Data Engineer Technical Assessment

The purpose of this exercise is for you to demonstrate your technical abilities in setting up an end-to-end data pipeline. Some of the technologies we'd like you to use are specific, while others we leave open for interpretation, so you will get to choose.

From a high level, we will provide you with a dataset that you will produce to a message queue, consume that dataset from the message queue and do some manipulations, and send it downstream to a datastore. This should all be done locally via a set of docker containers that communicate with each other.

The deliverables for this assessment are a github repository with the following:

1. A docker compose file that will stand up your data pipeline end-to-end.
2. A set of helper scripts that will help you interact with the data pipeline. Potential helper scripts could do the following:
    a. Start the data pipeline.
    b. Stop the data pipeline.
    c. Produce the events to the data pipeline.
    d. Monitor the data pipeline.
    e. Give the status of all of the components of the data pipeline.
3. A README that will document how to use the docker compose file and helper scripts.
4. A design document describing your technical solution and an architecture diagram

Please be ready to explain your pipeline design process, as well as any factors that were taken into consideration.

Instructions:

1. Produce this dataset to a message queue of your choice using the producer or your choice.
    a. Potential message queues you could use are RabbitMQ, Redis, Kafka, etc.
    b. Potential producers you could use are Python, Fluentd, Logstash, etc.
2. Consume the data produced in step 1 from your message queue, manipulate it into the following format, and index the events into an OpenSearch cluster.
    a. Potential consumers you could use are Python, Fluentd, Logstash, etc.

```
Unset
{
    "time": 1426279439, // epoch time derived from the time field in the event
    "sourcetype": "nginx",
    "index": "nginx",
    "fields": {
        "region": "us-west-1",
        "assetid": "8972349837489237"
    },
    "event": {
        "remote_ip": "93.180.71.3",
        "remote_user": "-",
        "request": "GET /downloads/product_1 HTTP/1.1",
        "response": 304,
        "bytes": 0,
        "referrer": "-",
        "agent": "Debian APT-HTTP/1.3 (0.8.16~exp12ubuntu10.21)"
    } // this should be all of the data from the event itself, minus time
}
```

3. Using OpenSearch Dashboards, create a dashboard using the data you indexed into OpenSearch. What visualizations you add are up to you, but create something that is insightful for the data set that is provided.