**Open telemetry with Prometheus, Jaeger, and Grafana**

**Step 1: Write code for a flask app.**

from flask import Flask, render\_template, request, redirect, url\_for, flash

from prometheus\_flask\_exporter import PrometheusMetrics

from flask\_opentracing import FlaskTracing

from jaeger\_client import Config

app = Flask(\_\_name\_\_)

app.config['SECRET\_KEY'] = 'your\_secret\_key' # Replace with a real secret key

metrics = PrometheusMetrics(app)

# Jaeger configuration

config = Config(

config={

'sampler': {

'type': 'const',

'param': 1,

},

'logging': True,

},

service\_name='flask-app',

)

jaeger\_tracer = config.initialize\_tracer()

tracing = FlaskTracing(jaeger\_tracer, True, app)

@app.route('/', methods=['GET', 'POST'])

def hello\_world():

if request.method == 'POST':

name = request.form['name']

email = request.form['email']

# Place your data processing or saving logic here

print(f"Received submission: {name}, {email}") # Placeholder for saving data

with jaeger\_tracer.start\_span('form-submit') as span:

span.log\_kv({'event': 'form-submit', 'name': name, 'email': email})

flash('Form submitted successfully!', 'success')

# Stay on the same page without redirecting

return render\_template('form.html')

# Even if it's a GET request, we want to trace it

with jaeger\_tracer.start\_span('hello-world') as span:

span.log\_kv({'event': 'hello-world', 'value': 'user accessed /'})

return render\_template('form.html')

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

**Step 2: Create a Dockerfile for the flask app.**

# Use an official Python runtime as a parent image

FROM python:3.8-slim

# Set the working directory in the container

WORKDIR /app

# Copy the current directory contents into the container at /app

COPY . .

# Install any needed packages specified in requirements.txt

RUN pip install --no-cache-dir -r requirements.txt

# Make port 5000 available to the world outside this container

EXPOSE 5000

# Define environment variable

ENV NAME World

# Run app.py when the container launches

CMD ["flask", "run", "--host=0.0.0.0"]

**Step 3: Create a file named as requirements.txt which includes all the packages that need to be installed.**

**Step 4: Create a docker-compose file for flask-app, Prometheus, and jaeger. Prometheus will export metrics and jaeger will export traces.**

version: '3.7'

services:

flask-app:

build: ./flask-app

ports:

- "5000:5000"

environment:

JAEGER\_AGENT\_HOST: jaeger

JAEGER\_AGENT\_PORT: 6831

prometheus:

image: prom/prometheus

ports:

- "9090:9090"

volumes:

- ./prometheus/prometheus.yml:/etc/prometheus/prometheus.yml

depends\_on:

- flask-app

jaeger:

image: jaegertracing/all-in-one:latest

ports:

- "16686:16686"

- "6831:6831/udp"

**Step 5: Create a prometheus folder with a prometheus.yml configuration file as shown below.**

global:

scrape\_interval: 15s

scrape\_configs:

- job\_name: 'flask-app'

static\_configs:

- targets: ['flask-app:5000', 'localhost:9090',]

**Step 6: Run and start all the containers in the background using the command below.**

$ docker-compose up -d

**Step 7: Visualization and monitoring. Now, we can run the flask application by running it in the browser at:**

https://localhost:5000

For prometheus: http://localhost:9090

and jaeger: http://localhost:16686

**\*\*Note: Use IP address of the machine on which these services are running.**

**Step 8: Now, run grafana in a docker container and access it in a browser at: http://localhost:3000.**

You can add Prometheus and jaeger as data sources and visualize the metrics and traces in the Grafana.

**P.S.: You can clone the repository from this link and work on this on your local machine by making few changes in the IP addresses.**