Assignment 2 – Report

Calum Noble – K00239718

Table of Contents

Documentation

This is the documentation for a REST API for a fictitious beer importer. To ensure correct implementation of the API please read through the methods below. Any questions on anything within this API feel free to contact me via email at K00239718@student.lit.ie.

Beer Methods

count method – a GET method that counts the amount of beers in the table

http://localhost:8888/beers/count

This is how it’s called in the API. What it returns is the number of beers in the beers table.

delete method – a DELETE method that deletes a beer by id *

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

This is how it’s called in the API. What it does is deletes the beer by id entered. You must replace {id} with an id of a beer.

add method – a PUT method that adds a beer *

http://localhost:8888/beers

This is how it’s called in the API. What it does is adds a beer in the table.

edit method – a PUT method that edits a beer *

http://localhost:8888/beers

This is how it’s called in the API. What it does is edits a beer in the table.

getBeerImage method – a GET method that gets the image associated with the beer id

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

This is how it’s called in the API. This returns the image associated with the beer id entered, and also returns the image based on what size entered too. You must replace {size} with “large” or “thumbnail” and {id} with an id of a beer.

getAllBeers method – a GET method that gets all the beers in the table and returns them in JSON

http://localhost:8888/beers/listBeers

This is how it’s called in the API. This returns all the beers in the table in JSON format. It also includes a self-link and a link that if followed will display the beer name, beer description, and the name of the associated brewery.

beerInfo method – a GET method that gets three values associated with a beer id

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

This is how it’s called in the API. This returns three values (beer name, beer description, and associated brewery name) of the id entered in JSON format. You must replace {id} with an id of a beer.

Brewery Methods

getAllBreweries method – a GET method that gets all the breweries in the table and returns them in JSON

http://localhost:8888/breweries/listBreweries

This is how it’s called in the API. This returns all the beers in the table in JSON format

breweryMap method – a GET method that displays a Google Map with brewery address plotted on it

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

This is how it’s called in the API. This displays a Google Map with brewery address plotted on it. You must replaced {id} with an id of a brewery.

qrCode method – a GET method that returns a QR code with contact information of a brewery

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

This is how it’s called in the API. This displays a QR code with the contact information of the brewery associated with the id entered. You must replace {id} with a brewery id.

Self-Evaluation

Most work that people do has its fair share of strengths and weaknesses. The API I have created for this assignment is no different! An API is vital in the communication of two applications, therefore we must be aware of its strengths and weaknesses. I am going to discuss some strengths and weaknesses of the one I have developed.

We'll start with the weaknesses. In an ideal world I wouldn't have many, but we're not in an ideal world! A first weakness of my API would be the incompleteness of it. The API has two parts missing, one for zipping a folder of images, and the other a pdf brochure for a brewery. These parts were too complicated to figure out and I unfortunately ran out of time when attempting to figure them out. It's also missing specific error handling which doesn't help the API user to see where they are going wrong.

The second weakness would be how things display. I was unable to complete the pagination for the get all beers method. This is a pain as someone using the API has to load a multitude of records when using the get all beers method. The QR code generator also isn't ideal. I would've preferred to output it with some text on the same page, but that proved to be too difficult with the timeframe I had.

A third weakness that links to the QR code generator just mentioned is that when the QR code is scanned, it doesn't add the correct number format. It will still add the phone number to your contacts, but it doesn't have any prefix for the country, etc, meaning the contact won't be callable within your contact list. This is a minor thing, but a weakness no less.

The final weakness I will discuss is security. The API developed in assignment two is incredibly vulnerable as it has no security attached to it. Even as far as GitHub, I have my Google Maps API key on the web publicly available. The API is exposed in numerous ways. There is as no encoding and no authentication features meaning anyone who wants to access the data is able to do so with no issues. The API is prone to injections that could result in shutdown of systems. Luckily, there isn't much threat from this API currently as it isn't a publicly available API on the web, if it were it would be a disaster!

The strengths of the API are next. A first strength, and arguably the best one, is the things that can spring from the API that has been developed. The world of opportunity has now opened with the development of this API. It now enables developers to create more flexible applications with more automation involved. Having an API on hand for your application is a massive advantage.

Another strength of the API is the QR code generator. This was definitely the most satisfying part of the project to complete. It’s a great feature to be able to add the contact details of a brewery to your phone using a QR code.

The brewery map feature is a big strength of the API. It’s an API call that manages to be very personalised as it shows the individual brewery name and point on the map. By far a huge strength of the API as it has endless possibilities for the breweries that may use the API.

Overall, there are huge strengths but also weaknesses to the API. It was an enjoyable challenge creating the API and seeing everything come together.

Benchmarking and Enhancements

Creating an API will always bring benefits toward a project. It can firstly enable a level of automation within the project. Computers can manage the work rather than people with an API. Being able to enter your address to the API and it return a Google Map with your address pinned is a huge advantage to a project. Using the same example we can also say that an API can also bring a level of personalisation to a project. An API can also make further development of a project easier. It can make things in the future of the project just simpler.

The API created for this assignment is a great API, but I feel can benefit from further enhancements. On a level of personalisation for each brewery, we could add a feature that opens a “mail:to” to the brewery email. We could also implement a feature that gets the brewery phone number and returns it with prefix and country code of the users request, e.g, if a user requested an American brewery from Ireland it would return the American +1 prefix.

An obvious enhancement to the API would be an increase in security. Possibly creating some form of authentication, be it an API key or login etc. This would ensure that only people with permission can use the API. Adding best practices for API security overall would be ideal.