FuelCheck Data Retrieval and Processing

— COMP5339 Project Assignment Stage 2

In this data engineering project assignment, your team will retrieve, process and visualise real-time FuelCheck data from the New South Wales (NSW) government's website and API. The project aims to provide practical experience in core data engineering skills such as data retrieval, integration, cleaning, transformation, storage, and visualisation.

This is a **group assignment**, comprising teams of 4 to 5 students. By default, you will remain in the same group as in Stage 1, but you are free to switch groups (in assignment 2 groups setting) if you prefer.

The assignment includes two stages with separate submissions:

- Stage 1 (Week 8): Already passed.
- **Stage 2 (Week 12)**: Submit your Stage 2 report and code by **23:59, Thursday 22 May**.

All submissions will undergo plagiarism checks.

Stage 2 Tasks

For Stage 2, develop Python programs to complete the following tasks:

1. **Data Retrieval**
   Retrieve live fuel pricing across NSW service stations via Fuel API (https://api.nsw.gov.au/Product/Index/22). That is, use v1 endpoints.
2. **Data Integration and Storage**
   Combine the retrieved data into a single, consolidated dataset. During this process, you may need to clean and preprocess the data to ensure consistency and reliability. Tasks may include handling missing values, converting data types, and filtering out irrelevant or inconsistent data. Store the consolidated data into a single **csv** file.
3. **Data Publishing via MQTT**
   For each retrieved **price** record, publish a message to a MQTT server containing all the necessary information required for subsequent tasks. Introduce a **0.1**-second delay between each published message.
4. **Data Subscribing and Visualisation**
   Create a dashboard similar to the FuelCheck's web app. Specifically, subscribe to the MQTT messages published in the previous task. For each received message, dynamically add a marker to the map at the corresponding station location, following the dynamic rendering approach shown on the left side of https://folium.streamlit.app/dynamic_map_vs_rerender. Each marker should display the station's brand and the price of a **default** fuel type, which can be changed via a dropdown menu. When a user clicks a marker, it should display a pop-up with the station's name, address, and the latest fuel prices and update time for **all** available fuel types.

5. **Continuous Execution**
   Ensure your program for Task 1—3 runs continuously to simulate an unbounded data stream. Implement a **60**-second delay between each new round of API data retrieval; this delay is in addition to the 0.1-second delay between publishing messages.
6. **Documentation and Reporting**
   Maintain detailed documentation throughout your project workflow. Your final report should clearly summarise your methods for data retrieval, integration, publishing, subscribing and visualisation. Specifically:
   - Highlight key insights and findings.
   - Describe challenges encountered and how you overcame them.
   - Provide recommendations for future improvements.

Deliverables

Submit the following three types of files via Canvas:

1. Python files (**.py**)
   - Consolidate your Python scripts into as **few .py files** as possible (ideally **two**: one for Tasks 1—3 and one for Task 4). While **Streamlit** is highly recommended for building the dashboard, you are free to use other python frameworks if preferred.
   - Your program should be optimized to **minimize both the number of API calls** and the **volume of data retrieved from the API**. Only request the data you need, and avoid redundant or excessive queries.
2. **requirements.txt** file
   - Include a requirements.txt file listing all Python packages required to run your program within a clean Python virtual environment.
3. Project Report (**.pdf**)
   - Provide a clearly structured report of **up to 6 pages** (with optional additional appendix)
   - You report must include a short paragraph not exceeding 100 words, detailing the contribution of individual team members, example:
     - AXX: Led data collection and preprocessing, …
     - BXX: Developed the dashboard…
     - …

Submission Deadline

All Stage 2 deliverables are due by **23:59, Thursday 22 May.** Late submissions will incur penalties according to university policy.