## Music Manager Exercise for Full Stack Javascript training

1. **Technologies**

Type: **Single Page Application**

Server side technology: **Node.js, Express.js, Async.js, Q.js, …**

Database: **MongoDB**

Client side: **AngularJS, Angular Material Design Kit (**[**https://github.com/angular/material-start**](https://github.com/angular/material-start)**), …**

Unit test and coverage report: **Mocha for Server side, Karma for client side.**

1. **Requirements**

Functional Requirements:

* Nothing special here, looking at the mockup should be self-explanatory
* But the pagination, sorting will be done by the server side, client side only need to send appropriate requests

Unit Test Requirements:

* **Test must be written at the same time with code, no exception.**
* **Focus on unit test first (mock all the dependencies such as database, connection, stream, … and focus on logical tests), functional tests / acceptance tests will be added later**

Misc. Requirements:

* Try to follow REST APIs design guideline as much as possible: <https://developer.atlassian.com/docs/atlassian-platform-common-components/rest-api-development/atlassian-rest-api-design-guidelines-version-1>
* Follow Angular / Google Material Design guideline as much as possible: <http://www.google.com/design/spec/material-design/introduction.html>

1. **Advanced Features (to be implemented later):**

Authentication:

* Allow to authenticate with username & password.

Authorization:

* Users have full permissions on their own songs / playlists.
* User with Admin role can add/edit/delete any songs / playlists

Acceptance Test:

* Use protractor to write acceptance test