CAB230 Assignment 3, 2024
Server-Side Web Application

Introduction

In the 2nd assignment, you developed a React application that allowed users to view and analyse data about volcanoes that we exposed via a REST API. In this assignment, your task is to write and deploy an Express application that replicates, and in some case extends, that REST API.

The extended API for this assignment is published and documented at **http://4.237.58.241:3001/** - you will notice additional routes relating to user profiles that were not present in Assignment 2, but are now required for this assignment. We will discuss these briefly below. In addition, you will be required to make your own decisions about how to expand the API further in order to meet the requirements for higher grade levels.

This assignment builds upon the lecture and practical material covered in the second half of CAB230. In particular, you must use the following technologies:

- Node
- Express
- MySQL (or MariaDB)
- Swagger (source code supplied alongside this specification, but you will have to extend it)
- Knex
- JSON Web Tokens

No alternatives to these core technologies are permitted. An essential part of this assignment is to deploy your Express application to the provided virtual machine that we have allocated to you. Information on this will be provided in Week 11.

Although this assignment shares a lot in common with the second assignment, insofar as we are using the same volcanoes dataset and working with a common API, your grade level in assignment two will have no bearing on your grade level for assignment three – so treat it as a clean slate!

The Data

The volcano database is provided as a SQL dump on the Canvas page for this assignment. You should import this database following a similar process to the "World Cities" database that we have been working on in the practicals. Once the dataset has been successfully imported, you should not touch the volcano data at all (except to add a table or tables for storing user details, and whatever additional columns or tables are necessary for adding your own custom functionality.)

The database consists of a single table, **volcanoes**, with the following columns:

- **id** – the id number of the volcano, used as the parameter to the /volcano/{id} endpoint
- **name** – the name of the volcano
- **region** – the volcano's location (typically country and continent, but can vary)
- **subregion** – more precise description of the volcano's location
- **last_eruption** – the year the volcano last erupted or Unknown. Suffixed with BCE or CE
- **summit** – the height of the volcano's summit, in metres above sea level

- **elevation** – the same value as the summit column, but measured in feet instead of metres
- **population_{5km,10km,30km,100km}** – the number of people that live within 5/10/30/100km of the volcano
- **latitude/longitude** – the location of the volcano

Open up the database in MySQL Workbench and look around to get a feel for the fields in the table and the types of values to be found in each. Note that we have not provided a table for storing user data – you will need to create this yourself. Additionally, note that advanced SQL knowledge is not required for any part of this assignment – it is not necessary to use JOINs, although you are permitted to use them.

The REST API

The API you are required to implement is identical to the one hosted at **http://4.237.58.241:3001/**, except for additional custom functionality (described later in this document).

The profile endpoints

You are required to provide, in addition to the routes in the Assignment 2 API, two additional endpoints: GET /user/{email}/profile and PUT /user/{email}/profile. These are documented in the Swagger documentation provided at **http://4.237.58.241:3001/**, so refer to this when implementing these routes.

The /me endpoint

The /me endpoint is another endpoint you must implement – it is used by us for marking purposes and without it, your submission will not be accepted. It's a very simple endpoint that simply returns a JSON object containing your name and student number, e.g.:

```
{
  "name": "Mike Wazowski",
  "student_number": "n1234567"
}
```

Custom endpoints

You are required to extend the API with your own additional functionality related to the overall purpose of the API. At a minimum, two additional endpoints are required:

- At least one GET endpoint returning information (e.g. about volcanoes, users, whatever) – it must take parameters (either path parameters or query parameters).
- At least one POST endpoint that requires authorisation (that is, it can only be used by logged-in users) and makes a modification of some sort to the underlying database. The request must require information to be passed in the body of the request (though path and query parameters may be used in addition to body data.)

These endpoints must feature suitable error checking, must produce appropriate error codes and must be documented by adding to the Swagger page served by your web app.

You may add additional tables to the database or additional columns to the existing tables to facilitate the new features you want to add.

Here are some potential ideas for additional functionality you might want to consider:

- Allow users to comment on volcanoes and other users to see the comments, as well as a ratings system as well so users can give a star rating and you can retrieve the comments and an average star rating of a particular volcano.
- Allow users to upload photos of volcanoes and other users to see the photos.
- Allow users to update volcano data (POST only, so you would need something else for the GET endpoint).
- Allow users to retrieve all volcanoes within a certain kilometre range of a longitude/latitude coordinate (GET only, so you would need something else for the POST endpoint).

These are just a few suggestions – use your creativity here. You will be marked on the implementation quality and sophistication of the functionality, as well as how relevant it is to the purpose of the web API.

We will use your Swagger documentation to learn about your additional functionality, so make sure that you document everything that you want us to know in the Swagger file and make sure that your Swagger file is served correctly; otherwise you may not receive marks for additional functionality even if it is present in your code.

Swagger documentation

When we visit the root of your site (e.g. `http://localhost:3000/`) your Swagger documentation should appear there. To serve it correctly and avoid interfering with your routing code, you should do something like this:

```
app.use('/', swaggerUI.serve);
app.get('/', swaggerUI.setup(swaggerDocument));
```

We provide an example Swagger file that you can use as a base, but you will have to extend it to document your custom endpoints.

Grade standards

We expect that you will follow a professional approach in the architecture and construction of the server. In particular:

- The routes and the overall structure are professional – logical and uncluttered, with appropriate use of specific routers mounted on the application
- There is appropriate use of middleware for managing components such as database connectivity and security
- There is appropriate error handling and the responses match those listed in the Swagger documentation
- That the Swagger documentation has been amended to include additional endpoints as meeting the requirements described above
- The application successfully serves the Swagger docs on the root of the site
- The application is successfully deployed to a QUT VM using HTTPS with a self-signed certificate
- There is appropriate attention to application security

Broadly speaking, the grade standards for this assignment correspond to the feature levels laid out below. Similarly to assignment two, the grade levels below assume that the features have been

implemented competently. If the implementation is substandard, the marks awarded will be reduced (perhaps substantially).

[Grade of 4 level]: Successful deployment to a QUT VM of an Express-based REST API which supports the basic endpoints (/countries, /volcanoes, /volcano/{id}), though authentication may be missing or with significant issues. The Swagger docs may not be served, or may simply be unaltered from the file we provide.

[Grade of 5 level]: All of the endpoints are implemented. Registration, login, JWT token handling and the profile routes must be present, operating correctly both when logged in and logged out. The Swagger docs are served, but may be unaltered from the file we provide.

[Grade of 6 or 7 level]: All of the endpoints have been successfully implemented, and additional custom endpoints have been implemented and documented on the Swagger page. The distinction between the 6 and 7 level is based on the sophistication and usefulness of the custom functionality, how well the API implements error responses, successful use of middleware security, database connectivity and the correctness of the Swagger docs. There is no one requirement that means the difference between a 6 or a 7 – these requirements are just to give you an idea of what we are expecting.

Unit tests

Unit tests (using the Jest unit testing framework) are provided on the assignment page on Canvas. You can follow the instructions provided in the unit test .zip to test your assignment once you have it running on your machine. We will use similar (but not identical) unit tests for evaluating your functionality for marking purposes.

Submission

You are required to both submit your assignment code to Gradescope (see submission instructions on Canvas) and deploy your assignment to the provided virtual machine. More details on the virtual machine will be made available in Week 11 – watch the Deployment videos and the lecture recording.

Unlike in Assignment 2, no report needs to be submitted this time – the only documentation you need to write is the modified Swagger file.

Acknowledgements

The volcanoes dataset is collated by the Smithsonian Institution's Global Volcanism Program and is publicly available at: https://volcano.si.edu/. We are providing a slightly modified version to better suit the learning objectives of this assignment. The relevant citation is:

Global Volcanism Program, 2013. Volcanoes of the World, v. 4.10.5 (27 Jan 2022). Venzke, E (ed.). Smithsonian Institution.