***Music Store – part 2***

In part 2 of the exercise, we use *scaffolding* to generate the *CRUD* functionalities of the *Admin* menu. We will also program the user part of the store.

***1. Scaffolding***

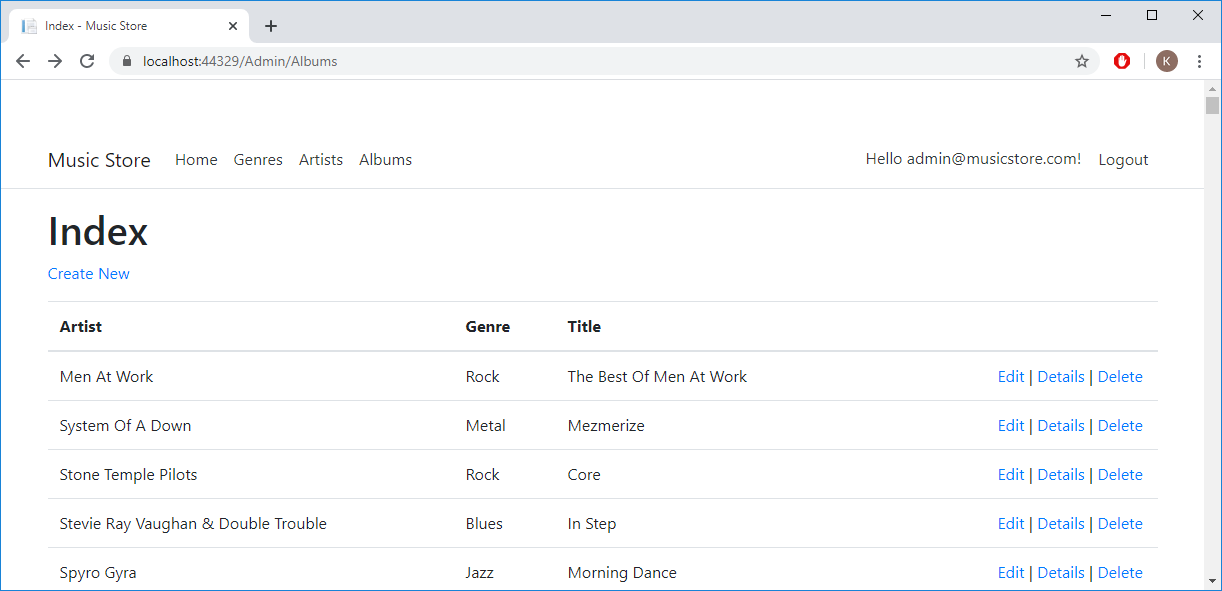
First delete *AlbumsController.cs* and the view that belongs to it (from part 1 in the *Admin*-Area).

Now use scaffolding to generate the *CRUD* functionalities for *genres*, *albums* and *artists*. Place the generated code in the *Administrator Area* and make sure that the pages can only be used by the *Administrator Role*.

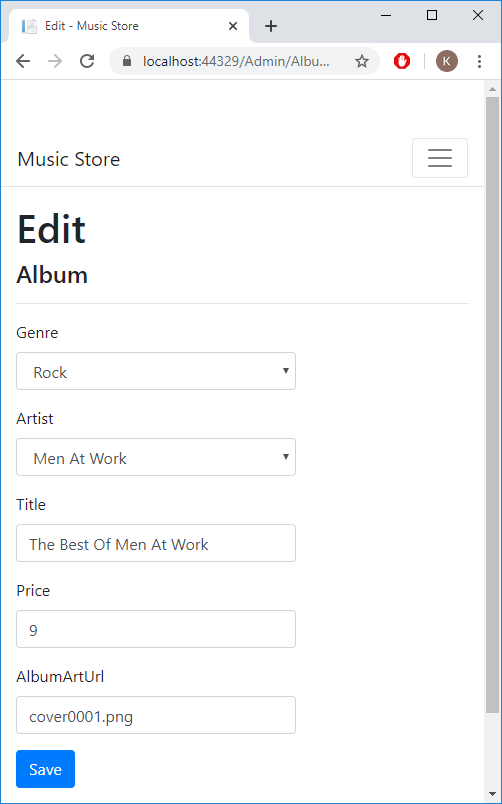
Then test the menu items in the *Admin* menu of your application.

***2. Additional modifications***

On the *Albums* overview page, show the genre and artist of the album in addition to the title:



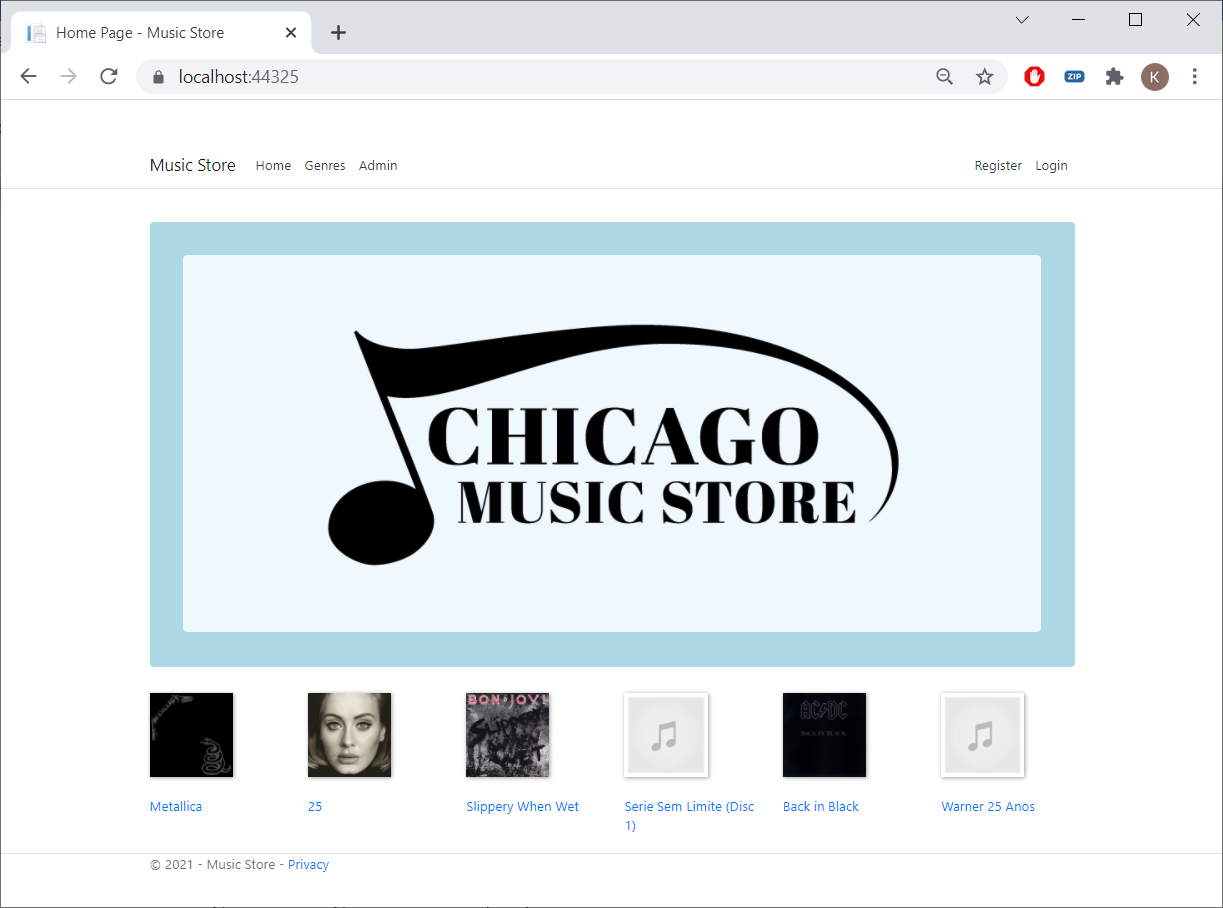
Make sure you can use nice drop-down lists for artist and genre when editing and creating an album:



The administrator part of our application is finished for the time being.

***3. Homepage***

Customize the homepage of the web shop and show six random albums under the logo:



You can read six albums at random as follows:

\_context.Albums.OrderBy(a => Guid.NewGuid()).Take(6);

You don't have to code the layout for the six albums yourself. You will find a start in the resources. You just need to complete it.

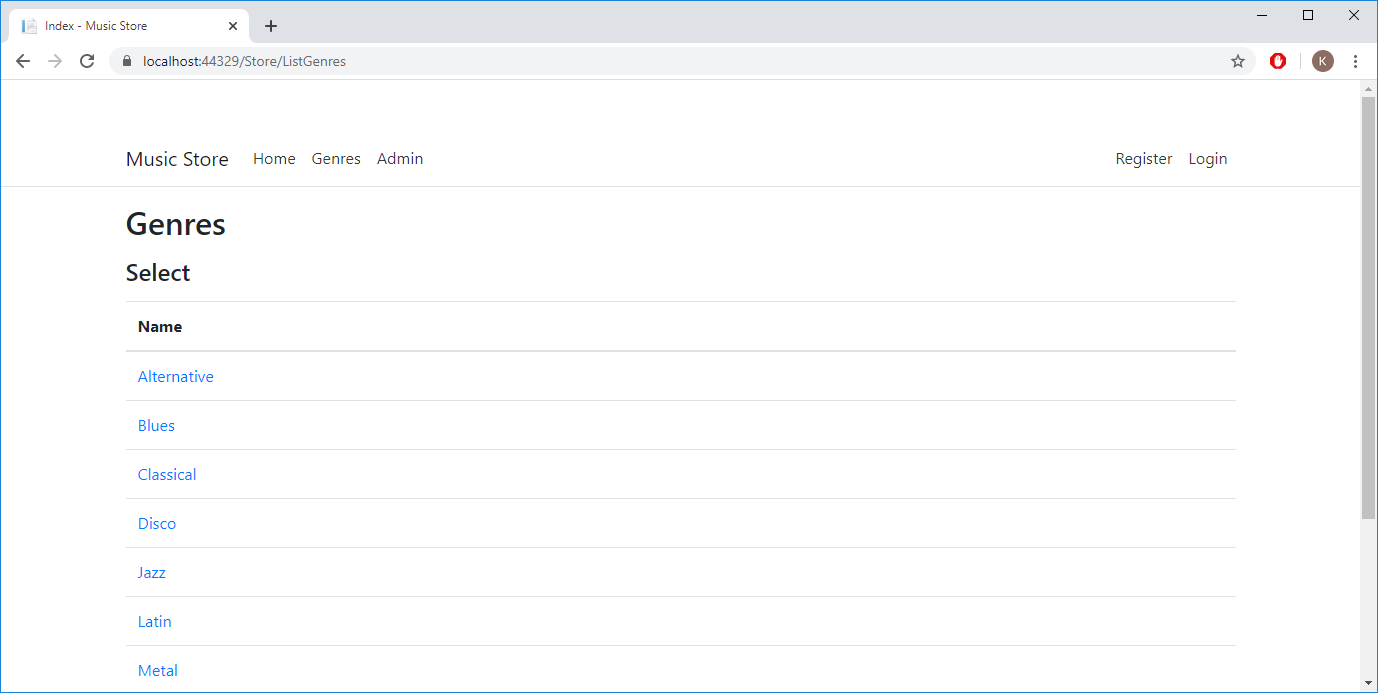
Clicking on an image or title will jump to the *Details* method of the *Store* Controller. The album's *id* should be given as a parameter (you'll have to do this yourself).

***4. ListGenres***

Create an additional empty controller *StoreController.cs*.

Hint: For the remainder of the exercise, it may be useful to copy bits of program code from the generated classes.

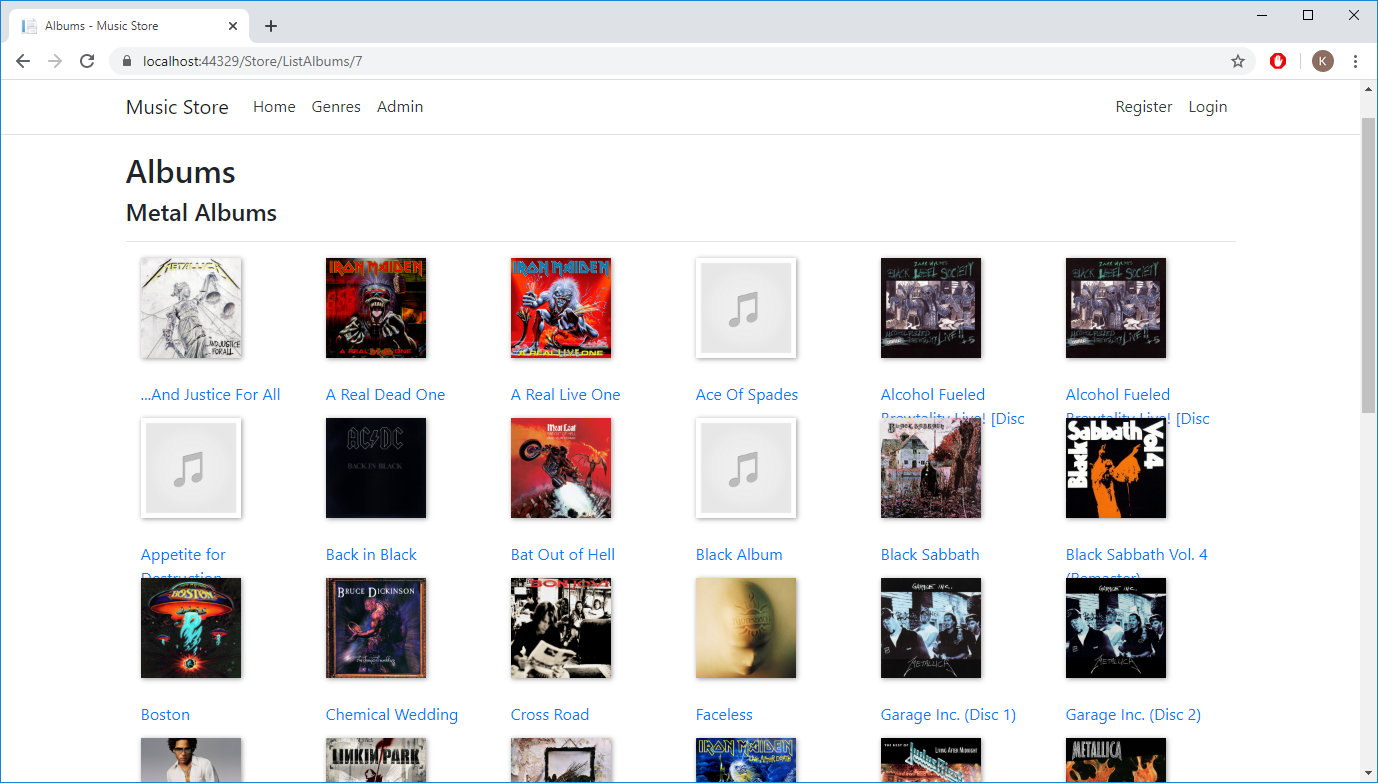
Give the *controller* the usual *constructor* for controllers. Create a *ListGenres* method to retrieve all genres (alphabetically). Also code the associated view *ListGenres.cshtml*. If you click on *Genres* in the menu, you get:



If you click on a genre, jump to the *ListAlbums* method of the *Store* Controller. The *id* of the genre is the parameter.

***5. ListAlbums***

In the *Store* Controller, create a *ListAlbums* method. The method has the following parameter: the *id* of a genre. Display all albums of that genre alphabetically. For *Metal* you get:



If you click on an image or title, you jump to the *Details* method of the *Store* Controller. The *id* of the album is given as a parameter.

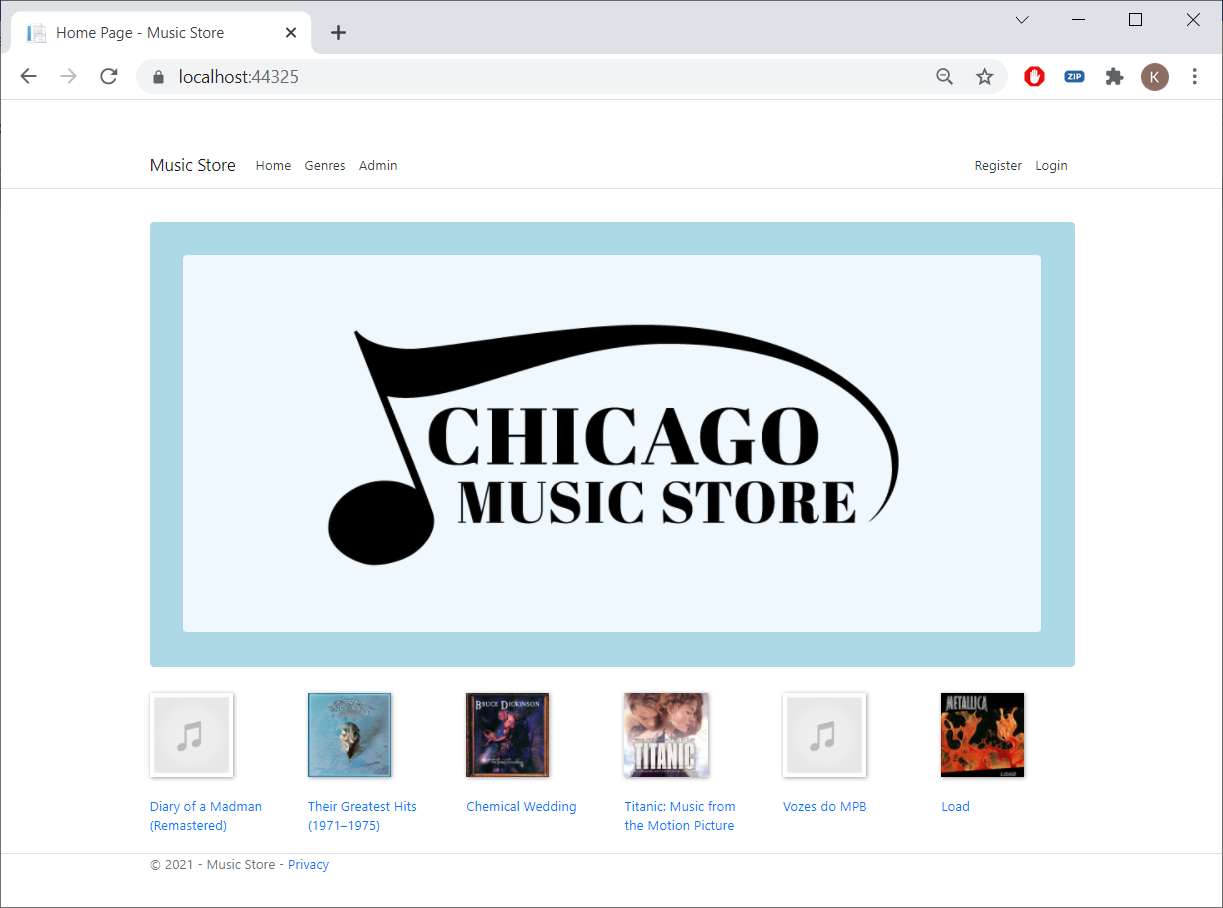
***6. Details***

Create a method *Details* in the *Store* Controller. The method has the following parameter: the *id* of an album. Show the details of the album. Try it out:



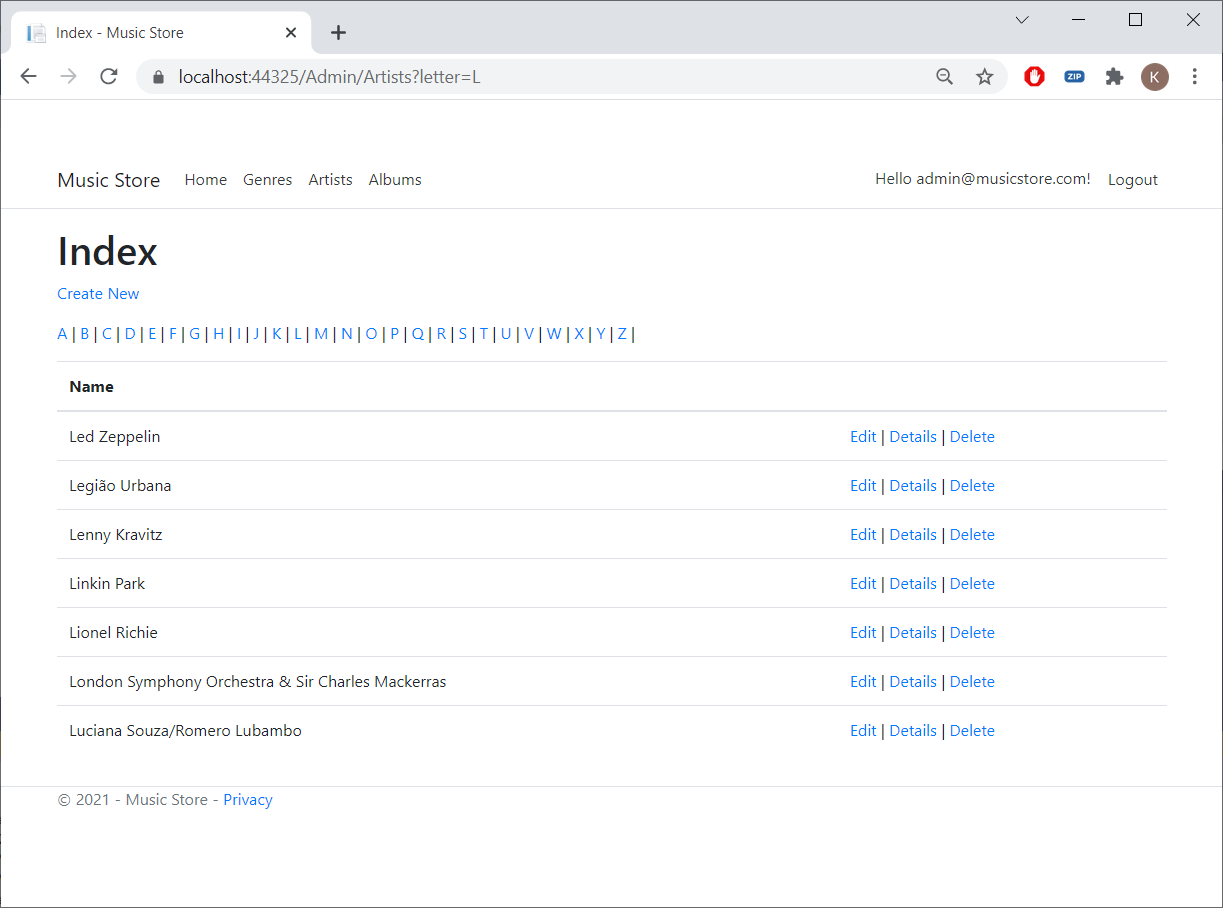
The *Add to cart* hyperlink refers to the *AddToCart* method of the *ShoppingCart* controller. The *id* of the album is the parameter.

Please test all hyperlinks of the homepage:



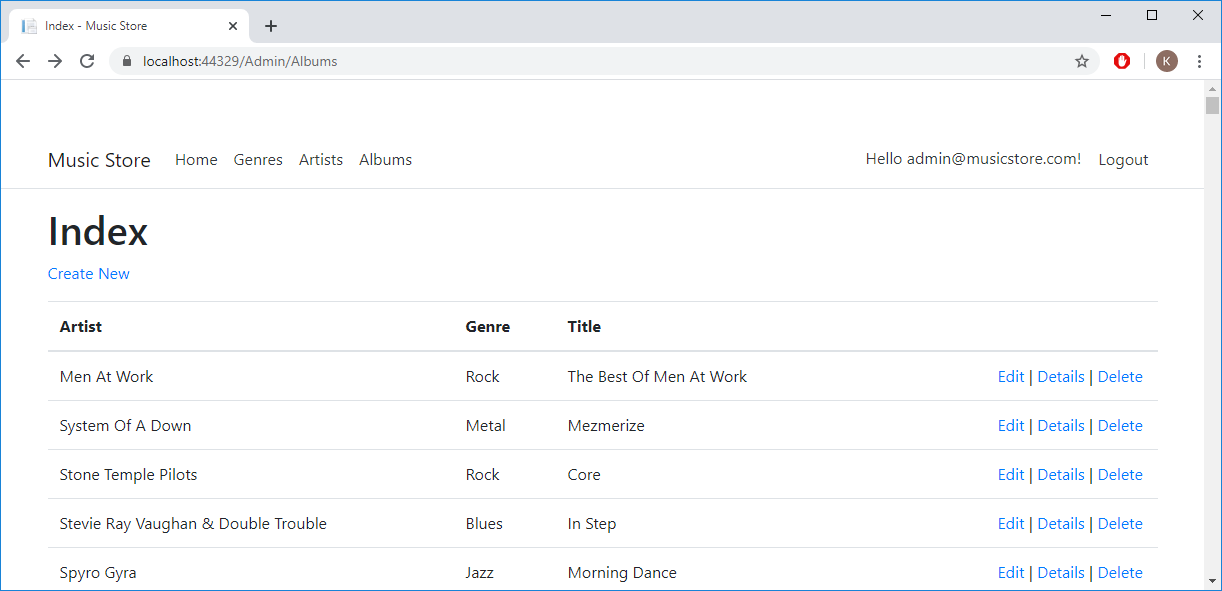
***7. Artists alphabetically***

In the *Artists* overview (in the *Admin* area), show all the letters of the alphabet at the top. Display the artists in alphabetical order and finally make sure that you can filter the artists by clicking on a letter. If you click on the *L* for example, you will only see the artists whose names start with an *L*:



***8. Modifying Albums***

In the admin page to manage albums, you currently get a list of all albums, which is not very user-friendly:



At the top of this page, put a select list with all the artists and a second select list with the genres. Make sure you can filter the albums using these two select lists. Display the albums in alphabetical order.

Put another text box at the top so you can also select albums using (part of) the album title:

