# dKomp

Sanaa Fidahussain

SWDV691

# Table of Contents

## Database Document Design

I will be using MongoDB for my app: dKomp, so it will be a Document Database Design. The reason for using MongoDB is it's easily compatible with the Ionic framework as well as its JSON structure. For this application, the database needs to be flexible enough to retrieve data, and MondoDB has been recommended as a good option for mobile databases. MongoDB has flexible data modeling, which would be perfect for this application in case more needs to be added. Document databases are also known for their fast write performance as well as their fast query performance, both of which are useful for this application in particular as we are only writing to the database with data provided by the Spotify API and retrieving it, both of which MongoDB excels at.

## Table Design

There will be only one table and that will consist of two columns: Spotify username, and an array of keywords associated with that account. This will exist so the Word Cloud package has a way of instantly retrieving the list of data it will use to generate. That is so far the only use for a database in the app, as it will not require a user registration. However, it will require verifying their Spotify credentials. Once their Spotify credentials have been verified, the service will write a list of the keywords generated through various API calls to the database. Once that is finished, the service will use that array to create the word cloud using the word cloud npm package, and then pass control back to the UI. There will be no other relationships, since there is only one table.

## Example

An example of a row inserted into the database will be like the following:

```
{ username: "username", keywords: [ "Taylor Swift", "pop", "Akon", "Bollywood", "Foreign", "Instrumentals", "Love Story", "Romance", "We Don't Talk Anymore", "Selena Gomez", "Justin Beiber", "K-POP", "BTS", "Exo", "DNA" ] }
```

The username will be of type String and modeled as such: `{ "username" : 1 }, {unique:true}`

The keywords will be of type array filled with String modeled as such: `{ "keywords" : [] }`