Cluster and Cloud Computing Assignment 2 - Melbourne the Most Liveable City…?

Background

In development and delivery of non-trivial software systems, working as part of a team is generally (typically!) the norm. This assignment is very much a group project. Students will be put into software teams to work on the implementation of the system described below. These will be teams of up to 5 students. In this assignment, students need to organize their team and their collective involvement throughout. There is no team leader as such, but teams may decide to set up processes for agreeing on the work and who does what. Understanding the dependencies between individual efforts and their successful integration is key to the success of the work and for software engineering projects more generally. If teams have "*issues*", then please let me know asap and I will help resolve them.

Assignment Description

The software engineering activity builds on the lecture materials describing Cloud systems and especially the UniMelb Research Cloud (MRC - https://dashboard.cloud.unimelb.edu.au) and its use of OpenStack; on data from the Twitter APIs, and CouchDB and the kinds of data analytics (e.g. MapReduce) that CouchDB supports as well as data from the Australian Urban Research Infrastructure Network (AURIN – https://portal.aurin.org.au). The focus of this assignment is to harvest tweets from Melbourne on the MRC and undertake a variety of social media data analytics scenarios that explore liveability of Melbourne **and** importantly how the Twitter data can be used alongside/compared with/augment the data available within the AURIN platform to improve our knowledge of the liveability of Melbourne. Teams can download data from the AURIN platform, e.g. as JSON, CSV or Shapefiles, or using the AURIN openAPI (https://aurin.org.au/aurin-apis/). This data can/should be included into the team's CouchDB database for analysis with Twitter data.

The liveability of a city is based on a set of indicators including *housing, neighbourhood, transportation, environment, health, engagement,* and *opportunity* (see https://livabilityindex.aarp.org/how-are-livability-scores-determined). The goal here is to explore aspects of liveability through one or more of these indicators. You are not expected to develop scenarios covering them all.

The teams should develop a Cloud-based solution that exploits a multitude of virtual machines (VMs) across the MRC for harvesting tweets through the Twitter APIs (using both the Streaming and the Search API interfaces). A large corpus of historic Twitter data for Melbourne will also be provided. The teams should produce a solution that can be run (in principle) across any node of the MRC to harvest and store tweets and scale up/down as required. Teams have been allocated 4 servers (instances) with 8 virtual CPUs and 500Gb of volume storage. All students have access to the NeCTAR Research Cloud as individual users and can test/develop their applications using their own (small) VM instances, e.g., using personal instances such as pt-1234. (Remembering that there is no persistence in these small, free and dynamically allocated VMs).

The solution should include one or more Twitter harvesting applications for Melbourne, e.g., you may have one for housing related tweets and one for health-related tweets. Each team member should get one or more Twitter developer tokens to access the Twitter APIs as accounts are capped. The teams are expected to have multiple instances of this harvesting application running on the MRC together with an associated CouchDB database containing the amalgamated collection of Tweets. The CouchDB setup may be a single node or based on a cluster setup. The system should be designed so that duplicate tweets will not arise.

Students are free to explore other sources of data they find on the Internet, e.g. information on weather, sport events, etc however AURIN data has to be used for the scenarios.

Teams are expected to develop a range of analytic scenarios, e.g. using the MapReduce capabilities offered by CouchDB for social media analytics and comparing the tweets with official data from AURIN. Teams are free to explore any liveability scenarios that connect "in some way" to the AURIN data. Teams are encouraged to be creative here. *A prize will be awarded for the most interesting scenarios identified!* For example teams may look at scenarios such as:

- How many tweets mention Covid-19 or coronavirus and are these clustered in certain areas, e.g. rich vs poor suburbs or in areas where there are more/less hospitals/GP practices etc? (Health indicator)
- What do the movement patterns of people look like before Covid-19, during and after lockdown etc? (Transportation indicator)
- How many tweets mention topics related to the environment, e.g., air quality and how does this correlate with the traffic on the road network? (Environmental indicator)
- How many different languages are used for tweeting in given areas and how does this compare with the official statistics as captured by the Census? (Opportunity indicator)
- Are people tweeting positively or negatively regarding the house price bubble and is there a positive correlation with the house price growth in areas where prices have increased the most? (Housing indicator)

Some scenarios may require sentiment analysis of the Tweets. The above are examples – students may decide to create their own analytics based on the data they obtain. Students are not expected to build advanced "general purpose" data analytic services or complex compound indicators that can support any scenario but show how tools like CouchDB with targeted data analysis capabilities like MapReduce when provided with suitable inputs can be used to capture aspects of the essence of liveability in Melbourne.

A front-end web application is required for visualising these data sets/scenarios.

For the implementation, teams are recommended to use a commonly understood language across team members – most likely Java or Python. Information on building and using Twitter harvesters can be found on the web, e.g. see https://dev.twitter.com/ and related links to resources such as Tweepy and Twitter4J. Teams are free to use any pre-existing software systems that they deem appropriate for the analysis and visualisation capabilities, e.g. Javascript libraries, Googlemaps etc.

Error Handling

Issues and challenges in using the MRC for this assignment should be documented. You should describe the limitations of mining twitter content and language processing (e.g. sarcasm). You should outline any solutions developed to tackle such scenarios. The database may contain re-tweets. You should demonstrate how you tackled working within the quota imposed by the Twitter APIs through the use of the Cloud.

Final packaging and delivery

You should collectively write a team report on the application developed and include the architecture, the system design and the discussions that lead into the design. You should describe the role of the team members in the delivery of the system and where the team worked well and where issues arose and how they were addressed. The team should illustrate the functionality of the system through a range of scenarios and explain why you chose the specific examples. Teams are encouraged to write this report in the style of a paper than can ultimately be submitted to a conference/journal.

Each team member is expected to complete a confidential report on their role in the project and the experiences in working with their individual team members. This will be handed in separately to the final team report. (This is not to be used to blame people, but to ensure that all team members are able to provide feedback and to ensure that no team has any member that does nothing!!!).

The length of the team report is not fixed. Given the level of complexity of the assignment and total value of the assignment a suitable estimate is a report in the range of 20-25 pages. A typical report will comprise:

- A description of the system functionalities, the scenarios supported and why, together with graphical results, e.g. pie-charts/graphs of Tweet analysis and snapshots of the web apps/maps displaying certain Tweet scenarios;
- A simple user guide for testing (including system deployment and end user invocation/usage of the systems);
- System design and architecture and how/why this was chosen;
- A discussion on the pros and cons of the MRC and tools and processes for image creation and deployment;
- Teams should also produce a video of their system that is uploaded to YouTube (these videos can last longer than the MRC deployments unfortunately!);
- Reports should also include a link to the source code (github or bitbucket). It is recommended that all students commit their code to the code repository rather than delegate this to a single team member. This can provide an evidence base if teams have "issues".

It is important to put your collective team details (team, city, names, surnames, student ids) in:

- the head page of the report;
- as a header in each of the files of the software project.

Individual reports describing your role and your teams' contributions should be submitted through a Qualtrics link that will be sent through in due course.

Implementation Requirements

Teams are expected to use:

- a version-control system such as GitHub or Bitbucket for sharing source code.
- MapReduce based implementations for analytics where appropriate, using CouchDB's built in MapReduce capabilities.
- The system should have scripted deployment capabilities. This means that your team will provide a script, which, when executed, will create and deploy a virtual machine and orchestrate the set up of the software on said machine, e.g., a twitter harvester, that connects to Twitter and collects tweets that are fed into the existing CouchDB instance after running through a sentiment analyser etc. There is no need to dynamically deploy the whole system through the script. Note also that this setup need not populate the database but demonstrate the ability to orchestrate the necessary software environment on the MRC. Teams should use Ansible (http://www.ansible.com/home) for this task.
- Teams may wish to utilise container technologies such as Docker, but this is not mandatory.
- The server side of your analytics web application may expose its data to the client through a ReSTful design. Authentication or authorization is NOT required for the web front end.

Teams are also encouraged to describe:

- How fault-tolerant is your software setup? Is there a single point-of-failure?
- Can your application and infrastructure dynamically scale out to meet demand?

Deadline

One copy of the team assignment is to be submitted through Canvas. The zip file must be named with your team, i.e. *<CCC2022-TeamN>.zip*.

Individual reports describing your role and individual team member contributions should be submitted a Qualtrics link that will be distributed in due course. These individual reports will be completion of web-based forms, i.e. they do not require Word/PDF documents etc.

The deadline for submitting the team assignment is **Tuesday 10th May (by 12 noon!)**.

Marking

The marking process will be structured by evaluating whether the assignment (application + report) is compliant with the specification given. This implies the following:

- A working demonstration of the Cloud-based solution with dynamic deployment – **25% marks**
- A working demonstration of tweet harvesting and CouchDB utilization for specific analytics scenarios – **25% marks**
- Detailed documentation on the system architecture and design **– 20%**
- Report and write up discussion including pros and cons of the MRC and supporting twitter data analytics – **20% marks**
- Proper handling of the errors and removal of duplicate tweets – **10% marks**

The (confidential) assessment by your peers in your team on the Qualtrics system will be used to weight your individual scores accordingly. Timeliness in submitting the assignment in the proper format is important. **A 10% deduction per day will be made for late submissions.**

Demonstration Schedule and Venue

The student teams are required to give a *presentation* (with a few slides) and a demonstration of the working application. This may be in the lecture theatre or via zoom. Note that a single representative from each team should give the presentation, i.e. not all students are required to present. The presentation should include the key data analytics scenarios supported as well as the design and implementation choices made. Each team has **up to 15 minutes** to present their work. I will randomly identify a team on the day (using a random number generator for fairness!!!). Note the first presentations are on the same day as submission hence the deadline for submission is a hard one!!!

As a team, you are free to develop your system(s) where you are more comfortable with (at home, on your PC/laptop, in the labs...) but obviously the demonstration should work on the MRC.