DA5402 - Assignment 6

Anirudh Rao be21b004

Task 1

To scrape the required IO and CPU metrics, a function called `collect_iostat_metrics()` was created. This uses the `subprocess` library to run `iostat`. The output of this is parsed to identify the specific metrics. The five IO metrics are recorded for each device. In the case of CPU metrics, the average percentage is found for each mode except for the '`steal`' mode, as mentioned in the assignment. The scraped metrics are assigned to Prometheus gauges that are exposed at http://localhost:18000/ using a Prometheus client. This scraping is repeated every 1 second using a thread (from the `threading` library). Logging is implemented to record successful scraping or errors that occur during scraping.

Task 2

Memory information metrics are collected using a function called `collect_meminfo_metrics()`. This scrapes the metrics by reading and parsing the `/proc/meminfo` file into key-value pairs. The keys (e.g., `MemFree`, `Active(anon)`, `SwapTotal`) are normalized into Prometheus-friendly metric names that have the prefix '`meminfo_`'. This is done using a helper function called `normalize_meminfo_key()`. The scraped metrics are assigned to Prometheus gauges that are exposed at http://localhost:18000/ using a Prometheus client. This scraping is repeated every 1 second using a thread. Logging is implemented to record successful scraping or errors that occur during scraping.

Task 3

To setup a Prometheus server, a `prometheus.yml` file was created. This specifies the scrape interval as 2 seconds and the scraping target as http://localhost:18000/. The metrics exposed by the Python script can be queried at the Prometheus UI server. This server is available on http://localhost:9090/ after Prometheus is run using the provided `prometheus.yml` file.

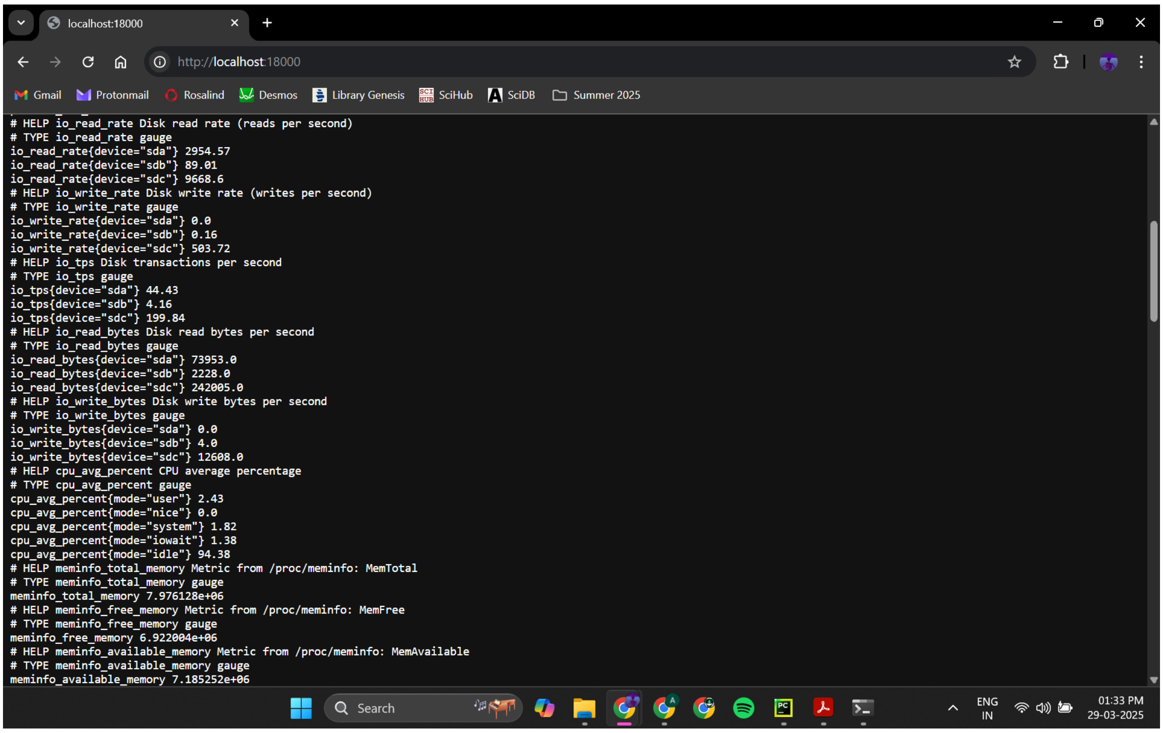
Files submitted

- `requirements.txt` – Text file that contains the Python libraries to be installed prior to running the script.
- `main.py` – Python script that contains the functions involved in scraping the metrics from the Linux environment. Exposes the scraped metrics at http://localhost:18000/ using a Prometheus client. Creates a logfile called `script.log` to collect logging information.
- `prometheus.yml` – YML file that contains the configuration for setting up the Prometheus server with the required scraping target and frequency. This server will be available on http://localhost:9090/.

Steps to run code

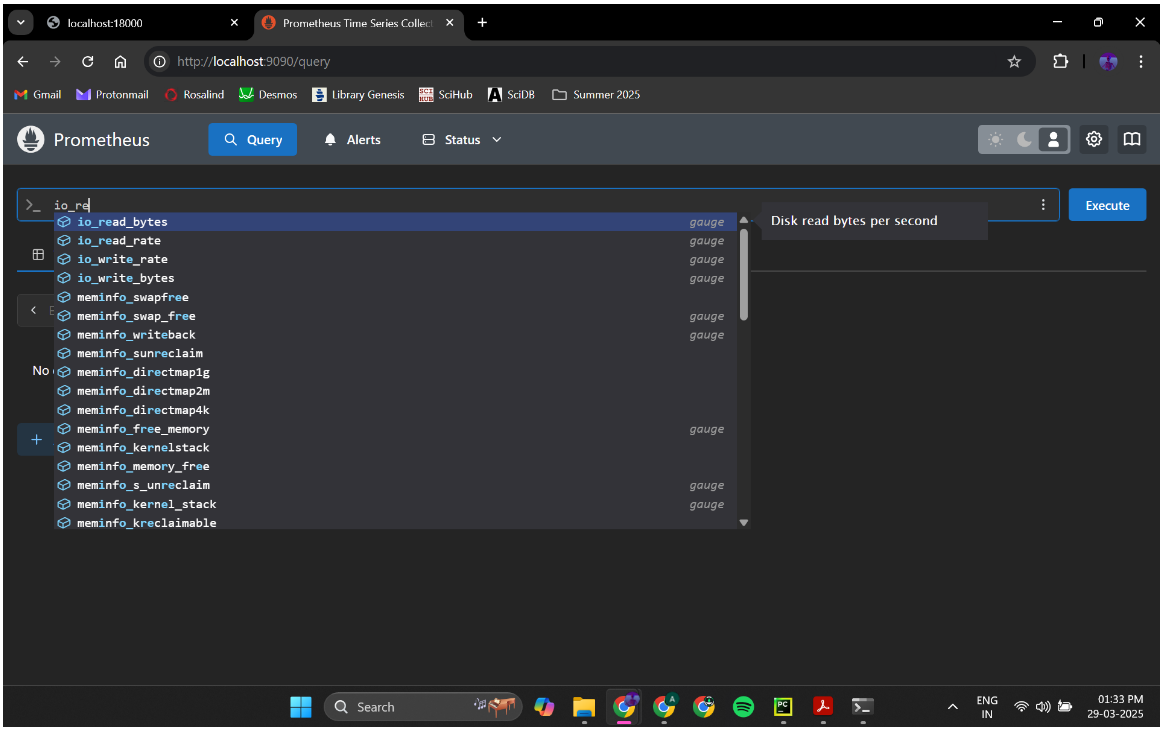
- Open a Linux environment.
- Ensure that Python3, `pip`, and Prometheus are installed in the Linux environment.
- Install the requirements given in `requirements.txt` using `pip`.
- Run `main.py`.
- Verify that `main.py` is running by checking http://localhost:18000/ or the `script.log` logfile.
- In the Linux environment, run Prometheus using the command `./prometheus --config.file=./prometheus.yml`.
- Check the Prometheus UI server at http://localhost:9090/ and query for the relevant metrics.
- To stop the collection of metrics, press Ctrl. + C for both the `main.py` script and the Prometheus server.

Sample outputs



```
# HELP io_read_rate Disk read rate (reads per second)
# TYPE io_read_rate gauge
io_read_rate{device="sda"} 2954.57
io_read_rate{device="sdb"} 89.01
io_read_rate{device="sdc"} 9668.6
# HELP io_write_rate Disk write rate (writes per second)
# TYPE io_write_rate gauge
io_write_rate{device="sda"} 0.0
io_write_rate{device="sdb"} 0.16
io_write_rate{device="sdc"} 503.72
# HELP io_tps Disk transactions per second
# TYPE io_tps gauge
io_tps{device="sda"} 44.43
io_tps{device="sdb"} 4.16
io_tps{device="sdc"} 199.84
# HELP io_read_bytes Disk read bytes per second
# TYPE io_read_bytes gauge
io_read_bytes{device="sda"} 73953.0
io_read_bytes{device="sdb"} 2228.0
io_read_bytes{device="sdc"} 242005.0
# HELP io_write_bytes Disk write bytes per second
# TYPE io_write_bytes gauge
io_write_bytes{device="sda"} 0.0
io_write_bytes{device="sdb"} 4.0
io_write_bytes{device="sdc"} 12608.0
# HELP cpu_avg_percent CPU average percentage
# TYPE cpu_avg_percent gauge
cpu_avg_percent{mode="user"} 2.43
cpu_avg_percent{mode="nice"} 0.0
cpu_avg_percent{mode="system"} 1.82
cpu_avg_percent{mode="iowait"} 1.38
cpu_avg_percent{mode="idle"} 94.38
# HELP meminfo_total_memory Metric from /proc/meminfo: MemTotal
# TYPE meminfo_total_memory gauge
meminfo_total_memory 7.976128e+06
# HELP meminfo_free_memory Metric from /proc/meminfo: MemFree
# TYPE meminfo_free_memory gauge
meminfo_free_memory 6.922004e+06
# HELP meminfo_available_memory Metric from /proc/meminfo: MemAvailable
# TYPE meminfo_available_memory gauge
meminfo_available_memory 7.185252e+06
```

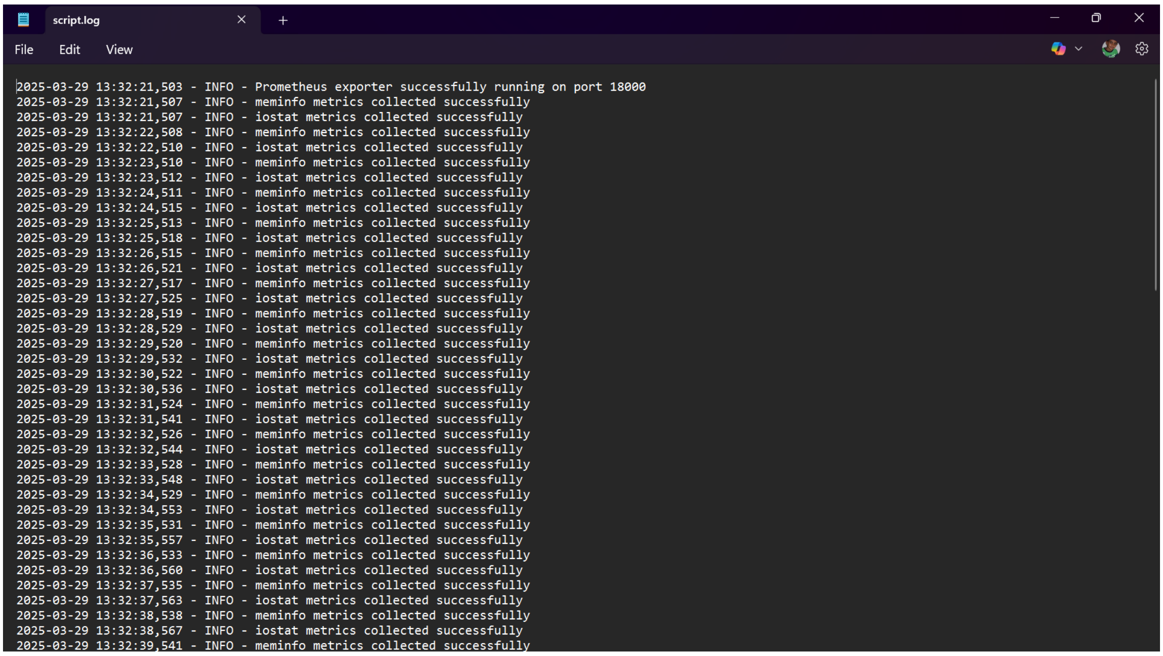
Figure 1: The scraped metrics exposed at http://localhost:18000/



```
>_ io_re
io_read_bytes                                  gauge
io_read_rate                                   gauge
io_write_rate                                  gauge
io_write_bytes                                 gauge
meminfo_swapfree
meminfo_swap_free                              gauge
meminfo_writeback                              gauge
meminfo_sunreclaim
meminfo_directmap1g
meminfo_directmap2m
meminfo_directmap4k
meminfo_free_memory                            gauge
meminfo_kernelstack
meminfo_memory_free
meminfo_s_unreclaim                            gauge
meminfo_kernel_stack                           gauge
meminfo_kreclaimable
```

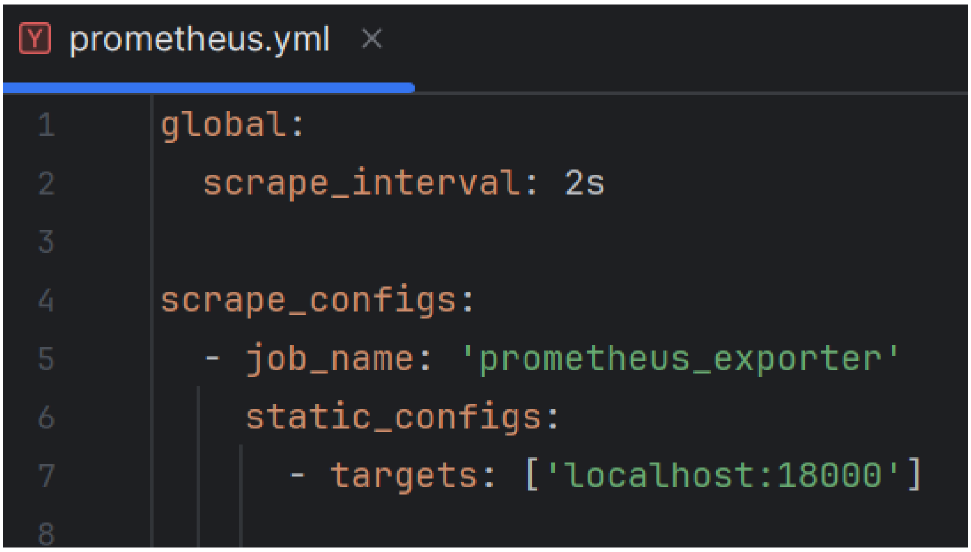
Disk read bytes per second

Figure 2: Querying the metrics in the Prometheus UI server at http://localhost:9090/



```
2025-03-29 13:32:21,503 - INFO - Prometheus exporter successfully running on port 18000
2025-03-29 13:32:21,507 - INFO - meminfo metrics collected successfully
2025-03-29 13:32:21,507 - INFO - iostat metrics collected successfully
2025-03-29 13:32:22,508 - INFO - meminfo metrics collected successfully
2025-03-29 13:32:22,510 - INFO - iostat metrics collected successfully
2025-03-29 13:32:23,510 - INFO - meminfo metrics collected successfully
2025-03-29 13:32:23,512 - INFO - iostat metrics collected successfully
2025-03-29 13:32:24,511 - INFO - meminfo metrics collected successfully
2025-03-29 13:32:24,515 - INFO - iostat metrics collected successfully
2025-03-29 13:32:25,513 - INFO - meminfo metrics collected successfully
2025-03-29 13:32:25,518 - INFO - iostat metrics collected successfully
2025-03-29 13:32:26,515 - INFO - meminfo metrics collected successfully
2025-03-29 13:32:26,521 - INFO - iostat metrics collected successfully
2025-03-29 13:32:27,517 - INFO - meminfo metrics collected successfully
2025-03-29 13:32:27,525 - INFO - iostat metrics collected successfully
2025-03-29 13:32:28,519 - INFO - meminfo metrics collected successfully
2025-03-29 13:32:28,529 - INFO - iostat metrics collected successfully
2025-03-29 13:32:29,520 - INFO - meminfo metrics collected successfully
2025-03-29 13:32:29,532 - INFO - iostat metrics collected successfully
2025-03-29 13:32:30,522 - INFO - meminfo metrics collected successfully
2025-03-29 13:32:30,536 - INFO - iostat metrics collected successfully
2025-03-29 13:32:31,524 - INFO - meminfo metrics collected successfully
2025-03-29 13:32:31,541 - INFO - iostat metrics collected successfully
2025-03-29 13:32:32,526 - INFO - meminfo metrics collected successfully
2025-03-29 13:32:32,544 - INFO - iostat metrics collected successfully
2025-03-29 13:32:33,528 - INFO - meminfo metrics collected successfully
2025-03-29 13:32:33,548 - INFO - iostat metrics collected successfully
2025-03-29 13:32:34,529 - INFO - meminfo metrics collected successfully
2025-03-29 13:32:34,553 - INFO - iostat metrics collected successfully
2025-03-29 13:32:35,531 - INFO - meminfo metrics collected successfully
2025-03-29 13:32:35,557 - INFO - iostat metrics collected successfully
2025-03-29 13:32:36,533 - INFO - meminfo metrics collected successfully
2025-03-29 13:32:36,560 - INFO - iostat metrics collected successfully
2025-03-29 13:32:37,535 - INFO - meminfo metrics collected successfully
2025-03-29 13:32:37,563 - INFO - iostat metrics collected successfully
2025-03-29 13:32:38,538 - INFO - meminfo metrics collected successfully
2025-03-29 13:32:38,567 - INFO - iostat metrics collected successfully
2025-03-29 13:32:39,541 - INFO - meminfo metrics collected successfully
```

Figure 3: The `script.log` file capturing the logging information



```yaml
global:
  scrape_interval: 2s

scrape_configs:
  - job_name: 'prometheus_exporter'
    static_configs:
      - targets: ['localhost:18000']
```

Figure 4: The `prometheus.yml` file containing the configuration for the Prometheus server