקטים ב-Databases כך שהשליפה שלהם תהיה

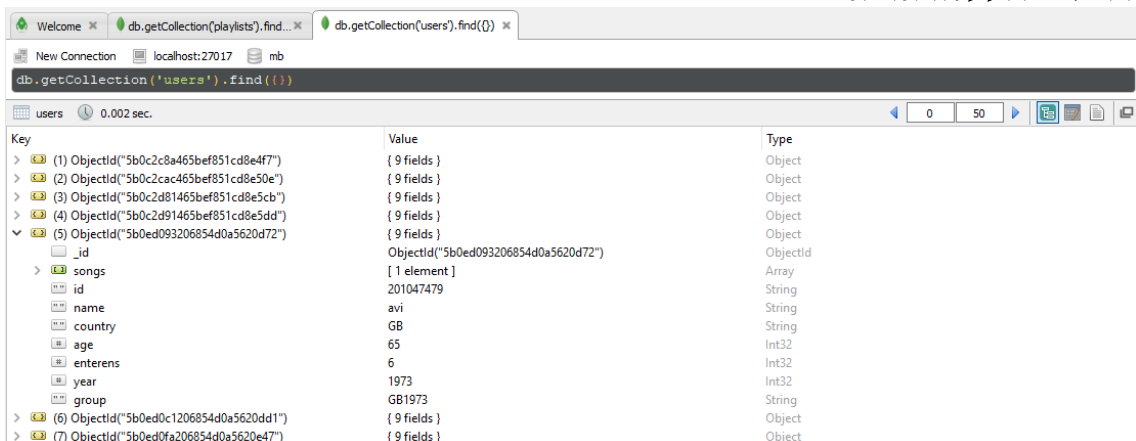
יעילה ומהירה. הוא מסדר אותם בסדר יורד (או עולה, תלוי בהגדרה) ובכך במקום לסרוק את כל הcollection לחפש תוצאות מתאימות, הוא ניגש בדיוק למקום בו הם נמצאים. כל שיר הינו document נפרד בתוך ה collection של השירים.

להלן צילום מסך ב3 Robo T [2] של הcollection המכיל את השירים ב-Database, ניתן לראות את השדות השונים, המכילים מידע על שיר ספציפי(בסדר יורד): ID שניתן לו ע"י מונגו, ID שניתן לו על ידי MusicBrainz, שם השיר, שנת הוצאה, אמן (שם, ID, שפה), ארץ מוצא, מידע מ-Youtube (מספר צפיות, לינק לצפייה, תגים).

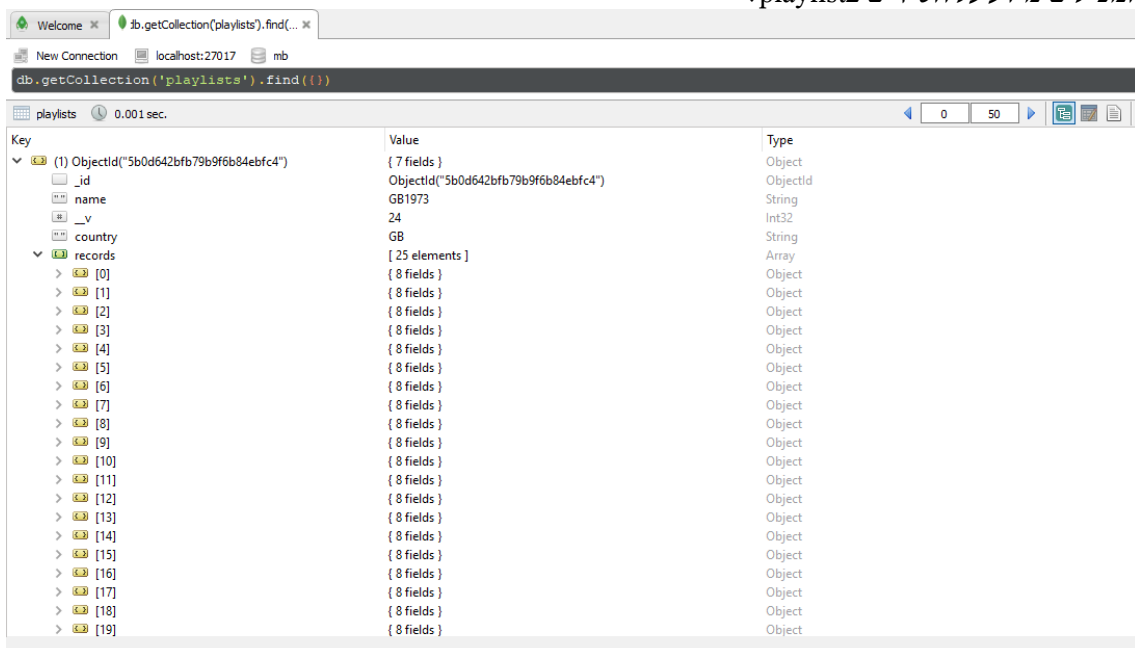


Collection נוסף נוצר עבור המשתמשים. עבור כל משתמש נוצר document נפרד, המכיל את הפרטים שאיתם נרשם למערכת. המערכת מחשבת עבור צרכיה את השנה בה המשתמש היה בגיל 20. המערכת בנוסף מבצעת clustering בין המשתמשים ומחלקת אותם לקבוצות על פי מאפייניהם. שם הקבוצה נשמר גם הוא בפרטי המשתמש.

להלן צילום מסך של הcollection המכיל את המשתמשים ב-Database, ניתן לראות את השדות השונים, המכילים מידע על המשתמש :

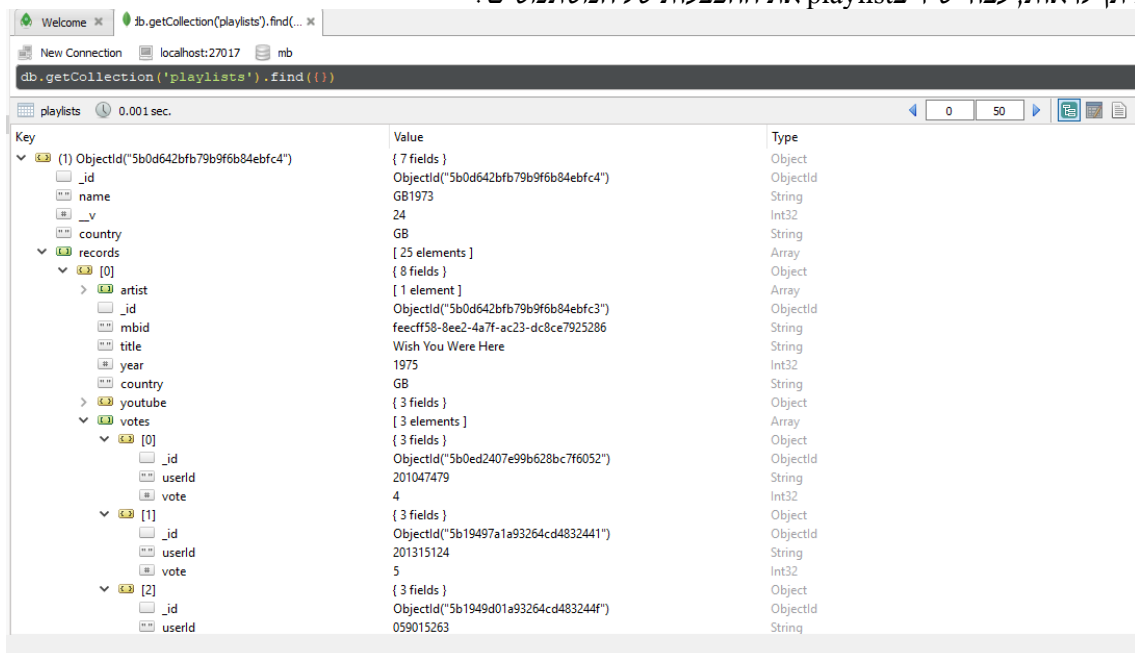


Collection נוסף נוצר עבור playlist. כל playlist הוא תוצאה של חיפוש בDatabase ושליפה של שירים עבור ארץ ומוצא ושנת הוצאה. עבור כל playlist נוצר document נפרד, המכיל את רשימת 25 השירים. כל שיר הוא אובייקט המכיל את המידע על השיר. כאשר משתמש צפה בשיר ודירג אותו, מידע זה נשמר כאן וכולל את ID של המשתמש והדירוג שלו. העובדה שהמידע הנ"ל נשמר במרוכז עבור על המשתמשים מאותה הקבוצה, מאפשרת לנו לחשב את ציון ההתאמה ביעילות. להלן צילום מסך של collection המכיל את playlist ב-Database, ניתן לראות את השדות השונים, המכילים מידע על השירים בplaylist:



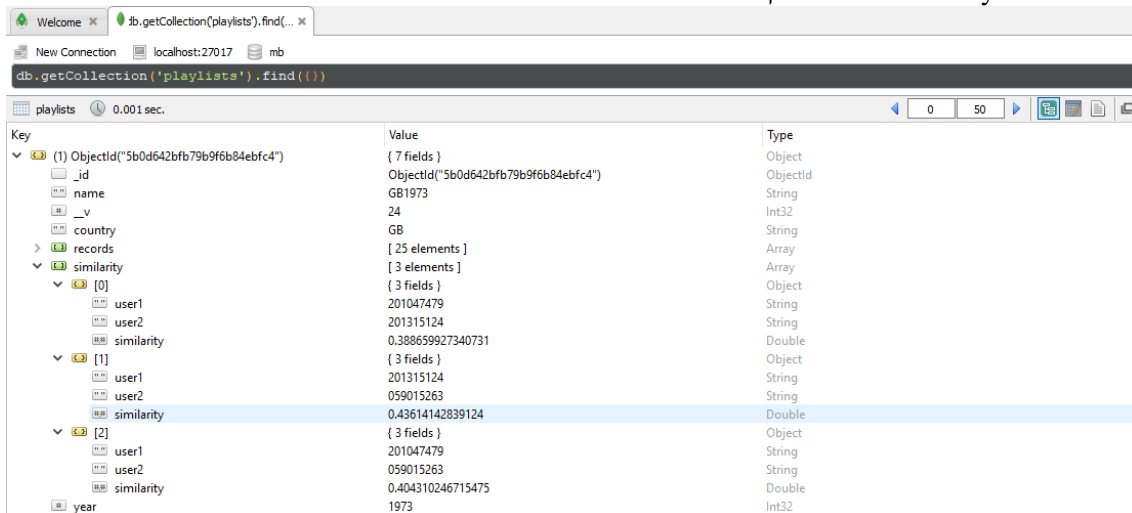
Key	Value	Type
(1) ObjectId("5b0d642bfb79b9f6b84ebfc4")	{ 7 fields }	Object
_id	ObjectId("5b0d642bfb79b9f6b84ebfc4")	ObjectId
name	GB1973	String
_v	24	Int32
country	GB	String
records	[25 elements]	Array
[0]	{ 8 fields }	Object
[1]	{ 8 fields }	Object
[2]	{ 8 fields }	Object
[3]	{ 8 fields }	Object
[4]	{ 8 fields }	Object
[5]	{ 8 fields }	Object
[6]	{ 8 fields }	Object
[7]	{ 8 fields }	Object
[8]	{ 8 fields }	Object
[9]	{ 8 fields }	Object
[10]	{ 8 fields }	Object
[11]	{ 8 fields }	Object
[12]	{ 8 fields }	Object
[13]	{ 8 fields }	Object
[14]	{ 8 fields }	Object
[15]	{ 8 fields }	Object
[16]	{ 8 fields }	Object
[17]	{ 8 fields }	Object
[18]	{ 8 fields }	Object
[19]	{ 8 fields }	Object

ניתן לראות, עבור שיר בplaylist את ההצבעות של המשתמשים:



Key	Value	Type
(1) ObjectId("5b0d642bfb79b9f6b84ebfc4")	{ 7 fields }	Object
_id	ObjectId("5b0d642bfb79b9f6b84ebfc4")	ObjectId
name	GB1973	String
_v	24	Int32
country	GB	String
records	[25 elements]	Array
[0]	{ 8 fields }	Object
artist	[1 element]	Array
_id	ObjectId("5b0d642bfb79b9f6b84ebfc3")	ObjectId
mbid	feecff58-8ee2-4a7f-ac23-dc8ce7925286	String
title	Wish You Were Here	String
year	1975	Int32
country	GB	String
youtube	[3 fields]	Object
votes	[3 elements]	Array
[0]	{ 3 fields }	Object
_id	ObjectId("5b0ed2407e99b628bc7f6052")	ObjectId
userId	201047479	String
vote	4	Int32
[1]	{ 3 fields }	Object
_id	ObjectId("5b19497a1a93264cd4832441")	ObjectId
userId	201315124	String
vote	5	Int32
[2]	{ 3 fields }	Object
_id	ObjectId("5b1949d01a93264cd483244f")	ObjectId
userId	059015263	String

ואת חישוב ה-cosine similarity בין המשתמשים :



Key	Value	Type
(1) ObjectId("5b0d642bfb79b9f6b84ebfc4")	{ 7 fields }	Object
_id	ObjectId("5b0d642bfb79b9f6b84ebfc4")	ObjectId
name	GB1973	String
_v	24	Int32
country	GB	String
records	[25 elements]	Array
similarity	[3 elements]	Array
similarity [0]	{ 3 fields }	Object
user1	201047479	String
user2	201315124	String
similarity	0.388659927340731	Double
similarity [1]	{ 3 fields }	Object
user1	201315124	String
user2	059015263	String
similarity	0.43614142839124	Double
similarity [2]	{ 3 fields }	Object
user1	201047479	String
user2	059015263	String
similarity	0.404310246715475	Double
year	1973	Int32

3.2 גישה למאגרים חיצוניים

על מנת לבסס את Databases המקומי של המערכת, ניגשנו למאגרים חיצוניים על מנת לקבל מהם מידע. לאחר מחקר מקיף, לראשונה ניגש לMusicBrainz לקבל שירים ומידע עליהם כגון שם השיר, שם אמן, שנה וארץ הוצאה. בקשת המידע התבצעה לפני שימוש המערכת, וזאת מאחר והMusicBrainz לא מתקשר לאחר 16 בקשות. ולכן ביצענו מראש הורדה גדולה של מידע. לאחר מכן, נשלח את המידע באמצעות YouTube, ממנו נבקש מידע כגון לינק לצפייה עבור כל שיר, מספר צפיות ותגים שמאפיינים את השיר.

הפניה לYouTube התבצעה על ידי ספריית YouTube Node [4]. שלחנו בקשת http request וקיבלנו key שבאמצעותו נוכל לתקשר עם YouTube API. אנו בונים את השאלות הרצויות ושולחים לשרת של YouTube, הוא מצידו מחזיר לנו את המידע המבוקש, כגון ID של שיר/וידאו. באמצעות ID הנתון, יכולנו לקבל מידע נוסף על שיר (כגון מספר צפיות ותגים) על ידי שאילתה שכתבנו בעצמנו ומכילה בקשה עבור כל המידע הנתון על הוידאו הספציפי, ותוכו בררנו את המידע הרצוי.

פניה לMusicBrainz התבצעה באופן דומה. הפניה של MusicBrainz API בבקשה לקבלת מידע, הייתה מוגבלת [7], מאחר והשרת עצמו מגביל אותנו ל16 פניות. את המידע קיבלנו בקבצי json, ואותם הכנסנו לDATABASE המקומי בסדר ובאופן הרצוי למערכת. MusicBrainz מבוסס RDF שהוא מודל החלפת נתונים ברשת המאפשר מיזוג נתונים גם אם מדובר בסכמות שונות.

3.3 אלגוריתם ההמלצה

אלגוריתם ההמלצה מבצע שליפה של תכני מדיה מתוך Databases על פי ארץ המוצא והשנה בה המשתמש היה בן 20 ובעלי דירוג פופולאריות גבוה ביותר. בנוסף, האלגוריתם מחשב דמיון בין משתמשים ולכן כאשר המשתמש יכנס שוב למערכת היא תמליץ עבורו שירים שהוא עשוי לאהוב על סמך דמיון למשתמשים דומים בנוסף לגיל וארץ המוצא.

דוח ולידציה על נכונות ההמלצה :

נבצע ניסוי המתאר את הפונקציונאליות של המערכת ונראה את נכונותה. במערכת רשומים 3 משתמשים, שלושתם נולדו באותה שנה ובאותה מדינה ולכן בעת ביצוע ה clustering המערכת שייכה אותם לאותה קבוצה (UK ובני 65). עבור שלושתם נשלפה רשימה 25 שירים המאפיינים מדינת המוצא ושנת הלידה.

להלן רשימת 25 השירים השייכים לplaylist המשותף של הקבוצה :

United Kingdom 1973		
	song	artist
1	Bohemian Rhapsody	Queen
2	Wish You Were Here	Pink Floyd
3	Smoke on the Water	Deep Purple
4	A Day In The Life	The Beatles
5	Alone Again (Naturally)	Gilbert O'Sullivan
6	Angie	The Rolling Stones
7	Roundabout	Yes
8	Can't You Hear Me Knocking	The Rolling Stones
9	Stand by Me	John Lennon
10	Rainy Days and Mondays	Carpenters
11	Suspicious Minds	Elvis Presley
12	Money	Pink Floyd
13	Lonely This Christmas	Mud
14	Wild Horses	The Rolling Stones
15	The Way We Were	Barbra Streisand
16	I'm Not in Love	10cc
17	Bang a Gong (Get It On)	T. Rex
18	Us and Them	Pink Floyd
19	Morning Has Broken	Cat Stevens
20	Have a Cigar	Pink Floyd
21	Brown Sugar	The Rolling Stones
22	It's Only Rock 'n' Roll	The Rolling Stones
23	Firth of Fifth	Genesis
24	24 Hours	Ace
25	Alive	Bee Gees

לכל משתמש מוצג 10 שירים רנדומאליים מתוך ה-playlist הוא מאזין להם ומדרג אותם על פי העדפה האישית שלו. על מנת לבדוק את נכונות המערכת, משתמש 1 דירג מס שירים בציון דומה למשתמש 2. הדירוגים של משתמש 3 שונים במעט מהדירוגים של משתמש 1 ו-2. ניתן לראות עבור כל משתמש את רשימת השירים שהוצגו לו ואת הדירוג :

United Kingdom 1973					
	song	artist	User1	User2	User3
1	Bohemian Rhapsody	Queen			
2	Wish You Were Here	Pink Floyd	5		
3	Smoke on the Water	Deep Purple			5
4	A Day In The Life	The Beatles		2	
5	Alone Again (Naturally)	Gilbert O'Sullivan		1	4
6	Angie	The Rolling Stones	5		
7	Roundabout	Yes	2	2	
8	Can't You Hear Me Knocking	The Rolling Stones			
9	Stand by Me	John Lennon			4
10	Rainy Days and Mondays	Carpenters	4	4	1
11	Suspicious Minds	Elvis Presley	1	1	
12	Money	Pink Floyd	4		
13	Lonely This Christmas	Mud		3	
14	Wild Horses	The Rolling Stones		3	1
15	The Way We Were	Barbra Streisand	2	2	
16	I'm Not in Love	10cc	4	4	
17	Bang a Gong (Get It On)	T. Rex			3
18	Us and Them	Pink Floyd	2		
19	Morning Has Broken	Cat Stevens		3	4
20	Have a Cigar	Pink Floyd			5
21	Brown Sugar	The Rolling Stones			
22	It's Only Rock 'n' Roll	The Rolling Stones	3		5
23	Firth of Fifth	Genesis			1
24	24 Hours	Ace			
25	Alive	Bee Gees			

לאחר שהמשתמשים מאזינים לשירים ומדרגים אותם, המערכת מחשבת את ציוני ההתאמה ביניהם, להלן ציוני ההתאמה בין המשתמשים:

users	cosine similarity
User1 & User2	0.427844419687375
User1 & User3	0.166130636485884
User2 & User3	0.137168197079925

ניתן לראות כי ציון ההתאמה בין משתמש 1 ומשתמש 2 גבוה יותר מאשר של כל אחד מהם עם משתמש 3. בפעם הבאה שמשתמש 1 (או משתמש 2) יכנס למערכת על מנת להאזין לשירים נוספים, המערכת תמליץ עבורו על שירים שמשתמש 2 דירג.

להלן רשימת השירים עבור משתמש 1 בפעם הבאה שיכנס למערכת :

user1-iteration2	
Wish You Were Here	top user1
Angie	top user1
Rainy Days and Mondays	top user1
Money	top user1
I'm Not in Love	top user2
Lonely This Christmas	top user2
Wild Horses	top user2
A Day In The Life	top user2
Bohemian Rhapsody	unseen
Smoke on the Water	unseen

בדיקה עבור שביעות הרצון מההמלצה תתבצע בהמשך פיתוח הפרויקט.

3.4 תיאור הכלים המשמשים לפתרון

בבניית המערכת השתמשנו בטכנולוגיות הבאות :

- צד הלקוח : css, html, javascript - צד הלקוח בנוי משלד של html ועליו עיצוב בעזרת css. מידע הגיל והארץ נשלח בעזרת javascript ו jquery לשרת וכך גם המידע המתקבל מצד השרת.
- צד השרת : צד השרת המבוסס node.js, ומשתמש בספריות כדוגמת- mongoose, youtube-node על מנת לשאוב מידע. צד השרת מתווך בין המידע שמאוחסן בשרתי mongo ובין צד הלקוח.
- גישה מהשרת לYouTube : על ידי ספריית YouTube Node, אנו שולחים בקשת http request ומקבלים key שבאמצעותו נוכל לתקשר עם YouTube API. אנו בונים את השאילתות הרצויות ושולחים לשרת של YouTube, הוא מצידו מחזיר לנו את המידע המבוקש.
- גישה מהשרת לMusicBrainz : אנו שולחים בקשת http GET לMB API בבקשה לקבלת מידע. הMB מבוסס RDF שהוא מודל החלפת נתונים ברשת.
- אחסון המידע- המידע על השירים מאוחסן בשרת mongodb בתוך collection אחד המאופיין באינקסים המאפשרים שליפה מהירה ויעילה של המידע. כל שיר מאוחסן ב document עם הפרטים הרלוונטיים למערכת. MLAB?!

4 תכנית בדיקות

בסיום כל שלב בפיתוח האפליקציה, יבוצעו מספר בדיקות תקינות שימולאו בהתאם בטבלה -

בדיקות מערכת

- ווידוא שהמערכת עונה לדרישות.
- בדיקות תקינות קלט מהמשתמש – האם המשתמש אכן יכול להכניס קלט לאפליקציה.
- בדיקות תקינות של בסיס הנתונים – היכולת לשלוף ולהכניס נתונים (C.R.U.D).
- בדיקת ממשק משתמש (GUI) עבור הפקדים והרכיבים במסכי האפליקציה(האם מגיבים ועובדים כנדרש).

מספר בדיקה	מודול נבדק	פעולה (תיאור הבדיקה)	תוצאה צפויה	תוצאה בפועל
1	App Launcher	הפעלת המערכת	המערכת עולה ומוצג מסך הבית שכולל הפניה ל2 עמודים : טופס מילוי נתונים והרשמה למערכת ומסך צפייה בשירים	✓
2	מסך ראשי	מעבר ל2 מסכים של מערכת בעזרת כפתורים	ניתן לעבור לשני המסכים בעזרת הכפתורים	✓
3	מסך הרשמה למערכת	הזנת פרטים אישיים : שם ות.ז.	ניתן להזין שם ות.ז. המאפיינת את המשתמש	✓
4	מסך הרשמה למערכת	בחירת שנה	ניתן להכניס גיל במספרים שלמים, גדולים מ20 וקטנים מ120	✓
5	מסך הרשמה למערכת	רשימת בחירה- מדינה	נפתחת רשימה של כל המדינות לבחירה	✓
6	מסך הרשמה למערכת	כפתור שליחת הנתונים, כפתור חזרה למסך הראשי	כפתורים עובדים כנדרש כאשר בוחרים בהם.	✓
7	מסך הרשמה למערכת	לחיצה על כפתור send	כפתור השליחה מאחסן את פרטי המשתמש במסד הנתונים נשלחים ומוצגת הודעה המציינת שהפעולה בוצעה בהצלחה	✓
8	מסך הרשמה למערכת	לחיצה על כפתור back	כפתור חזרה למסך הראשי מפנה חזרה למסך הראשי	✓
9	מסך הרשמה למערכת	קלט משתמש	המשתמש מצליח להכניס קלט לחיפוש	✓
10	מסך צפייה בשירים	הזנת ת.ז. לצפייה בשירים	ניתן להזין ת.ז.	✓
11	מסך צפייה בשירים	לחיצה על כפתור send	בלחיצה על הכפתור נשלפת רשימת שירים מתוך מסד הנתונים והיא מוצגת למשתמש	✓

בדיקות אינטגרציה

- בדיקה האם האפליקציה מתקשרת כנדרש עם השרת.
- בדיקה האם הנתונים שנשלפים מבסיס הנתונים הם נכונים ותיינים.
- האם השרת מתקשר כנדרש עם MusicBrainz
- האם השרת מתקשר כנדרש עם YouTube
- בדיקת תקשורת עם mongo
- אינטגרציה בין לקוח לשרת

מספר בדיקה	מודול נבדק	פעולה (תיאור הבדיקה)	תוצאה צפויה	תוצאה בפועל
1	התחברות לשרת	המערכת פונה לשרת node.js מקומי	המערכת מצליחה להתחבר לשרת	✓
2	שליפת נתונים	האפליקציה פונה לשרת בבקשה לקבלת מידע מבסיס הנתונים	המערכת מצליחה לקבל נתונים מבסיס הנתונים	✓
3	התחברות בין השרת ל-MusicBrainz	השרת פונה לmusicbrainz עם שאילתה לקבל מידע	המערכת מצליחה לתקשר עם musicbrainz	✓
4	התחברות בין השרת ל-YouTube	השרת פונה ל-YouTube עם שאילתה לקבל מידע	המערכת מצליחה לתקשר עם YouTube	✓
5	הוספה ועדכון נתונים מ-musicbrainz ל-youtube	השרת פונה לmusicbrainz ו-youtube על מנת לקבל נתונים ולהכניס או לעדכן נתונים אלה למסד הנתונים	השרת מצליח לעדכן ולהוסיף נתונים בבסיס הנתונים	✓
6	בדיקת תקשורת עם mongo	המערכת פונה לשירות mongo	נפתח שירות mongo המכיל database	✓
7	אינטגרציה בין לקוח לשרת	צד לקוח מבקש קבלת נתונים מהשרת	מקבל רשימת שירים או תגובה מתאימה	✓
8	אינטגרציה בין שרת ללקוח	צד שרת מחזיר ללקוח נתונים	מחזיר רשימת שירים או תגובה מתאימה	✓

בדיקות פונקציונאליות

בדיקות המוודאות שהמערכת מגיבה כנדרש לדרישות.

מספר בדיקה	מודול נבדק	פעולה (תיאור הבדיקה)	תוצאה צפויה	תוצאה בפועל
1	מסך ראשי	בדיקת תקינות לנתוני הקלט	הודעת שגיאה במקרה של נתונים לא תקינים (גיל קטן מ-20 או גדול מ-120)	✓
2	העברת נתונים לשרת	האם הנתונים עוברים לשרת ומתקבלים אצלו	הודעת שגיאה במקרה והנתונים לא הגיעו לשרת	✓
3	התקשרות בין השרת לבין db	בדיקה האם השרת ניגש לdatabase לקבלת נתונים	במקרה ויש נתונים, הם יחזרו לשרת, במקרה ואין השרת ייגש לmusicbrainz ולyoutube לקבל הנתונים	✓
4	התקשרות בין השרת לבין db	בדיקה האם השרת מחזיר נתונים מהdatabase על פי הקלט	המידע יעבור לתצוגה במסך הראשי	✓
5	שרת	בדיקה השירים מתאימים לשנה המתבקשת	השירים שהתקבלו אכן מתאימים לשנה המתבקשת בחיפוש בעזרת Index	✓
6	שרת	בדיקה האם השירים מתאימים למדינה המתבקשת	השירים שהתקבלו אכן מתאימים למדינה המתבקשת בחיפוש בעזרת Index	✓
7	מסך ראשי	בדיקה האם מוצג קישור מתאים לכל שיר	מוצג קישור מתאים להשמעה לכל שיר	✓
8	מסך ראשי	בדיקת תאימות בין כותרת לקישור שיר	הקישור תואם למידע המוצג בכותרת	✓

בדיקת תאימות

בדיקה שמטרתה לוודא שהמערכת מוצגת כרגיל בסוגי דפדפנים שונים בעלי רזולוציות שונות ובעלי גרסאות שונות.

מספר בדיקה	מודול נבדק	פעולה (תיאור הבדיקה)	תוצאה צפויה	תוצאה בפועל
1	סוגי דפדפנים שונים: chrom, explorer, safari, firefox	בדיקה האם המערכת מגיבה אותו דבר בדפדפנים שונים בעלי רזולוציות שונות וגרסאות שונות ומוצגת באופן זהה	המערכת מגיבה אותו דבר בכל דפדפן, הטופס ורשימת השירים מוצגת כנדרש והפונקציונאליות נשארת	✓

5 סקירת עבודות דומות בספרות והשוואה

Just-for-Me: An Adaptive Personalization System for Location-Aware Social Music Recommendation [10]

מאמרם של Zhiyong Cheng Jialie Shen משנת 2014 המפתח מערכת המלצה של שירים המשלב מיקום גיאוגרפי. במאמר מתוארת מערכת המלצת שירים, שממליצה על שירים על פי מדד פופולריות, תגים, הקשר מילולי ומיקום גיאוגרפי. במאמר מתואר כיצד נבנתה המערכת ומתוארים תוצאות המעידות שההמלצה מדויקת יותר ומשביעה רצון כאשר נלקחים הפרמטרים הנ"ל כאשר ממליצים שירים עבור משתמשים.

Follow the algorithm: An exploratory investigation of music on YouTube [11]

מאמרם של Massimo Airolidi, Davide Berald, Alessandro Gandini משנת 2016 החוקר את אלגוריתם ההמלצה של YouTube, המציע באופן אוטומטי רשימה של סרטונים קשורים למשתמש כתגובה לסרטון שצפה כעת. במהלך המחקר נבדקו רשת האסוציאציות בין 22,141 סרטוני מוזיקה של YouTube.

Music therapy in moderate and several dementia of alzheimer's type: a case-control study. [12]

מאמרם של H. B. Svansdottir, J. Snaedal משנת 2006 העוסק בקשר בין דצמנציה לבין טיפול באמצעות מוזיקה.

במהלך כתיבת המאמר, התבצע ניסוי על 38 אנשים הלוקים במחלת האלצהיימר והדמנציה ברמות שונות של חומרה, בשני בתי אבות, שבו השמיעו למטופלים מוזיקה.

תוך 6 שבועות ניתן היה להצביע על ירידה בהפרעה הנמדדת בסולם מחלת האלצהיימר וכמו כן ירידה בתוקפנות והחרדה.

מסקנות המאמר הם שקיים קשר ישיר, בטוח ויעיל לטיפול בחרדה ובחומרת מחלת האלצהיימר והדמנציה.

6 סיכום \ מסקנות

במסגרת פרויקט הגמר התבקשנו לבנות מערכת המלצת שירים עבור מיזם תמרינה. תמרינה הינה פלטפורמה היוצרת גירוי ושמע מותאמים אישית לחולי אלצהיימר ובכך משפרת את מצבם. על מנת לבנות מערכת המלצה נדרש מאיתנו להבין:

מה מטרת מערכת ההמלצה ומה המשתמש מצפה ממנה

מה Databases שעליו מתבססת מערכת המלצה

מהן השאלות שהמערכת תתבקש ליצר וכיצד לאחסן את המידע ב Databases
איך עובדת מערכת המלצה, מהן הסוגים השונים ומה מתאים יותר למערכת שלנו.

על מנת לענות על השאלה הראשונה, ביצענו פגישות עם מנחת הפרויקט ועם מיזם תמרינה על מנת להבין מה מטרת מערכת ההמלצה, אילו ההמלצה שהמשתמש מעוניין לקבל ומה מאפיין את המשתמש. הבנו כי המשתמש הינו קשיש הלוקה באלצהיימר והמידע שנוכל לקבל על העדפות המוזיקליות שלו הן מעטות ואף אפסיות. המידע שנמצא בידנו על אותו קשיש הוא לכל היותר שנת לידה וארץ מוצא. ההמלצות שהמשתמש מעוניין לקבל הן שירים שאפיינו את נעוריו וחיינו הבוגרים ונמצאים עמוק בתת המודע שלו. ובעצם כך הבאנו שמטרת המערכת היא לתת המלצות עבור שירים שהיו להיטים באותה תקופה.

לאחר שהבאנו את מטרת מערכת ההמלצה, נדרשנו להבין מה Database שיבסס אותה. ידענו כי המידע שיש לנו על המשתמש שישמש להמלצה הוא לא רב ומכיל בעיקר שנת לידה וארץ ומוצא, ולכן היה חשוב לנו למצוא מידע על מוזיקה שיכלול בין היתר שנת וארץ הוצאה. ביצענו מחקר מעמיק בפלטפורמות שונות המחזיקות מידע שיתופי על מוזיקה. במתואר בפרקים הקודמים, המחקר כלל מספר של פלטפורמות שונות כגון MusicBrainz, Last.fm, YouTube, Discogs. במהלך המחקר גלינו שקיים קושי רב להשיג את המידע והוא אינו מוכלל, מאורגן ומקוטלג כפי שנוח לחשוב בעידן שבו אנו נמצאים ובהתאם לדרישות המערכת. ראינו שעל פי דרישות המערכת, הפלטפורמות MusicBrainz ו-YouTube עונות על הדרישה ונעבוד מולם. שלפנו את המידע הרלוונטי מכל פלטפורמה באמצעות API שלה ואכנסו אותו ב Databases של המערכת בצורה יעילה כך שהיה לנו נוח לשלוף אותו.

כאשר עמדנו לפני בניית Databases של המערכת, חשבנו כיצד נארגן ונאחסן את המידע בו. חקרנו Database שונים ולאחר פגישה עם מומחה ב mongoDB בחרנו להשתמש בו. במחשבה כיצד לארגן את המידע

Databases עמדה מולנו השאילתה העיקרית שמאפיינת את הפרויקט, והיא: שלילת שירים מתוך המאגר על פי שנה הוצאה וארץ מוצא, עם דירוג פופולאריות גבוה ביותר. את המידע עבור השיר שכלל בין היתר את שר השיר, שם האמן, שנה וארץ הוצאת השיר, לקחנו מMusicBrainz. מידע הכולל הפופולאריות, תגים מאפיינים ולינק לצפייה לקחנו מYouTube. את כל המידע עבור אותו שיר אכסנו תחת אותו Document ואת כל השירים אכסנו בcollection כולל שנקרא Records. השתמשנו בindexing בmongoDB לפי שנת הוצאה וארץ מוצא, דבר המאפשר שליפה מהירה ויעילה מאחר והשירים מסודרים בסדר עולה (או יורד) על פי פרמטרים אלה.

קיימים שני מנגנונים עיקריים של מערכות המלצה. סינון שיתופי CF אשר מחשב דמיון בין משתמשים ומציע למשתמש המלצות על פי משתמשים שדמו לו, השני הוא מבוסס תוכן CB אשר מבוסס על התבוננות על תוכן המוזיקה. בחינה מעמיקה בשני המנגנונים, גרמה לנו לחשוב על היתרונות והחסרונות בין המנגנונים ומה יהיה יעיל יותר עבור המערכת שלנו. ידענו שהמערכת שהינה מיועדת לקשישים הלוקים באלצהיימר, המוגבלת במידע עבור העדפות המוזיקליות של המשתמש, ויהיה יותר נכון עבורה להשתמש במנגנון סינון שיתופי CF. ההחלטה להתבסס בפרויקט שלנו על מנגנון סינון שיתופי CF מאחר והבאנו שדמיון בין משתמשים יהיה יעיל יותר עבור המשתמשים בהיבט של המלצת התכנים.

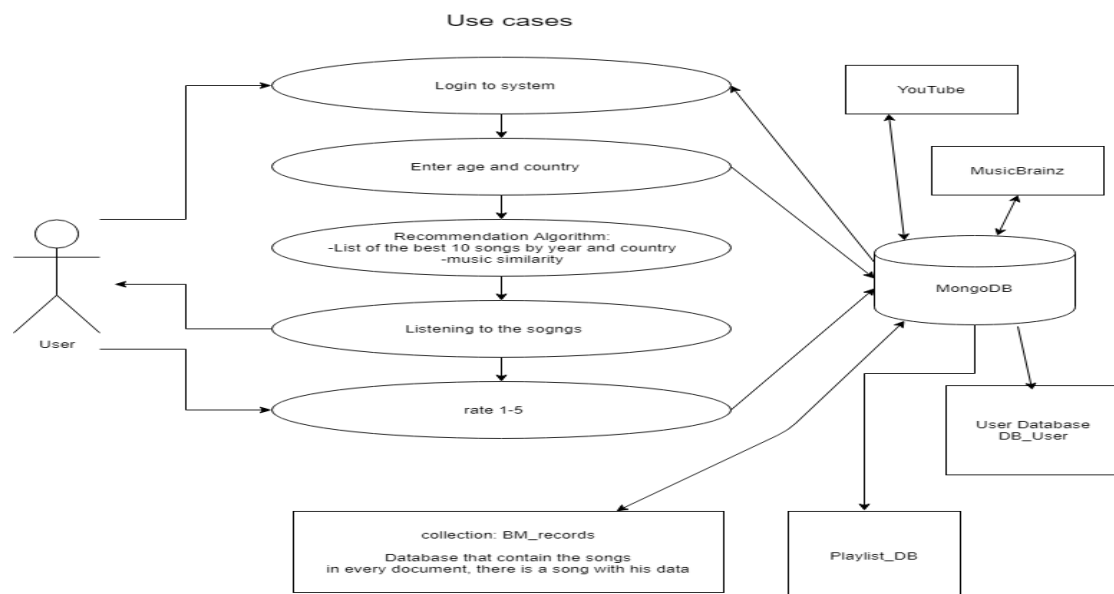
תחילת הפרויקט, אשר כלל מחקר מקיף וארוך בפלטפורמות שונות המחזיקות מידע על מוזיקה, גרם לנו להבין לראשונה כי למרות העידן בו אנו נמצאים, הקושי להשיג את המידע ולארגן אותו בDatabase היה מורכב וכלל אינטגרציה בין פלטפורמות שונות והיווה חלק גדול מהפרויקט שלנו. ראינו כי קיים קושי לחשב את מדד הפופולאריות של שיר בתקופה בה הוא השתחרר והיינו צריכים להסתמך על מדד פופולאריות עכשווי עבור שיר שהתנגן לפני 30 שנה. ברור כי הקושי נובע מטכנולוגיה שלא הייתה קיימת באותה תקופה. מסקנה נוספת שעלתה בזמן ביצוע הפרויקט הוא שיהיה עלינו להתגבר על אלמנט חשוב במערכות המלצה, שהוא המידע המערכת יודעת על המשתמש. מערכת המלצה קלאסית תאסוף מידע רב על המשתמש והעדפותיו ובכך תדע לתת המלצה על תכנים שמתאימים לו. בפרויקט נתקלנו בקושי שהמידע על המשתמש במערכת מוגבל והשתדלנו לתת המלצה טובה ביותר בהתבסס על נתונים אלה (גיל וארץ מוצא) ועליהם ביססנו את המערכת כולה.

7 נספחים

תרשימים נוספים, תכנון הפרויקט, טבלת ניהול סיכונים, טבלת דרישות (URD),

7.1 תרשימים וטבלאות

:Use cases



```
graph TD
    Start([Start]) --> MainScreen[main screen]
    MainScreen --> NewUser[New User: Enter age and country]
    MainScreen --> ExistUser[Exist User: Enter ID]
    
    NewUser -- "Not enter age or country" --> NewUser
    NewUser --> GetAge{Get age and country}
    GetAge --> ServerOp1[Server operation]
    ServerOp1 -- "Calculate year" --> ServerOp1
    ServerOp1 --> CreatePlaylist{-Create playlist with 25 best song by year and country}
    CreatePlaylist --> MongoDB[(MongoDB)]
    
    ExistUser -- "Login fail" --> ExistUser
    ExistUser -- "Login success" --> GetUser{Get User Details}
    GetUser --> ServerOp2[Server operation]
    ServerOp2 --> SetLink{-Set video link to YouTube  
-Random 10 song from Playlist}
    SetLink --> DisplayMusic[Display recommended music  
(video link to YouTube)]
    DisplayMusic -- "request to play video" --> PlayVideo[Play video]
    PlayVideo --> RateSong{Rate song}
    RateSong --> MongoDB
    RateSong --> YouTube[YouTube]
    
    MongoDB -- "Update DB with: video ID, views, tags" --> YouTube
    MusicBrainz[MusicBrainz] -- "Insert record: mbID, title, year, country, artist name" --> MongoDB
    YouTube --> MongoDB
```

פגישת הכרות עם המנחה האקדמי של הפרויקט- ד"ר מרים אללוף. במהלך פגישה זו שמענו על הפרויקט והמיזם וקבענו להיפגש עם מייסדות המיזם לפגישה.	21.7.2017
פגישת הכרות עם מייסדות מיזם טמרינגה, לאה וסטלה. במהלך פגישה זו הכרנו את לאה וסטלה ואת מיזם טמרינגה.	3.8.2017
פגישה עם המנחה ומייסדות המיזם. בפגישה זו קבענו על היעדים והדרישות מהפרויקט.	20.8.2017
פגישה עם המנחה לשם קביעת משימות. בפגישה זו המשכנו להגדיר משימות והחלה העבודה על הפרויקט.	18.9.2017
כתיבת מסמך ההצעה, כתיבת ניהול סיכונים, קריאה של מאמרים אקדמיים וכתיבת סקר שוק	19.11.2017
הגשת שלב ההצעה הכולל סקר שוק (איטרציה 0 ZFR)	19.11.2017
איטרציה MVP 1	01.12.2017
הצגת פונקציות לשליפת המידע הרלוונטי	20.12.2017
הגשת דוח אלפא (איטרציה 2)	01.02.2018

הגשת דוח בטא (איטרציה 3)	10.05.2018
הגשת הפרויקט (איטרציה 4 final)	14.06.2018

7.3 טבלת סיכונים

דרגת החומרה : 1 : לא חמור בכלל, 5 : חמור ביותר

#	הסיכון	חומרה	מענה אפשרי
1	אי התאמה של youtube api לצרכי הפרויקט	5	שימוש בפלטפורמה אחרת
2	אי התאמה של musicbrainz api לצרכי הפרויקט	5	שימוש בפלטפורמה אחרת
3	אי התאמה של last.fm api לצרכי הפרויקט	1	שימוש בyoutube
4	אי התאמה של discogs api לצרכי הפרויקט	1	שימוש בmusicbrainz
5	בחירת קטגוריות שאלון לא נכונה	3	שינוי שאילתות ומידע נשלף
6	אי אפשרות של ניגון youtube באתר נפרד	4	בדיקת זכויות יוצאים
7	שמירת מסד הנתונים על ידי שרת לא איכותי ולא מהיר	5	החלפת השימוש במונוגו לשרת מהיר יותר
8	ממשק לא ידידותי מספיק למשתמש	3	שינוי תצוגה והתאמה
9	אי מציאת מדיה להשמעה המתאימה לנתונים	5	אי הכנסה של השיר למסד הנתונים
10	אלגוריתם שליפה איטי	4	החלפה לאלגוריתם מהיר יותר
11	למידת הטכנולוגיות api ושלילת המידע	2	מחקר רציני בנושא
12	עדכון גרסה לאחד מסוגי api	1	התאמת השאילתות לגרסה החדשה
13	אינדקס שירים לא מעודכן בlast.fm	2	שימוש בyoutube
14	אי מציאת מאגר המתאים לנתוני המשתמש	4	ניסיון חיפוש לפי מדינה קרובה או גיל קרוב אחר
15	שינוי אופי שימוש התוכנה ומטרתה	1	התאמתו מחדש למטרתו החדשה
16	נפילה של מיוזיקברין, יוטיוב	5	שימוש בפלטפורמה אחרת
17	שרת node.js לא מצליח לעלות (פורט)	3	בדיקה שהחיבור אכן הצליח, אחרת נצא בהודעת שגיאה
18	קבלת הנתונים MusicBrainz ארוכה מדי	4	נחשוב לבצע את קבלת הנתונים מראש, או דרך אחרת יעילה יותר
19	שהלינקים מYouTube לא מתאימים למה שביקשנו	3	נבצע בדיקת תאימות בין הלינק לשמו על מנת לוודא. אחרת לא נציג את הלינק
20	קושי בהעלאת המערכת לשרת מארח חינומי	3	העלאת המערכת לשרת מארח בתשלום
21	שרת המכללה ייפול	2	מתחברים בremote או פונים למחלקת מחשוב להרים את השרת

7.4 רשימת טבלת דרישות

טבלת דרישות (User Requirement Document)

מס' דרישה	תיאור
1	מערכת קלה ונוחה לתפעול
2	המערכת תתבסס על ממשק web
3	המערכת תתבסס על שפת ה JavaScript
4	תשתמש ב node.js לצד שרת
5	תשתמש ב mongo להחזקת מסד נתונים
6	המערכת תאפשר הזנת נתונים של המשתמש
7	המערכת תלמד את גילו, ארץ מוצאו של המשתמש ובהתאם תציג רשימת השירים המתאימים עבורו
8	המערכת תציג את תוכן המדיה באופן ברור וקל להשמעה.
9	המערכת תנגן את תוכן המדיה מבלי לצאת לאתר אחר או תוכנה אחרת
10	המערכת תציג את תוכן המדיה בזמן קצר שלא יעלה על מספר שניות
11	במקרה ונתוני הלקוח לא נמצאים, המערכת תבקש מהלקוח להמתין לקבלת כל הנתונים ורק לאחר מכן תציג אותם
12	המערכת תאפשר למשתמש לסמן האם תוכן המדיה ששמע מתאים או לא מתאים (לייק או דיס-לייק) בשבילו

7.5 רשימת ספרות \ ביבליוגרפיה

[1] mongoDB הוא בסיס נתונים מקטגוריית NoSQL, נשען על מבנה מסמך מעל מימוש של BSON (Binary JSON file). <https://www.sciencedirect.com/science/article/pii/S1742287616300317>

[2] T3 Robo (לשעבר Robo mongo) הוא כלי ניהול של פלטפורמת MongoDB. <https://github.com/Studio3T/robomongo>

[3] ממשק API של YouTube מאפשר לשלב פונקציות המתבצעות בדרך כלל באתר YouTube במערכות חיצוניות (אפליקציות ותוכנות) <https://developers.google.com/youtube/v3/docs>

[4] מחשבון המחשב כמה עולה כל שאילת ל API YouTube. YouTube מגביל אותך במספר הבקשות שניתן לבצע בכל ששן בקשות וכך תדע כמה מידע תוכל לבצע בכל ששן כזה. https://developers.google.com/youtube/v3/determine_quota_cost

[5] מאמר העוסק בהגבלה של מספר הבקשות שניתן לבצע ל API של תוכנות של גוגל, הכוללות את YouTube בין היתר. <https://books.google.co.il/books?id=LUAlDwAAQBAJ&pg=PA46&lpg=PA46&dq=youtube+api+request+limit&source=bl&ots=zV6IJUGs-&sig=SIUoHTfgV33vtPWBJCXHn4bsg&hl=iw&sa=X&ved=2ahUKewilz9CugoXZAhUCVywKHZAJAYg4FBD0ATAHegQIBhAB#v=onepage&q=youtube%20api%20request%20limit&f=false>

[6] ממשק API למסד הנתונים של MusicBrainz. המיועד למפתחים של נגני מדיה ויישומים אחרים הדורשים מטה-דטה של מוסיקה. הארכיטקטורה של השירות עומדת בעקרונות העיצוב של REST.

התוכן מוגש בפורמט XML ו-JSON.

https://wiki.musicbrainz.org/Development/XML_Web_Service/Version_2

[7] MusicBrainz גם כן מגביל את כמות הבקשות שניתן לבצע אל השרת שלו. בכל סשן ניתן לבצע עד 16 בקשות, מה שלא הספיק. המטרה שלנו הייתה לקבל בכל פעם 150 אלף שירים, ובעקבות ההגבלה יכלנו לקבל 100 שירים בכל בקשה, סך הכל 1600, וזאת הסיבה שהיינו צריכים לבנות את Database של המערכת מראש. https://musicbrainz.org/doc/XML_Web_Service/Rate_Limiting

[8] Cosine similarity הינו מדד של דמיון בין שני וקטורים שונים מאפס ומוודד את גודל הזווית ביניהם. חישוב זה משמש למדידת דמיון בין משתמשים על סמך הדירוג שלהם במערכות המלצה שונות. בפרויקט נעשה שימוש בחישוב זה על מנת לחשב דמיון בין משתמשים שהצביעו עבור אותם שירים. פרק 9 Recommendation System בספר Mining of Massive Datasets מציג וממחיש את יעילות החישוב. <http://infolab.stanford.edu/~ullman/mmds/book.pdf>

[9] github מגביל את משקל הקבצים שניתן להעלות אליו עד 100 מגה, ולכן השתמשנו בdump כדי להעלות את Database שלנו <https://help.github.com/articles/working-with-large-files>

Just-for-Me: An Adaptive Personalization System for Location-Aware Social Music Recommendation [10]

מאמרם של Zhiyong Cheng Jialie Shen משנת 2014 המפתח מערכת המלצה של שירים המשלב מיקום גיאוגרפי. במאמר מתוארת מערכת המלצת שירים, שממליצה על שירים על פי מדד פופולריות, תגים, הקשר מילולי ומיקום גיאוגרפי. במאמר מתואר כיצד נבנתה המערכת ומתוארים תוצאות המעידות שההמלצה מדויקת יותר ומשביעה רצון כאשר נלקחים הפרמטרים הנ"ל כאשר ממליצים שירים עבור משתמשים. <https://dl.acm.org/citation.cfm?id=2578751>

Follow the algorithm: An exploratory investigation of music on YouTube [11]

מאמרם של Massimo Airolidi, Davide Berald, Alessandro Gandini משנת 2016 החוקר את אלגוריתם ההמלצה של YouTube, המציע באופן אוטומטי רשימה של סרטונים קשורים למשתמש כתגובה לסרטון שצפה כעת. במהלך המחקר נבדקו רשת האסוציאציות בין 22,141 סרטוני מוזיקה של YouTube. <https://www.sciencedirect.com/science/article/pii/S0304422X16300973>

Music therapy in moderate and severe dementia of alzheimer's type: a case-control study. [12] מאמרם של H. B. Svansdottir, J. Snaedal משנת 2006 העוסק בקשר בין דצמנציה לבין טיפול באמצעות מוזיקה.

במהלך כתיבת המאמר, התבצע ניסוי על 38 אנשים הלוקים במחלת האלצהיימר והדמנציה ברמות שונות של חומרה, בשני בתי אבות, שבו השמיעו למטופלים מוזיקה.

תוך 6 שבועות ניתן היה להצביע על ירידה בהפרעה הנמדדת בסולם מחלת האלצהיימר וכמו כן ירידה בתוקפנות והחרדה.

מסקנות המאמר הם שקיים קשר ישיר, בטוח ויעיל לטיפול בחרדה ובחומרת מחלת האלצהיימר והדמנציה. <https://www.cambridge.org/core/journals/international-psychogeriatrics/article/music-therapy-in-moderate-and-severe-dementia-of-alzheimers-type-a-casecontrol-study/D51CCDFF17656F27C927D1BA322BA74C>

Abstract

The project was created in collaboration with the Tameringa project. Tameringa is a platform that creates personalized stimulation and hearing for Alzheimer's patients in thus to improves their condition.

During the project, we designed and built a song recommendation system based on criteria and similarity between users for Tamringa.

We expanded the pool of songs by adding media content by year and origin country, from external repositories such as YouTube, and stored the information in a dedicated database that was built for the recommendation system.

The recommendation system is based on shared music databases on the network and will show the user the appropriate songs and play links within the system without exiting to an external site. The system will calculate a match between users based on playback characteristics and preferences and will recommend it for future songs.

In this article, we will present the system of recommendation that was implemented, planed and the conclusions we received after many cases that were on various subjects.

Software Engineering Department

Final Project – 2018

MUSIC SIMILARITY

By

Avi Cohen

Sagi Marciano

Academic Supervisor:

Dr. Miriam Allalouf

Software Engineering Department

MUSIC SIMILARITY

By

Avi Cohen

Sagi Marciano

June 2018