Prescribe Architectural Design

The focal point of the Prescribe software is the transformation of an artist name that the user types into a search bar into a dynamically generated table on the web page containing a list of Artist recommendations based on their input. This is accomplished through a series of calls and relays to the Spotify API. Due to the data-transforming nature of the problem, the architecture of the software is best realized using the Data-Flow architecture model.

The Data-Flow process is initiated when the user hits the Enter key or the search button on the Prescribe home page. When the user submits their data, it is passed on to a JavaScript function which formats the string into a format where it can be used to search the API. The function then makes an Ajax (Asynchronous JavaScript And XML) GET call to the Spotify API which in turn will return a JSON formatted list of Artists that closely match the string the user entered. Each artist object that is returned by the API at this stage has three important pieces of information: Artist Name, Popularity, and ArtistID. First, Prescribe considers the Artist Names of the Artists returned in the search. If there is an Artist whose name is an exact match with the string the user inputted (ignoring case) then that artists ArtistID function is passed on to the next process. Should no artist names exactly match the user’s search criteria, then the Popularity field is given priority. The ArtistID of the most popular artist returned from the API will be passed to the next process (or the first artist whose popularity matches the maximum Popularity value of all Artists, should two have the same popularity).

The next process in the data-flow consists of taking an ArtistID (a unique identifier for a musician or band), formatting an appropriate URL, and then querying the API for a list of Artists recommended to the Artist with that ArtistID. There are not a lot of complications at this portion of the data-flow.

Next, the list of related Artists retrieved from the API in the previous step must be processed. There is a table defined in the HTML of the webpage with no elements. The related data about the related Artists is passed to a JavaScript function which dynamically builds the aforementioned table with data from the related artists JSON data. This table will include columns for artist picture, artist name, and a button which if pushed will create a modal pop-up displaying the biography of the artist in question.

If the user is registered and logged in, there is another step in the data-flow process that is available to them. They may choose to save a particular artist from the recommended artist table that interests them. Doing this will entail sending a POST-method call to the User-Artist database to save an entry with the user’s identifier, the artist’s ArtistID, and the date. This saved data can be retrieved later at the user’s convenience in a similar format to the original recommendation table.