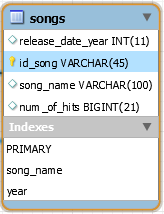
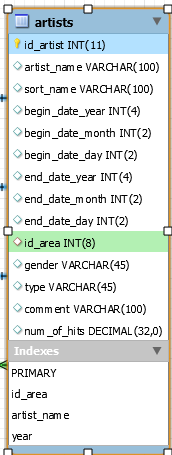
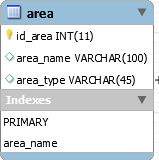
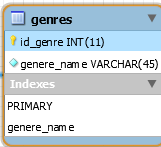
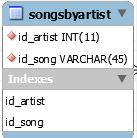
# מסד הנתונים

## תיאור הסכמה

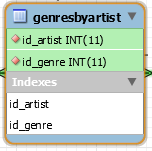
## הטבלאות

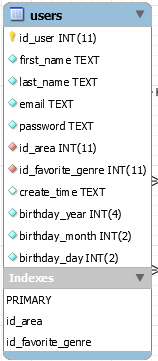


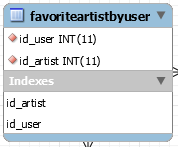
* **songs- טבלת השירים:**
  + עמודות-
    - Release\_date\_year- שנת שחרור השיר.
    - Id\_song- מפתח.
    - Song\_name- שם השיר.
    - Num\_of\_hits- מספר ההשמעות של השיר.
  + אינדקסים-
    - PRIMARY- המפתח
    - Song\_name
    - Year
* **Artists- טבלת האמנים:**
  + עמודות-
    - id\_artist- מפתח.
    - artist\_name- שם האמן.
    - sort\_name
    - begin\_date\_year- תאריך התחלה- שנה. עבור אדם- יום הולדת, עבור הרכב או להקה- שנת הקמה.
    - begin\_date\_month- תאריך התחלה- חודש.
    - begin\_date\_day- תאריך התחלה- יום בחודש.
    - end\_date\_year- תאריך סיום- שנה. עבור אדם- פטירה, עבור הרכב או להקה- פירוק או פרידה.
    - end\_date\_month- תאריך סיום- חודש.
    - end\_date\_day- תאריך סיום- יום בחודש.
    - id\_area- מקום. לידה עבור אדם, פעילות עבור הרכב או להקה. מפתח זר.
    - Gender- מין.
    - Type- סוג- אדם, להקה, הרכב וכד'.
    - Comment- הערות.
    - num\_of\_hits- מספר ההשמעות של האמן.
  + מפתחות זרים-
    - Id\_area- מקשר לטבלת area.
  + אינדקסים-
    - PRIMARY- המפתח.
    - Id\_area
    - Artist\_name
    - Year
* **Area- טבלת המקומות:**
  + עמודות-
    - Id\_area- מפתח.
    - Area\_name- שם המקום.
    - Area\_type- סוג המקום. למשל- עיר, מדינה, מחוז, אי.
  + אינדקסים-
    - PRIMARY- מפתח.
    - Area\_name
* **Genres- טבלת הז'אנרים:**
  + עמודות-
    - Id\_genre- מפתח.
    - genere\_name- שם הז'אנר.
  + אינדקסים-
    - PRIMARY- מפתח.
    - Genere\_name



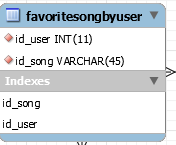
* **Songsbyartist- טבלת קישור בין שירים לבין אמנים (כי לשיר אחד יכולים להיות הרבה אמנים, ולהפך- לאמן אחד יכולים להיות הרבה שירים):**
  + עמודות-
    - Id\_artist- האמן.
    - Id\_song- השיר.
  + מפתחות זרים-
    - Id\_artist- מקשר לטבלת artists.
    - Id\_song- מקשר לטבלת songs.
  + אינדקסים-
    - Id\_artist
    - Id\_song



* **Genresbyartist- טבלת קישור בין אמנים לבין ז'אנרים (כי אמן אחד יכול להיות מתויג לכמה ז'אנרים, וז'אנר אחד יכול להיות מתויג לכמה אמנים):**
  + עמודות-
    - Id\_artist- האמן.
    - Id\_genre- הז'אנר.
  + מפתחות זרים-
    - Id\_artist- מקשר לטבלת artists.
    - Id\_genre- מקשר לטבלת songs.
  + אינדקסים-
    - Id\_artist
    - Id\_genre
* **Users- טבלת משתמשים, לצורך יצירת עובדות שרלוונטיות למשתמש:**
  + עמודות-
    - id\_user- מפתח.
    - first\_name- שם פרטי.
    - last\_name- שם משפחה.
    - Email- כתובת אימייל.
    - Password- סיסמה.
    - id\_area- מקום מגורים.
    - id\_favorite\_genre- הז'אנר האהוב (החלטנו אסטרטגית לאפשר רק ז'אנר אהוב אחד).
    - create\_time- תאריך יצירת המשתמש.
    - birthday\_year- שנת לידה.
    - birthday\_month- חודש לידה.
    - birthday\_day- יום בחודש.
  + מפתחות זרים-
    - Id\_area- מקשר לטבלת area.
    - id\_favorite\_genre- מקשר לטבלת genres.
  + אינדקסים-
    - PRIMARY- מפתח.
    - Id\_area
    - Id\_favourite\_genre



* **Favoriteartistbyuser- רשימת אמנים מועדפים על משתמשים, כי משתמש אחד יכול לאהוב הרבה אמנים, ואמן אחד יכול להיות אהוב על הרבה משתמשים:**
  + עמודות-
    - Id\_user- משתמש.
    - Id\_artist- אמן אהוב.
  + מפתחות זרים-
    - Id\_user – מקשר לטבלת users.
    - Id\_artist- מקשר לטבלת artists.
  + אינדקסים-
    - Id\_user
    - Id\_artist



* **Favoritesongbyuser- רש ימת שירים מועדפים על משתמשים, כי משתמש אחד יכול לאהוב הרבה שירים, ושיר אחד יכול להיות אהוב על הרבה משתמשים:**
  + עמודות-
    - Id\_user- משתמש.
    - Id\_song- שיר אהוב.
  + מפתחות זרים-
    - Id\_user- מקשר לטבלת users.
    - Id\_song- מקשר לטבלת songs.
  + אינדקסים-
    - Id\_user
    - Id\_song

## תהליך הבנייה

### נתונים גולמיים

< מאיפה לקחנו את הנתונים ואיך נרמלנו אותם>

נרמול כולל-

ביצוע joins

Drop לתווים מיוחדים

רק המידע הרלוונטי

הוספת אינדקסים לצורך מהירות חיפוש

### יצירת טבלת משתמשים

<איך ג'נרטנו משתמשים עם כל הנתונים שלהם>

# תיאור המחלקות

* **DataBaseLayer- שכבה זו אחראית ללוגיקה ולניהול של ה-DB מול האפליקציה.**
  + **Banks**- "בנקים" עבור שאילתות, יוריסטיקות[[1]](#footnote-1) ורשומות למיניהן.
    - HeuristicsBank- בנק עבור יוריסטיקות.
    - IKnowWhatIWantQuriesBank- בנק עבור שאילתות לכפתור IKnowWhatIWant (וכן עבור השלמת מילים).
    - RandomRecordsBank- בנק עבור רשומות אקראיות.
    - UserQueryBank- בנק עבור שאילתות שקשורות למשתמשים.
  + **Entities**- ישויות שמייצגות רשומות ב-DB. יש לציין שלא כל המידע נשמר בישויות אלו, אלא רק המידע הנדרש עבור הרצת השאילתות והצגת התוצאות.
    - Artist- אמן.
    - Genre- ז'אנר.
    - Place- מקום.
    - Song- שיר.
    - User- משתמש.
    - EntitiesFactory- "מפעל" לייצור ישויות ע"פ פרמטרים, ה"מפעל" מייצר את הישויות באמצעות שאילתות ל-DB.
  + **Executers**- מחלקות שמריצות שאילתות ע"פ פרמטרים שונים, ומחזירות string המייצגת את תוצאת השאילתה.
    - ExecuterFactory- מחלקה המייצרת Executer אקראי (לצורך הכפתור I'm feeling lucky).
    - GenreExecuter- Executer שמריץ שאילתה עבור כפתור ה-Hit me with => genre.
    - IExecuter- ממשק המייצג Executer עבור כפתור I'm feeling lucky.
    - LuckyExecuter- מחלקה המייצרת Executer אקראי באמצעות ה-ExecuterFactory ומריצה אותו. אם ה- Executer לא הצליח למצוא תוצאה מתאימה, היא מייצרת Executer אקראי חדש ומריצה אותו, וכך הלאה. עבור כפתור I'm feeling lucky.
    - NumberExecuter- Executer שמריץ שאילתה עבור כפתור ה-Hit me with => number.
    - PlaceExecuter- Executer שמריץ שאילתה עבור כפתור ה-Hit me with => place.
    - SignInExecuter- Executer המריץ את השאילתות הנדרשות עבור sign in (התחברות).
    - SignUpExecuter.cs- Executer המריץ את השאילתות הנדרשות עבור sign up (הרשמה).
    - SimpleArtistExecuter- Executer שמריץ שאילתה עבור כפתור ה-I know what I want => artist.
    - SimplePlaceExecuter- Executer שמריץ שאילתה עבור כפתור ה-I know what I want => place.
    - SimpleSongExecuter- Executer שמריץ שאילתה עבור כפתור ה-I know what I want => song.
    - YearExecuter- Executer שמריץ שאילתה עבור כפתור ה-Hit me with => year.
  + **מחלקות נוספות:**
    - **DataBaseConnector**- המחלקה שמדברת עם ה- DB ישירות. יוצרת את ה-connection, ומריצה בפועל את השאילתות.
    - Heuristics- יוריסטיקה כוללת את השאילתה עצמה, וכן את הדרך להצגת התוצאה.
    - **SelfExecuterHeuristics**- כמו Heuristics, אך בנוסף יודעת להריץ את השאילתה.
    - **HeuristicsFactory**- מחלקה המייצרת Heuristics ו-**SelfExecuterHeuristics** ע"פ פרמטרים המסופקים לה.
* **Controllers- שכבה זו היא שכבת ביניים בין ה-DataBaseLayer ל-Project (שכבת ה-GUI), כלומר מקשרת בין הלוגיקה של האפליקציה להצגה.**
  + CompletionController- מחלקה אבסטרקטית עבור Controller-ים שזקוקים ל-feature של השלמת מילים.
  + FeelingLuckyController- controller עבור הכפתור I'm feeling lucky.
  + HitMeWithController- controller עבור הכפתור hit me with.
  + IKnowWhatIWantController- controller עבור הכפתור I know what I want.
  + SignInController- controller עבור הכפתור sign in (התחברות).
  + SignUpController- controller עבור הכפתור sign up (הרשמה).
* **Project- שכבת ה-GUI.**
  + <להשלים>

*הערה: תיעוד מלא של הקוד ניתן למצוא בקוד המקור.*

## נקודות להתייחסות

* המחלקות SignInController ו-DataBaseConnector עוצבו כסינגלטונים, כיוון שמחלקות רבות משתמשות בהן ומסתמכות עליהן.
* כפי שניתן לראות, ישנה חלוקה ברורה בין שכבת הלוגיקה, שכבת הגישה ל-DB ושכבת ה-GUI.

# תיאור השאילתות

## שאילתות להרשמה (sign up)

אלו הן שאילתות update, שאנו מבצעים באמצעות טרנזקציה על מנת לוודא שהמידע המוכנס ל-DB קונסיסטנטי.

### מסך הרשמה ראשוני

במסך ההרשמה הראשוני על המשתמש להקליד: שם מלא (string), כתובת דוא"ל (string), סיסמה (string), יום הולדת (int\*int\*int- עבור יום בחודש, חודש ושנה), ז'אנר אהוב (string) ומקום מגורים (string).

ראשית אנו נדרשים לוודא שהז'אנר והמקום שהוקלד חוקיים- כלומר, קיימים ב-DB.

לצורך כך אנו משתמשים בשאילתות הבאות (אותן ניתן למצוא במחלקה EntitiesFactory):

* *select id\_area from area where lower(area\_name) = <שם המקום שהוקלד>*
* *select id\_genre from genres where lower(genere\_name) = <שם הז'אנר שהוקלד>*

אם חוזרת תשובה משתי השאילתות הנ"ל, אזי המקום והז'אנר קיימים וניתן להמשיך הלאה.

לצורך הוספת המשתמש ל-DB אנו משתמשים בשאילתה הבאה (אותה ניתן למצוא במחלקה SignUpExecuter):

* *INSERT INTO users (first\_name, last\_name, email, password, id\_area, id\_favorite\_genre, birthday\_year, birthday\_month, birthday\_day) VALUES(@firstName, @lastName, @email, @password, @idArea, @idGenre, @year, @month, @day)*

לאחר ביצוע השאילתה הזו בהצלחה, עוברים למסכי הוספת שירים ואמנים מועדפים.

### מסכי הוספת שירים ואמנים מועדפים

במסכים אלו על המשתמש להקליד שמות של שירים ואמנים מועדפים (string).

ראשית אנו נדרשים לוודא שהשירים והאמנים שהוקלדו חוקיים- כלומר, קיימים ב-DB.

לצורך כך אנו משתמשים בשאילתות הבאות (אותן ניתן למצוא במחלקה EntitiesFactory):

* *select \* from artists where lower(artist\_name) = <שם האמן שהוקלד> limit 1*
* *select id\_song from songs where lower(song\_name) = <שם השיר שהוקלד>*

אם לא חוזרת תשובה מהשאילתה, האמן/ השיר לא מתווספים לרשימת האמנים/ השירים המועדפים ע"י המשתמש.

אחרת- מוסיפים את השיר/ האמן לרשימת השירים/ האמנים המועדפים ע"י המשתמש, באמצעות השימוש בשאילתות הבאות (אותן ניתן למצוא במחלקה SignUpExecuter):

* *INSERT ignore INTO favoritesongbyuser (id\_user, id\_song) VALUES (@idUser, @idSong)*
* *INSERT ignore INTO favoriteartistbyuser (id\_user, id\_artist) VALUES (@idUser, @idArtist)*

## שאילתות להתחברות (sign in)

## שאילתות להשלמת מילים

## שאילתות לקבלת רשומות אקראיות

## שאילתות ליצירת עובדה (קבלת תוצאות והצגתן)

1. יוריסטיקות הן הבסיס עבור יצירת העובדות. יוריסטיקה כוללת את השאילתה עצמה, וכן את הדרך להצגת התוצאה. [↑](#footnote-ref-1)