Specifications of Acted Features

Artist browser

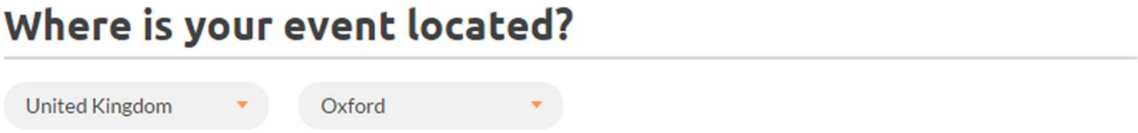
The main purpose of this feature is to generate dynamically the Acted artist browser page and to implement the search of artist based on few criteria to allow Acted customers find the right artist for their event.

The page URL will be:
http://www.acted.co/search

Search page

Here are the dynamic elements of the search page to be generated:

1. Location

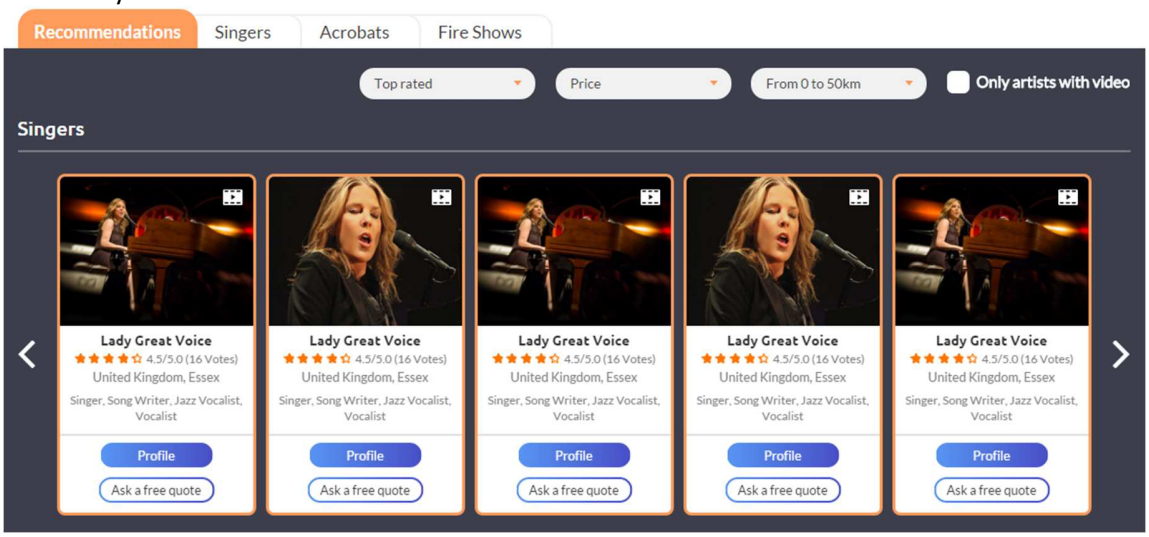


First dropdown list should be filled with the list of countries, “United Kingdom” should be selected.
Second dropdown list should be filled with the list of the related regions (new table/entity to create) of the selected countries (first drop-down list).

2. Categories

The main titles (working as tabs) should be filled with the list of the “category” tables.
Below the tab, the list of sub-categories of the selected category should appear.

3. By default results



The first tab "Recommendations" should be the only tab appearing when the page http://www.acted.co/search is loaded without search criteria/variables.
The dropdown lists appearing under a tab will allow the user to order or to filter the displayed results, without reloading the page. The ordering and filtering criteria will be the following hard coded in the HTML page:

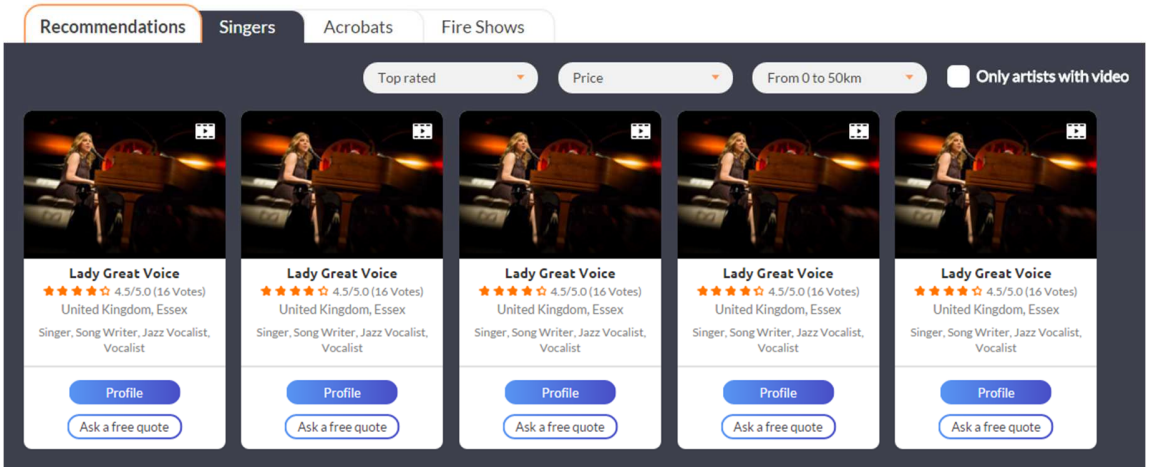
- ★ Rating:
  - Top rated
  - Lowest rated
- ★ Price:
  - Cheapest
  - More expensive
- ★ Distance:
  - From 0 to 50 km
  - From 50 to 200 km
  - From 200 km to 1000 km

Moreover, when the user clicks on the "Only artists with video" checkbox, the list of artist blocks should be filter to display only the performances attached to at least one video.

The "Recommendations" tab results should be composed by blocks of each "recommended" (boolean attribute to be added to the "category" entity) sub-category. Under each recommended sub-category, the list of "recommended" artists should appear in a row of artist performance blocks.
Each performance block should be generated from the database, using the performance first picture or video, the performance title, the artist rating average (in star icons, rate and number of ratings), the artist country and city, the artist sub-categories (max 4).
User will be able to scroll the recommended artists using the left and right arrow of each sub-category row.

When the user clicks on the performance block image or on the “Profile” button, the user should be redirected to the artist profile page, using the http://www.acted.co/profile/[ARTIST_NAME] URL.
When the user clicks on the “Ask a free quote” button, a “Quote request” popup (under construction, which will have to be integrated on the further milestone) will appear.

4. Search results



When a user searches for a specific list of sub-categories artists, each searched sub-category should appear in a separated tab, following the “Recommendations” tab.

Each sub-category tab should contain the list of all the artists matching the search criteria, only displaying the 5 first rows of 5 artist blocks. When the user scrolles down and reach the end of the displayed rows, the page will load the 5 following rows to display dynamically (without reloading the page) and the list should be refreshed as long as some matching criteria artists are not displayed.

Input

- ➢ http://www.acted.co/search URL.
- ➢ The design mockup of the search page, under the “search.html” file, in the Mockup repo.
- ➢ Acted database.

Expected output

A dynamically generated HTML document, respecting the “search.html” file design, filled with the requested information (stored in the database).
This dynamic page should allow the Acted users to search any artist, using search and filter criteria.

Core development

The "search.html" file will have to be split into the Twig templates, using the 3 levels layout inheritance:

1. App layout: 1st level containing the shared HTML structure for all the Acted website pages: the <html> tags, the <head> tags, the Acted blue menu bar (<nav> tag in the <header> tag) and the “by default” white Acted footer (Facebook, Instagram and Twitter icons + footer links and copyright message).
2. Bundle layout: 2nd level, inheriting from the “App layout”, containing the shared HTML structure for all the templates of the bundle (defining the few blocks architecture, composing the bundle).
3. Template layout: final structure of the template, inheriting from the “Bundle layout”, containing the content to be displayed.

Constraints

No commercial or licensed frameworks or API should be used to achieve this module.
Only open-source codes or frameworks could be integrated in the final module.

Test materials

No test materials.

Documentation

A Word document will be required to describe all the useful info about the 2 features of this specification, to explain, if it applies:

- What is the main goal of the feature?
- What are the feature inputs?
- What are the feature outputs?
- What are the main business rules implemented?

- Define a quick “How it works” schema.
- Which bundle handles the feature?
- What are the main classes?
- What are the services?
- What are the Twig templates?
- What are the add-on bundle you installed?
- What are the commands to run to install the feature?
- What are the commands to run to test the feature?
- How to test the feature?