A Festival Search Engine based on REST architecture

Nabil Abdennadher, Anthony Boulmier,

Description

The goal of this project is to create a concert/festival search engine that enables users to :

- locate concerts/festivals locations on an interactive map,
- display useful information about the concert/festival (location, online ticket shop, artists, etc.).
- listen to songs excerpts of artists who will attend the concert/festival,
- display useful information about the artists (type of music, most popular songs, hometown, etc.) who will attend the concert/festival

For that purpose, you will build a new service on top of "elementary" existing services. The software to develop is composed of two components:

- REST Server which is an aggregation of several elementary existing services.
- REST Client which is the GUI used to interact with the REST service.

Five existing REST services can be used:

- three concerts/festivals/events services: *BandsInTown*, *EventFul* and *JameBase*
- one music streaming service: *Spotify*.
- one service for artist information: *MusicBrainz*.

Here is a short description of these five elementary REST services:

Spotify *Spotify* is a Swedish music streaming service. It allows users to search for songs, artists, albums, and users' playlists in a media library. Moreover, many songs can be listened for 30 seconds for free. Registration is free. However, you need to ask for an application access-key/secret-key. https://developer.spotify.com

BandsInTown *BandsInTown* is a free concert/festival database. You can search for a concert/festival of an arbitrary artist. This service is public and do not require any registration. https://app.swaggerhub.com/apis/Bandsintown/PublicAPI/3.0.0

EventFul *EventFul* is larger than "BandsInTown". It contains also other types of events, such as arts exhibit, art festival and outdoor local festivals. The registration is free, However, you need to create an account and generate an API key. http://api.eventful.com

JamBase *JamBase* is, like BandsInTown, a concert/festival database. Data access requires a free registration. However, the number of queries per day is limited to 50. http://developer.jambase.com/

MusicBrainz *MusicBrainz* is a music artist information database that gives you access artists' information about the artists (birth date, aliases, hometown, etc.). You do not need to register to access to this service. https://musicbrainz.org/doc/Development

You are free to use any programming language and framework for this project. However, it's highly recommended to use Python with Flask for the back-end modules and JavaScript for the front-end.

Work planning

The work will be split into 5 steps

Step 1: Elementary REST services discovery. Deadline : 8th Nov. 2017

1. Browsing of the 5 elementary REST services, and selecting those that will be used within your project. You must use *Spotify* and *MusicBrainz*. For others, you must choose (and justify) which ones to use.
2. System Analysis: Enumerate the features your system will support, explain how these features will be designed: which elementary REST service to use for which feature? which elementary REST service will gather which information? What is the workflow of each of your features? You must justify your choices.
3. Selection of the technology you will use: Programming language, database (if any), framework (if any), etc. You must justify your choices.

Deliverable: A document that answers the questions listed in point 2.

Step 2: REST Client, version 0. Deadline: 15th Nov. 2017

1. Initialize a Google map with the locations of the concerts/festivals
2. Program the route that will randomly extract a song excerpts from a given concert/festival.

Deliverable: Software prototype of the Music Search Engine.

Step 3: Designing and coding the REST Server. Deadline: 22th Nov. 2017

1. Software Design: Represent each feature (see Step 1, point 2) of your REST service with one or several route. For each route, identify the elementary REST services to use. What kind of HTTP call for each feature? You must justify your choices.
2. REST API routes documentation: Create a document specifying, for each route: its purpose, its type (GET, POST, DELETE, HEAD?) and its parameters. An example must be given as an illustration. Use APIDOC tool.
3. Coding.

Deliverable: a web site describing the routes.

Step 4: Extending the REST Client (version 1). Deadline: 29th Nov. 2017

1. Extend the REST client developed in Step 2 to support the REST service developed in step 3.
2. When the user clicks on a concert/festival, the application must offer him the set of features you have defined (resp. described) in Step 1 (resp. Step 3).

Deliverable: A software (REST Client + REST server) fully operational.

Step 5: Documentation. Deadline: 6th Dec. 2017

1. Update and/or complete your documentation (Step 3)
2. Prepare your presentation.

Deliverable: Full documentation: web site (based on APIDOC)