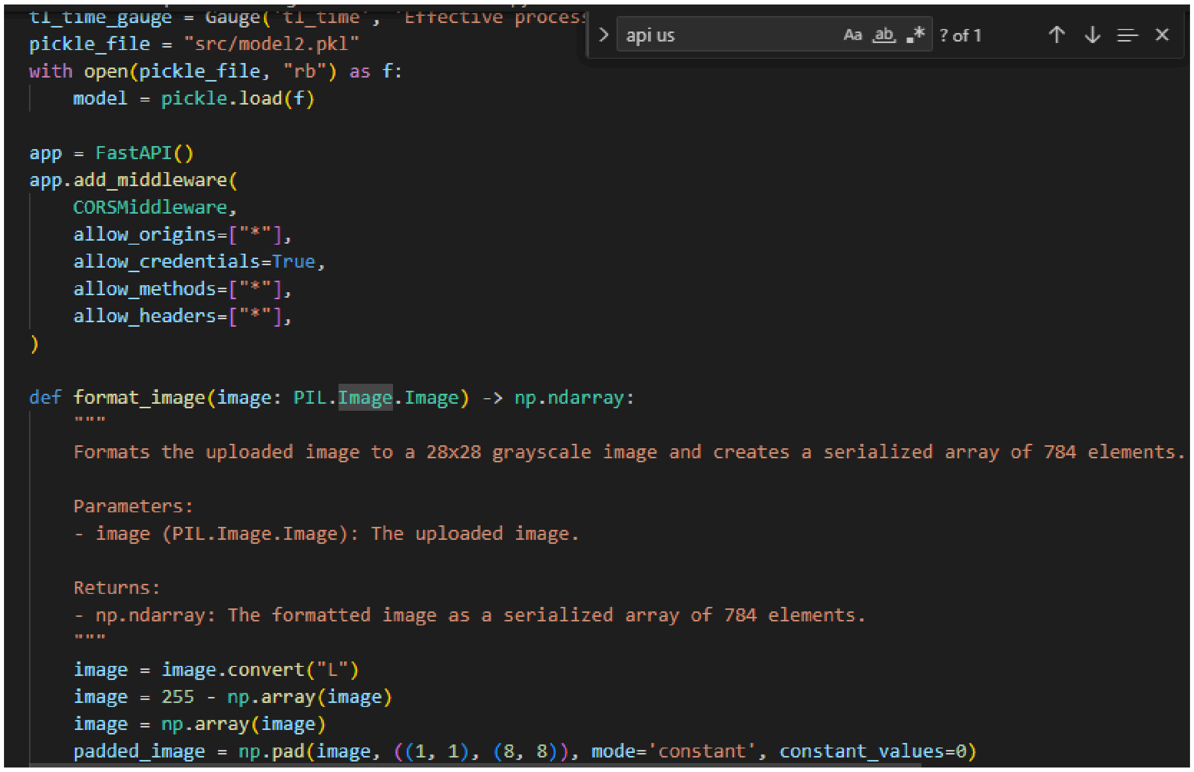
Report NA19B056 Assignment 7

Prometheus and Grafana Monitoring for FastAPI Application

Screenshots are within the report

This report outlines the setup and usage of Prometheus and Grafana for monitoring a FastAPI application. The objective is to monitor critical metrics such as API network I/O, CPU utilization, and memory usage.



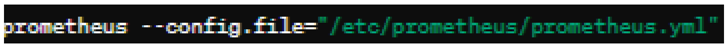
```
tl_time_gauge = Gauge('tl_time', 'Effective proces
pickle_file = "src/model2.pkl"
with open(pickle_file, "rb") as f:
    model = pickle.load(f)

app = FastAPI()
app.add_middleware(
    CORSMiddleware,
    allow_origins=["*"],
    allow_credentials=True,
    allow_methods=["*"],
    allow_headers=["*"],
)

def format_image(image: PIL.Image.Image) -> np.ndarray:
    """
    Formats the uploaded image to a 28x28 grayscale image and creates a serialized array of 784 elements.

    Parameters:
    - image (PIL.Image.Image): The uploaded image.

    Returns:
    - np.ndarray: The formatted image as a serialized array of 784 elements.
    """
    image = image.convert("L")
    image = 255 - np.array(image)
    image = np.array(image)
    padded_image = np.pad(image, ((1, 1), (8, 8)), mode='constant', constant_values=0)
```

Setting Up Prometheus

Prometheus is an open-source monitoring and alerting toolkit. It collects and stores metrics as time series data, providing a powerful query language (PromQL) to analyze this data.

Installing Prometheus

1. **Run Prometheus with the configuration file**:



```
prometheus --config.file="/etc/prometheus/prometheus.yml"
```

Configuring Prometheus

1. **Modify the Prometheus configuration file**:

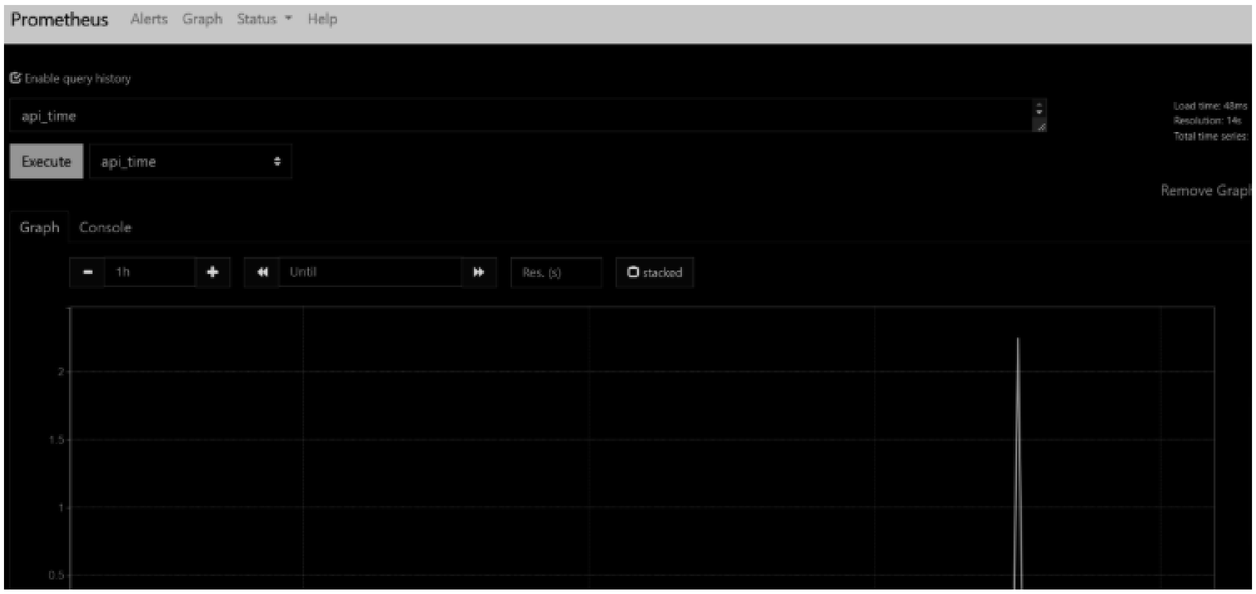
```
prometheus --config.file="/etc/prometheus/prometheus.yml"
```

Added the following scrape configuration to monitor the FastAPI application:

```
scrape_configs:
  - job_name: 'fastapi'
    static_configs:
      - targets: ['localhost:8000']
```

2. **Start Prometheus**:

```
prometheus --config.file=/etc/prometheus/prometheus.yml
```



```
api_time
```

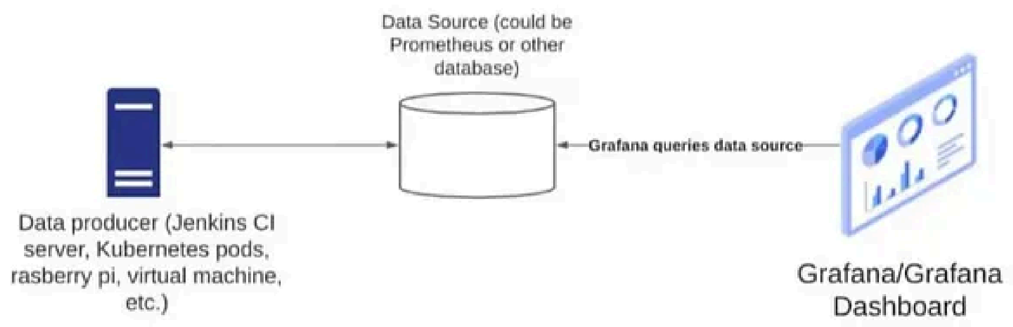
Selected: api_time · Range: 1h

```
Load time: 48ms
Resolution: 14s
Total time series:
```

Setting Up Grafana

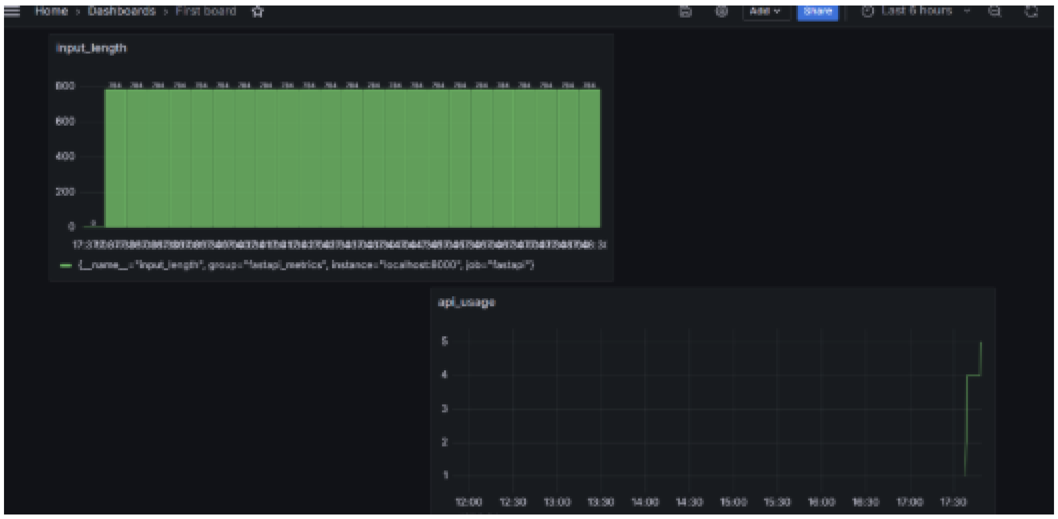
Grafana is an open-source platform for monitoring and observability. It allows you to query, visualize, alert on, and explore metrics from multiple data sources, including Prometheus.

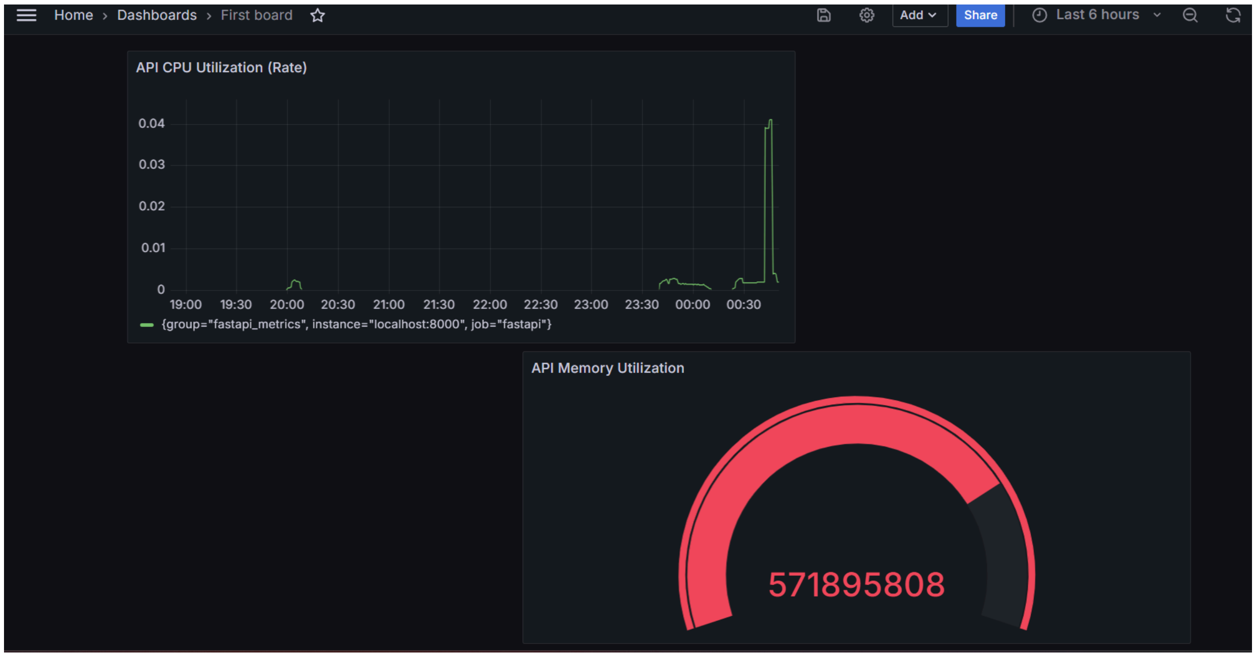
Grafana

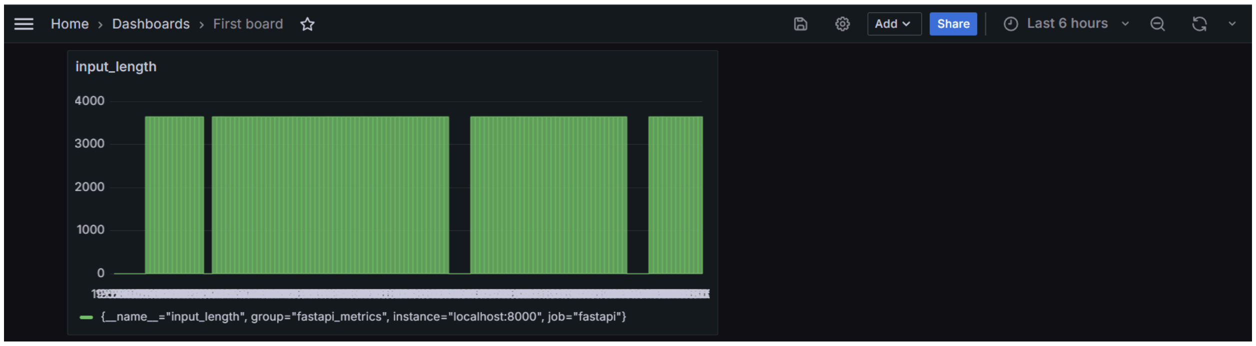

Data Source (could be Prometheus or other database)
Grafana queries data source
Data producer (Jenkins CI server, Kubernetes pods, rasberry pi, virtual machine, etc.)
Grafana/Grafana Dashboard

<tc.onyemaobi@gmail.com>

Source: Tech and Beyond With Moss


Home › Dashboards › First board
Add
Share
Last 6 hours
input_length
800
600
400
200
0
{__name__="input_length", group="fastapi_metrics", instance="localhost:8000", job="fastapi"}
api_usage
5
4
3
2
1
12:00
12:30
13:00
13:30
14:00
14:30
15:00
15:30
16:00
16:30
17:00
17:30




Home › Dashboards › First board
Add
Share
Last 6 hours
input_length
4000
3000
2000
1000
0
{__name__="input_length", group="fastapi_metrics", instance="localhost:8000", job="fastapi"}

Installing Grafana

1. **Download and install Grafana**:

```
wget https://dl.grafana.com/oss/release/grafana-7.3.6.linux-amd64.tar.gz
tar -zxvf grafana-7.3.6.linux-amd64.tar.gz
cd grafana-7.3.6
```

2. **Start Grafana**:

```
./bin/grafana-server
```

Configuring Grafana

1. **Access Grafana**: Navigate to local host in your web browser.

2. **Login**: Default credentials are `admin` / `admin`.

3. **Add Prometheus as a data source**:

   - Go to Configuration > Data Sources > Add data source.

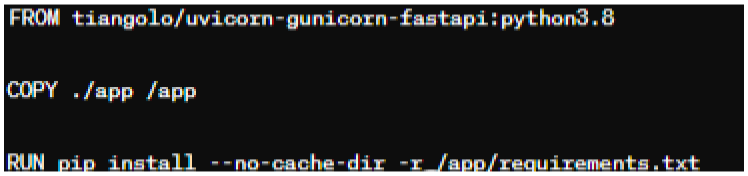
   - Select Prometheus and configure it with the local host URL.

4. **Create a Dashboard**: and add metrics you want to monitor.

Dockerizing the FastAPI Application

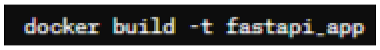
Docker allows you to package your application and its dependencies into a container, ensuring consistency across different environments.

1. **Create a Dockerfile**:



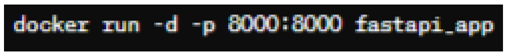
```
FROM tiangolo/uvicorn-gunicorn-fastapi:python3.8

COPY ./app /app

RUN pip install --no-cache-dir -r /app/requirements.txt
```

2. **Build the Docker image**:



```
docker build -t fastapi_app
```

3. **Run the Docker container**:



```
docker run -d -p 8000:8000 fastapi_app
```

Monitoring Metrics with Prometheus

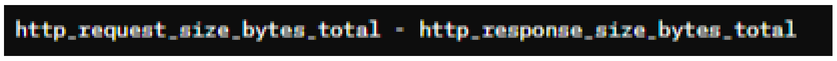
Prometheus scrapes metrics from the FastAPI application at regular intervals. The following metrics are collected and monitored:

API Network I/O Bytes

1. **Rate of API network I/O bytes**:

rate(http_request_size_bytes_created[5m]) - rate(http_response_size_bytes_created[5m])

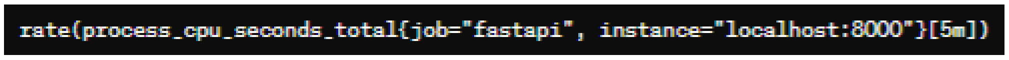
2. **Total API network I/O bytes**:



```
http_request_size_bytes_total - http_response_size_bytes_total
```

API CPU Utilization

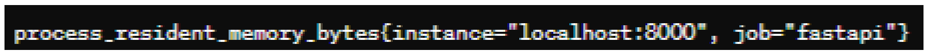
1. **Rate of CPU utilization**:



```
rate(process_cpu_seconds_total{job="fastapi", instance="localhost:8000"}[5m])
```

API Memory Utilization

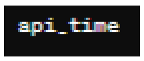
1. **Memory utilization**:



```
process_resident_memory_bytes{instance="localhost:8000", job="fastapi"}
```

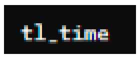
Additional Metrics

1. **API Time**: Monitor the time taken by the API to respond.



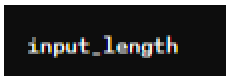
```
api_time
```

2. **Task Length Time**: Monitor the time taken to complete specific tasks.



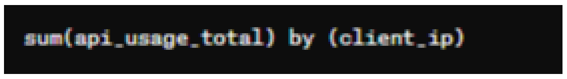
```
tl_time
```

3. **Input Length**: Monitor the length of input data.



```
input_length
```

4. **API Usage**: Sum of API usage by client IP.



```
sum(api_usage_total) by (client_ip)
```

Github Link: https://github.com/Rinkle-S/CS5830/tree/main/A07