REST API documentation

1. **Add the necessary documentation blocks to your endpoints, including the expected URI parameters, request body, success response, response errors and examples of successful and successful responses.**
   ***Hint*: See examples**

   To install apidoc use:
   npm install apidoc -g

   And for the comments, I used the [apidoc documentation] as my guide.

Create endpoint

```
/**
 * @api {post} /create Request to add a city
 * @apiName CreateCity
 * @apiGroup City
 * @apiParam {String} city          Name of the city
 * @apiParam {String} countryCode   Country code of the city
 * @apiParam {String} district      District of the city
 * @apiParam {Number} population    Population of the city
 *
 * @apiSuccess {String} city          Name of the city
 * @apiSuccess {String} countyCode    Country code of the city
 * @apiSuccess {String} district      District of the city
 * @apiSuccess {Number} population    Population of the city
 *
 * @apiSuccessExample {json} Succes-Response:
 *  HTTP/1.1 200 OK
 *  {
 *      "city": "Luxembourg",
 *      "countryCode": "LU",
 *      "district": "Luxembourg",
 *      "population": "100000"
 *  }
 */
```

Read endpoint

```
/**
 * @api {get} /read/:city Request city information
 * @apiName GetCity
 * @apiGroup City
 * @apiParam {String} city Name of the city
 * @apiSuccess {String} city          Name of the city
 * @apiSuccess {String} countyCode    Country code of the city
 * @apiSuccess {String} district      District of the city
 * @apiSuccess {Number} population    Population of the city
 *
 * @apiSuccessExample {json} Succes-Response:
 *  HTTP/1.1 200 OK
 *  {
 *      "city": "Luxembourg",
 *      "countryCode": "LU",
 *      "district": "Luxembourg",
 *      "population": "100000"
 *  }

 */
```

Update endpoint

```
/**
 * @api {get} /updatePopulation/:city/:population Request to update the population of a city
 * @apiName UpdateCityPopulation
 * @apiGroup City
 * @apiParam {String} city Name of the city
 * @apiParam {Number} population of the city
 *
 * @apiSuccess {String} city          Name of the city
 * @apiSuccess {String} countyCode    Country code of the city
 * @apiSuccess {String} district      District of the city
 * @apiSuccess {Number} population    Population of the city
 *
 * @apiSuccessExample {json} Succes-Response:
 *  HTTP/1.1 200 OK
 *  {
 *      "city": "Luxembourg",
 *      "countryCode": "LU",
 *      "district": "Luxembourg",
 *      "population": "100000"
 *  }
 */
```

Delete endpoint

```
/**
 * @api {get} /delete/:city Request to delete a city
 * @apiName DeleteCity
 * @apiGroup City
 * @apiParam {String} city Name of the city
 *
 * @apiSuccess {Number} Integer 1, inticating that the deletion was successful
 *
 * @apiSuccessExample {json} Succes-Response:
 *  HTTP/1.1 200 OK
 *  {
 *      1
 *  }
 *
 * @apiError {Number} Integer 0, inticating that the deletion failed
 * @apiErrorExample {json} Error-Response:
 *  HTTP/1.1 200 OK
 *  {
 *      0
 *  }
 *
```

2. **Generate a webpage using the apidoc command. All endpoints should appear properly documented in that webpage. *Hint*: See apiDoc example for the kind of output you should produce.**

To generate the webpage you want to use the command:

apidoc -i . -o apidoc/

This will generate a directory called “apidoc” with an index.html file. To see the website you want to open the index.html file with a browser