CSCI 4145 / 5409 – Summer 2020

Assignment 3

May 20, 2020 – Clarifications/Notes:

- When not actively using your resources, make sure that they are down/shut/off so that they do not consume resources.
- You may use any relational DB server you wish. In fact, we recommend storing your data on a managed relational DB as was shown in the tutorial – however, this is not a requirement.
- You may build your web service (develop your code using Node and Express) on AWS, as is shown in the tutorial, and then "containerize" it or you may use your previously built web service deployed in a container.

May 25, 2020 – no requirements changes, but a separate section highlighting any points of excellence or exceeding requirements may be included in the submission.

Technologies used: As before, but now a managed web service on AWS Cloud.

In this assignment you are asked to apply the skills you obtained through tutorials and previous assignments. In short, you are asked to perform tasks similar to those you did on Assignment 2, but instead of using FCS OpenStack cloud you will be creating a managed web services on AWS Elastic Beanstack (EB). If you not have AWS Educate Starter account, you will need to apply for it as it provides EB services required for this assignment (services that our AWS classroom account does not provide). When you use EB to deploy you web service, a lot of configuration is already done for you and you need not have created a VM instance for your web service – infrastructure to run your web service is provided transparently by AWS. Managed web services will be used to deploy your web service on MS Azure cloud.

In case you already used up your free credits for an AWS Educate Starter account for some other purposes (e.g., for assignments in another course), you need to email me and provide me with documentation (about the used up credits on your starter account) and you may deploy your web services on an AWS VM instance (and in your submission you need to also include the info about your used up credits).

SUBMISSION REQUIREMENTS

- *Code and Data Submission Requirements* - Your JS code must be such that
  - Any identifier in your code (including function names for non-anonymous functions) needs to end with the last three digits of your Student ID. Exceptions are identifiers such as i in looping constructs. For instance, my Banner/Dal ID ends with "007" and therefore any name of an identifier needs to end in "0007".
  - Any values for IDs of parts or jobs must end with the three digits of your ID. For instance, when creating an array containing the job information with two objects would be:
    - let jobs007 = [{ 'j1007', 1007, 55 }, { 'j2007', 2007, 66}] … where 'j1007' and 'j2007' are job IDs; 1007 and 2007 are part IDs; and 55 and 66 are quantities

  - If the above is not satisfied – the submission gets F.

- *File Names* - when submitting a file, its name should be "A-X-Team#-StudentID-LastName.zzz", where
  - X … is the assignment number (3 in this case)
  - Team# … is a two digit ID/number of your group/team (enter "00" if the group/team has not been formed as yet)

- StudentId … is your Dalhousie student (B00…) number
- LastName … is your last name as it appears on you Dal's transcript
- zzz … is a file extension, e.g., 'zip' for a zipped file or pdf, docx, …)

- Brief statement stating whether you used Beanstack or VM for your web service deployment (in which case provide information about the spent credits on your AWS Starter account).
- In a separate section, brief description of deficiencies, such as containers are not used for deployment, or that the job information is not stored in a DB, but rather in an array, or that some of your endpoints do not work fully (and describe what does not work).
- Optionally, in a separate section, brief description of what you did in excess of requirements. Here you describe what you feel you did in excess of meeting the basic requirements – briefly list it here. Use this to highlight what you did particularly well or in excess of the requirements. In particular, if you are proud of your work and feel it deserves more than satisfied, highlight here why. Examples of work in excess of requirements:
  - You created and used a managed DB (DBaaS), such as using an AWS RDB in the tutorial
  - You thoroughly tested the parameters
  - …
- Brief description of the major steps you performed to achieve your task – as an example of major steps, see tutorials. Your description should be supported by few judiciously taken screenshots that demonstrate the results of these steps (and that they worked).
- Include in your appendix your code for web service routers/endpoints (in a textual form, not as a screenshot).
- Screenshots that demonstrate that you have a docker file:
  - Show the relevant part of your directory structure showing in, in the listing of files, the docker file you created
  - Content of your docker file
- When invoking your web service, your screenshots should show the URL and parameters/arguments passed.
- The results of testing of your endpoints while making sure that the HTTP request URL and parameters are shown, as well as the results returned by endpoints.

Make sure that your web service is off/down, but that it is ready to be activated as your TA/marker may ask you for a live demo/testing of your assignment, including running and invoking your web service. In such a case your TA/marker would contact you and arrange for a mutually convenient time to meet on MS Team call to start the demo (of course, the mutually convenient time would have to be within a reasonable time-frame (i.e., you may not delay the demo too much) – after all, to run the demo, all you should have to do is to activate your web service/VM and your TA/marker should be able to issue HTTP requests via a browser or POSTMAN). She/he may ask you to examine/review your environment pertinent to the assignment, such as content of some of your directories.