2022

Assignment 2

K00237403
SAOIRSE DEELY

API Documentation

The purpose of this API is to retrieve information on various beers and their associated breweries. This API does not require an API key to use.

The base URL for the beer functionalities is as follows:

http://localhost:8888/beers

This URL will return an unpaginated JSON result of every beer in the database. Each beer contains a "self-link" which when queried will return the details of just that beer. They will also contain a link to the beer's details, these details include the name and description of the beer, as well as the name of the brewery which stocks the beer.

The self-link will appear in the format:

http://localhost:8888/beers/{id}

The beer which is returned from querying this self-link will also contain a link to all beers.

The details link will appear in the following format:

http://localhost:8888/beers/{id}/details

This link will return the name, description, and brewery name of the chosen beer.

http://localhost:8888/beers/{id}/images

This link will return a zipped file of all the beer images, unfortunately this functionality presently corrupts the returned image files.

http://localhost:8888/beers/{id}/{size}

This URL will return an image for the specified beer, the image will be a large or thumbnail size depending on the size specified in the request

http://localhost:8888/beers/{1}/pdf

This URL will return a brochure in pdf format for the specified beer, the brochure will contain an image, name, ABV, description, price, the name of the brewery the beer is stocked at, the website of the brewery, the category name of the beer, and the style name of the beer.

http://localhost:8888/beers/add

This method is a POST and it will add a new beer to the database, it consumes a JSONObject.

http://localhost:8888/beers/edit

This is a PUT method which will update the details of a beer, it consumes a JSONObject.

http://localhost:8888/beers/{id}/delete

This DELETE method will delete the specified beer from the database.

http://localhost:8888/beers/count

This GET method will return the total number of beers in the database.

The base URL for the breweries functionality is as follows:

http://localhost:8888/breweries

This URL will return all of the breweries in JSON format, each brewery will have a self-link which when queried will return the details of just that brewery, as well as a link to all of the beers which are stocked in that brewery

http://localhost:8888/breweries/{id}/map

This URL will return the name and address of the specified brewery, as well as a static image of that brewery plotted on a map. The return type is a static html page. The map image is obtained using the coordinates of the brewery and the Bing Maps RESTful API.

http://localhost:8888/breweries/{id}/qr

This link will return for a specified brewery, a QR code which when scanned with your mobile phone will attempt to add the brewery to your contacts.

http://localhost:8888/breweries/add

This URL is a POST method which will add a new brewery to the database, it consumes a JSONObject. This method was tested using the C# windows form client application that I developed as part of this project.

http://localhost:8888/breweries/edit

This URL is a PUT method which will update the details of the specified brewery, it consumes a JSONObject.

http://localhost:8888/breweries/{id}/delete

This is a DELETE method which will delete the specified brewery from the database.

http://localhost:8888/breweries/count

This method will return the total count of breweries in the database.

Self-Evaluation

The API I have developed is simple to use and quite functional, it is user friendly because it would not take long for a user to learn how to format their requests in order to get the necessary response, and nearly all responses are either JSON responses or static file responses, meaning it will be easy to parse and use the results of the API query. On the other hand, the error handling and validation of the API could be greatly improved. Currently there is bean validation for verifying the the fields of the models beer, brewery, breweries_geocode, category, and style are not blank, but further validation to ensure valid values could be implemented. As well as this, when a user sends a request which contains an invalid parameter, then a relevant HTTP response is sent back, such as “Not found”, more information could be provided to the user to make it easier to determine what has gone wrong and why, this would make the API more user friendly. There are a few issues with my API, the Zip file functionality has an issue which corrupts the image files returned, and the API currently does not run on the embedded TomCat server due to issues on my development machine. The results from the get all breweries and get all beers requests are not paginated,

this makes the results very difficult and cumbersome to search through. The pinning a brewery on a map functionality uses the breweries coordinates to do so, however some breweries do not have coordinates, so in this case it would be better to use the breweries address to pinpoint it on a map, currently an error is returned if the brewery does not have coordinates. The documentation for this API could also be greatly improved, currently it is just written documentation made by myself, but it would be an improvement to document the API using a 3rd party software such as Swagger 2, unfortunately I was unable to get this software to work.

Benchmarking and Enhancements

The benefits of developing an API are that the API can be more streamlined than developing a fully-fledged application for the same purpose, the API is more reusable and could be implemented by many different applications depending on its functionalities, and the API could be used as a bridge or connecting interface between other applications. APIs are more efficient and streamlined because they generally have a predefined purpose and set of functionalities which they perform, and their only goal is to perform these functions well. You do not need to worry about developing UI for the API, nor do you need to worry about what to do with the API. This will be the problem of the API users. APIs tend to be quite flexible and could be implemented and used in a wide variety of ways by client applications, they are useful to clients as the API will automate certain functionalities and this will free up resources for the client application. The execution speed of client applications can also be improved through the use of APIs as RESTful services generally execute asynchronously, meaning the client does not need to wait for these services to return a result, they may continue on with other functions.

There are endless other improvements which could be made on this API, but these are the ones which I feel are most important. The API I have developed could be improved by adding more error handling and validation of the requests sent by clients, and improving the API documentation to make it more clear to users how the API can be used. It would also be a large improvement to implement API keys, these keys would be sent with all requests to the API to authenticate the client and restrict the number of requests a client may make. This would prevent a client from spamming requests and possibly crashing the API.