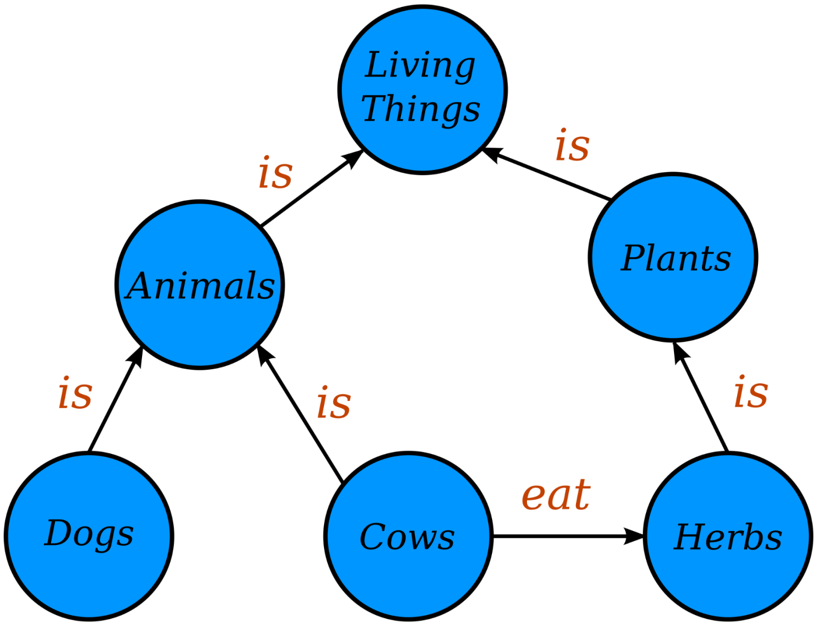
Ergodic - Full Stack Engineer task

Background

A knowledge graph is a knowledge base that uses a graph-structured data model or topology to represent and operate on data.
For instance,

“Animals and Plants are living things. Cows and Dogs are Animals. Cows eat Herbs, which is a Plant”

can be represented by the following graph.



Summary

As an engineer, you have built a system that collects knowledge graphs from text in a large organisation. This organisation in particular is a manufacturer, and the knowledge graph is related to their supply chain - namely: who are their suppliers and which products they manufacture.

You will find the knowledge graph here - kg.json. It is a dictionary of two components: nodes and edges. The data model for these is quite straightforward. In the same repository you will find a texts.json file, which contains a stream of texts.

Task

There are two parts of this task, one which involves a backend manipulation and querying of the graph object - and a front-end. You may use whatever stack you want for both the backend and the front-end. You can create a repo of your own and ship the code into this repo when you are finished.

For questions that don't require technical implementation you can write your thoughts in the README.md

If there's not enough time for everything, feel free to prioritise as you see fit. If you have any additional thoughts or think that there's something more exciting that can be done with the question, also feel free to go ahead and write it down / code it.

Questions

While there's a lot of information in these graphs, it's also very important for users to be able to:

- Understand all of the information related to entities in the graph, for instance:
  - Which products are manufactured by which supplier?
  - Where are certain products manufactured?
  - What would happen if there was a natural disaster in a given city?).

To do that, based on the data, can you come up with whichever endpoints you think could be interesting to facilitate the development of the application that queries this graph, motivated by these questions.

- Given two nodes, how are they related? What are all the intermediate edges and nodes that connect two nodes in the graph?

Furthermore:

- I have a heavy ML task which accepts a natural language question, the graph and returns a natural language output.

```
def my_heavy_task( question: str, graph: GraphModel ) -> str:
    time.sleep(1000)
    return "finished"
```

Once the user asks the question, we need to return a **task_id** for this job, and schedule it somehow so it doesn't block the entire API. How would you implement this - depending on the volume of queries and amount of data used.

- Take a look at the texts.json file, which contains the stream of texts. Some of the entities in the graph are mentioned in the text. For instance,

  *"**Global Raw Materials Inc**., located in **Country A**, is a leading supplier of essential raw materials, including **Aluminum** (Grade 6061-T6) and Polypropylene **plastic resin**."*

  Can you design an API that accepts a text and returns the entities mentioned in the text? How would you deal with this problem if there wasn't a 1-1 correspondence between the entities mentioned in the text and the name of the entities in the graph?