Team 6: Jinghu Lei, Lainey Hall, Jason Mayfield

Cache

For user authentication, we can use OAuth and browser cookies to let a user "stay" logged in without user interaction after the first time. For our database, we should also cache the user's custom preferences that are generated with our application on each instance as well as their own Spotify affinity that we can cache.

- For user preferences:

  We would cache the different weightings of each genre/category and save them after each song to the database. When the application is exited and starts up again, we can just load the users past preferences using their profile id. This would make it a seamless experience across each time they open our application and across different browsers; their preferences would be accessible anywhere.

- For user affinity:

  Spotify automatically maps a user's top artists and genres and provides them through API. This allows us to initially use this information to cache song and artist recommendations for the user, and thus we can start caching those songIDs and artistIDs before they even start playing a song. This would make their loading experience much faster between songs. Also different users can also use the stored cache to get song's cached in response to another user. Of course, this can cause massive expansion on the database, so we would set a cache limit of these.

Data Models

```
const UserSchema = new Schema({
    name: String, // user name, required
    profile : String, // user id from Spotify, required
    preferences : [PreferenceSchema] // preference array based on category
});
const PreferenceSchema = new Schema({
    name : String, // name of the category
    weight : double, // weight of the genre for the user, required: -1 <= x <= 1, -1 bad, 1 good
    categoryid : String // categoryID, required – maps to the genre cache
});
const CategoryCacheSchema = new Schema({
    categoryid : String, // categoryID, mapped from PreferenceSchema, required
    songs : [SongCacheSchema] // array of cached songs from the genre
});
const SongCacheSchema = new Schema({
    name : String, // name of the song
    songid : String, // songID from Spotify, required
    artistid : String // artistID
});
```

Correspondingly to the Mongo object schema, we would have two JSON documents. The first being a User JSON with embedded preferences. The second being our category cache, with songs embedded in those.

User JSON:

```
{
   “name” : “John Doe”,
   “profile” : “”,
   “preferences” :
         [
             {
               “name” : “Top List”,
               “categoryid” : “toplists”
             },
             {
               “name” : “Country”
               “weight” : 1,
               “categoryid” : “country”
             },
             {
               “name” : “Mood”
               “weight” : 0.4,
               “categoryid” : “mood”
             }
         ]
}
```

Category Cache JSON:

```
{
   “categoryid” : “toplists”,
   “songs” :
         [
             {
               “name” : “Happy Now”,
               “songid” : “4keoy2fqgwGnbWlm3ZVZFa”,
               “artistid” : “2qxJFvFYMEDqd7ui6kSAcq” // zedd
             },
             {
               “name” : “I Love It (& Lil Pump)”,
               “songid” : “4S8d14HvHb70ImctNgVzQQ”,
               “artistid” : “5K4W6rqBFWDnAN6FQUkS6x” // kanye
             }
         ]
}
```

Sequence Diagram – Also Attached