Cluster and Cloud Computing Assignment 2 - Big Data Analytics on the Cloud

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 technologies taught in the lectures and demonstrated in the workshops including the Melbourne Research Cloud, Kubernetes, Docker, Fission and ElasticSearch. The project focuses on big data analytics using a range of data sets including: the Spatial Urban Data Observatory (SUDO - https://sudo.eresearch.unimelb.edu.au) and data **from one or more** other external data providers including *Mastodon*, *the Environmental Protection Agency (EPA)*, the *Bureau of Meteorology (BoM)* and potentially other sources, e.g. https://discover.data.vic.gov.au/. Students may also use data from Twitter (provided from assignment 1 and downloadable from SPARTAN). Teams should come up with one or more scenarios that combine and analyse these data sets. Examples of clients for accessing data from external resources were demonstrated in the workshops.

Teams can download data from the SUDO platform, e.g., as JSON, CSV or Shapefiles (see workshop week 2). It is noted that the list of all suburbs/cities across Australia can be obtained from SUDO, e.g. search for any SA2 data at the Australia level with SA2 name included.

Teams need to develop clients that harvest/stream data from one or more of the aforementioned external platforms and include them in the ElasticSearch database using Fission and Kubernetes..

External Data Providers

Mastodon - There are multiple Mastodon servers that can be used for data collection (see https://mastodonservers.net/). Teams are required to obtain an API Key to use these servers. Students should explore Mastodon servers that have a connection with Australia or a connection with the specific scenarios being explored, noting that Mastodon users may come from anywhere, i.e., Mastodon.au and Aus Social include users from around the world. Mastodon posts (toots) do not contain specific location information e.g. as lat/longs, but the information in a post or the user profile often contains information that could be used to loosely identify a given location, e.g. based on state, city or suburb name or indeed based on the Mastodon server itself, e.g. Aus Social.

Twitter – the 120Gb file is available from SPARTAN and could be used to augment the scenarios.

EPA (https://portal.api.epa.vic.gov.au/) – includes data sets related to air quality. An example of a client to access data from a single monitoring station was shown in the workshop. Teams may extend this to collect data from many other monitoring stations across Australia or from around areas of interest, e.g. Melbourne. Teams are required to obtain an API Key to access the EPA API. Multiple keys may be acquired/used by different team members depending on the scenarios.

BoM for data related to the weather across Australia. This might include the list of monitoring stations (http://reg.bom.gov.au/vic/observations/melbourne.shtml) as well as particular station observations (http://reg.bom.gov.au/fwo/IDV60901/IDV60901.95936.json).

Teams have been allocated 11 virtual CPUs and 700 GB of volume storage. Students may also 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-12345, noting that there is no persistence in these small, free and dynamically allocated VMs.

Teams are expected to develop a range of analytic scenarios comparing harvested/streamed external data with official data from SUDO. Teams are free to explore any scenarios that connect "in some way" to the SUDO 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:

- What impact does the weather have on people's mood in Melbourne or Sydney, i.e. are people happier when it is warm and sunny? Are they sad when it rains?
- What percentage of the population use social media?
- What is the correlation between the number of vehicles on the road and air quality? (See for example data from the Department of Transport and Planning - https://discover.data.vic.gov.au/).
- How does air quality change with the strength of the wind?
- Is there any correlation between air quality and lung-related disorders, e.g. asthma, COPD?

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 that can support any scenario but show how tools like ElasticSearch with targeted data analysis capabilities can be used to capture the essence of life in Australia. It is noted that the majority of SUDO data is older whilst weather, air quality, and social media posts are current. This is unimportant for the analysis, i.e. this assignment is to focus on Cloud development and not accurate social science research.

The front-end to the system should be based on a JupyterNotebook.

For the implementation, teams are recommended to use a commonly understood language across team members – most likely Python. Teams are free to use any pre-existing software systems that they deem appropriate for the analysis and visualisation, e.g., NLTK, Vader, TextBlob and existing topic models, e.g. LDA, BERT.

Error Handling

Issues and challenges in using the UniMelb Research Cloud for this assignment should be documented. You should describe the limitations of mining content and language processing (e.g., sarcasm). You should outline any solutions developed to tackle such scenarios.

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 that 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 their 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/maps of tweet/toot analysis for the scenarios;
- A discussion on the pros and cons of the UniMelb Research Cloud and tools and processes for image creation and deployment;
- Teams should also produce a video of their system demonstrating its core functionality. This video should be uploaded to YouTube (these videos can last longer than the UniMelb deployments unfortunately!) and the link for this included in the report;
- 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 recommended that students follow the code repository structure identified here:

  README.md: a document that contains a brief description of the repo contents, installation instructions, and instructions on how to use the client
  frontend: source code of the client part of the application (Jupyter notebooks)
  backend: the application back-end source code (harvesters, analytics, etc)
  test: the application back-end automated tests source code
  database: ElasticSearch type mappings, queries, etc.
  data: Any data you want to put in the code repository
  docs: Documentation (your report, docs on the API of your backend, etc)

It is important to put your collective team details (team number, 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 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.
- Use of Fission for stream data processing and data ingestion.
- Use of ElasticSearch for data management and analytics.
- Use of Kubernetes.
- The server side of your application should expose its data to the JupyterNotebook through a ReSTful design. Authentication or authorization is NOT required for the front end.
- Rich analytics and visualisation of data through a JupyterNotebook.

Teams are also encouraged to describe:
- How fault-tolerant is your software setup? Is there a single point-of-failure?
- Support for integrated software testing during application development.
- 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. *<CCC2024-TeamN>.zip*.

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

The deadline for submitting the team assignment is **Wednesday 22nd May (by 12 noon!)**. Note that this is a hard deadline as we are at the end of the course!

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 and use of Kubernetes – **25% marks**
- A working demonstration of stream data harvesting using Fission and use of ElasticSearch for specific data collection and analytics scenarios – **25% marks**
- Detailed documentation on the scenarios, the system architecture and the design **– 20%**
- Report and write up discussion including pros and cons of the UniMelb Research Cloud and the use of Kubernetes, Fission and ElasticSearch for big data analytics – **20% marks**
- Proper handling of errors – **10% marks**

The (confidential) assessment by your peers in your team 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. 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. This will take place on:

- **3.15pm-5.15pm 22nd May (8 teams present)**
- **12-1pm 23rd May (4 teams present)**
- **2.15-3.15pm 24th May (4 teams present)**
- **5.15-6.15pm 24th May (4 teams present)**

Note that given the numbers of teams this year, not all teams will be able to present – however all teams should be prepared to present. I will advise on Canvas how the randomised selection process will be arranged.

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