**Music Manager**

1. The goal of the exercise is to make a music manager application.

A music manager application allows you to **browse** through a music library, **add** new entries and **delete** them (so the usual Create, Update and Delete stuffs (CRUD)).

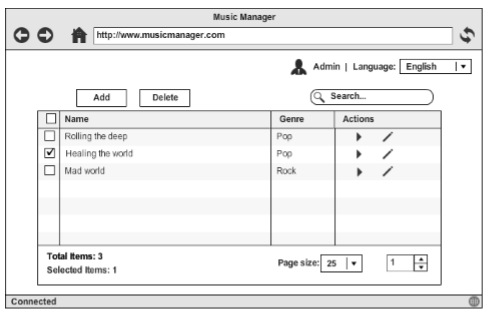
2. All music data should be stored server side in a database.

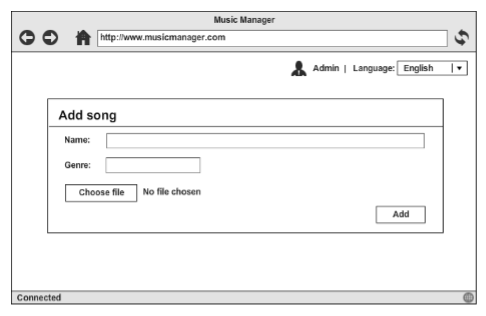
The client should be a **web** client. Communication between clients and server should go over **REST**.

On the server side we expect the application to be deployed in a Tomcat server. The client should always communicate to one dedicated service interface (MusicManagerService).

The MusicManagerService implementation is a **POJO** which is using Spring to interact with the database.

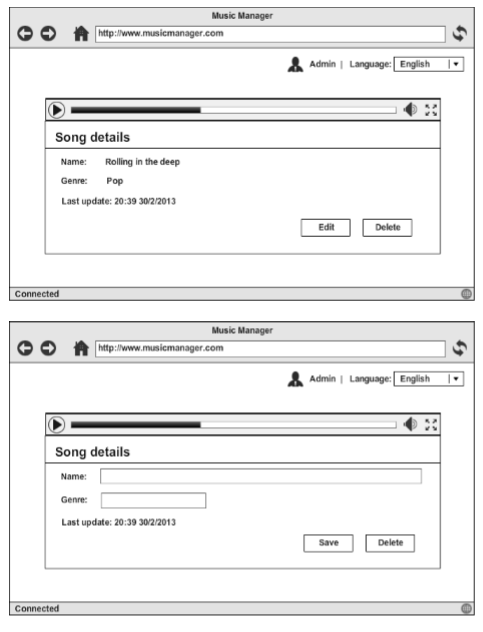
**How should the GUI look?**



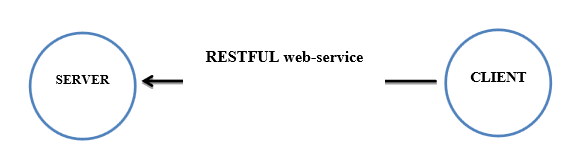


The GUI can be very basic: a table which **lists** all music files. There should be an action bar with 'Add’ (which leads to ‘Add Song’ page mentioned below), ‘**Delete’** buttons and a Search box.

We also need to support ‘**Play’** and ‘**Edit’** modes for each song, as followings:



**Technology requirements**



Technologies need to be used:

* Spring/Spring boot
* MyBatis (Transaction, Mapping, Paging)
* Spring MVC
* MySQL & Tomcat

Nice to have:

* Spring Security (Login)
* Caching (redis, memcached)

Technologies need to be used:

* Javascript
* Twitter Bootstrap
* HTML5/CSS3
* JSON

Nice to have:

* Unit-testing (Junit, Mockito)
* Localization (L10n) and Internalization (i18n)