

**SCHOOL OF INFORMATICS & IT**

**Diploma in Information Technology**

**Part 4a: Continuous Integration/ Continuous Deployment (CI/CD) –**

**Monitoring & Additional features**

Student Name (Matric Number) : 2304532A

Tutorial Group : PC02

Jenny LING (TP)

Tutor : Mr Ryan Justin

Submission Date : 10/02/2025

**Declaration of Originality**

I am the originator of this work and I have appropriately acknowledged all other original sources used as my references for this work.

I understand that Plagiarism is the act of taking and using the whole or any part of another person’s work, including work generated by AI, and presenting it as my own.

I understand that Plagiarism is an academic offence

and if I am found to have committed or abetted the offence of plagiarism in relation to this submitted work, disciplinary action will be enforced.

**Note to Tutor: This page is only to be used if your subject specifically allows AI-generated content to be used. Please delete this page if it is not required.**

**Declaration on the use of Generative AI tools for assignments**

|  |
| --- |
| Describe how you have used Generative AI tools such as ChatGPT or Dall.E- in your assignment.  Share the link to the conversations you had with the AI tool (i.e., the prompts you used and the responses you get from the AI tool).  **Please refer to this PDF on “How to share the conversations made with ChatGPT?”** |
|  |
| How do you indicate the reference?  The content generated by AI tools are not retrievable except by the user who generated them, so they are considered non-recoverable sources. Although non-recoverable data or quotations in APA Style papers are usually cited as personal communications, with ChatGPT-generated text there is no person communicating. Quoting text from ChatGPT chat is therefore more like sharing the output of an algorithm, with a reference list entry and the corresponding in-text citation.  According to the official APA Style site, ChatGPT references should be cited as:  E.g. OpenAI. (2024). *ChatGPT* (Feb 13 version) [Large language model].  <https://chat.openai.com/chat>  Note: The information in parentheses refers to the update or revision date of the model used. Refer to the release notes in the ChatGPT application. |

**Important Note:**

* Do not copy answers produced by the AI tool in totality as it is considered as plagiarism.
* Do not rely on any information produced by the AI tool blindly. You should always verify the answer with other sources. Do not assume that these answers provided by the AI tool are correct.
* To achieve quality outputs from the AI tool, you should provide good prompt that is clear and specific. Be precise and provide context. Avoid asking open-ended questions.

Table of Contents

**Declaration of Originality** 0

Monitoring Implementation 1

Additional Features 2

Plan-Perform-Monitor-Reflect (PPMR) Journal 3

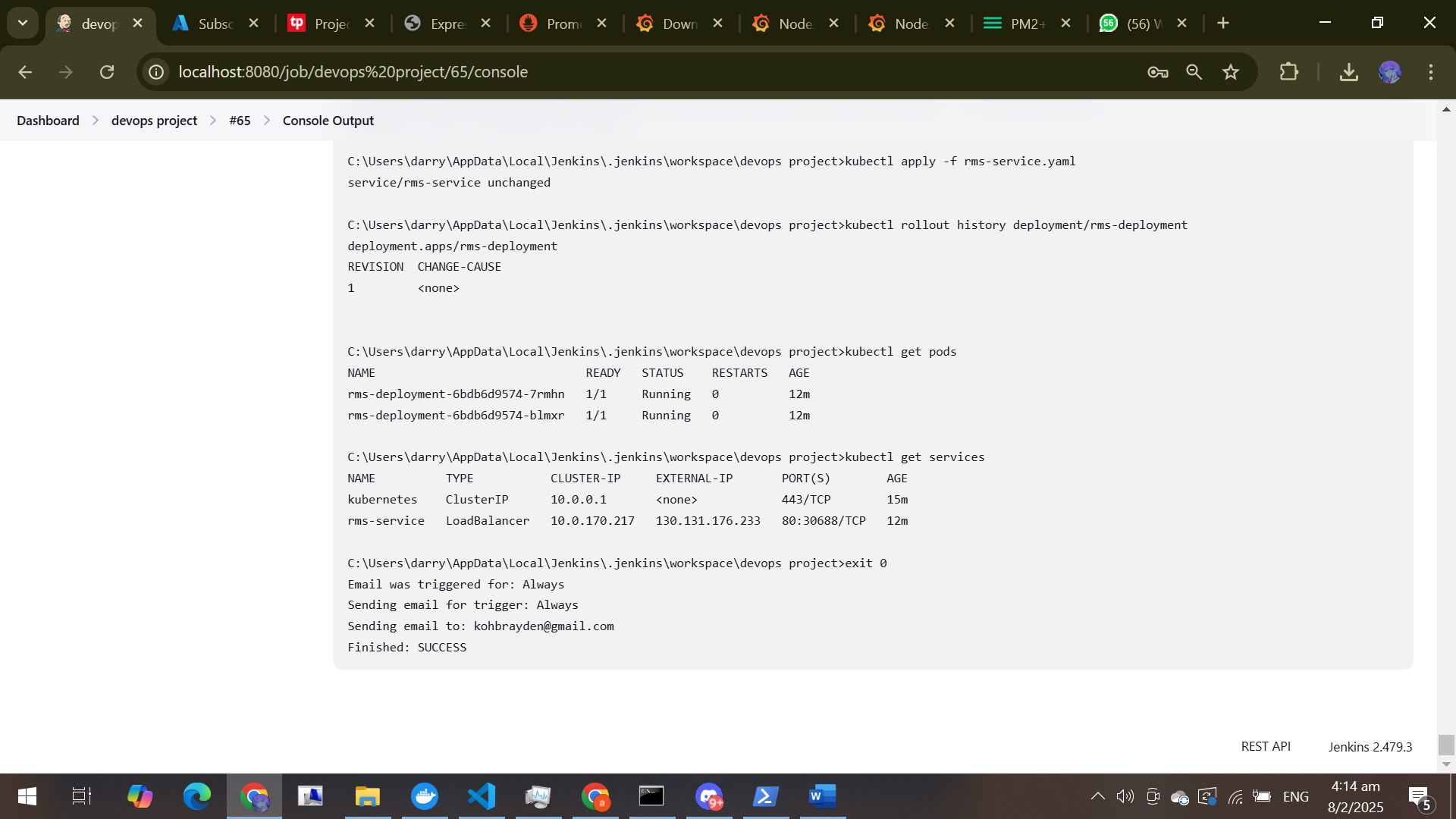
You are NOT allowed to modify the page margins, font sizes and line and paragraph spacings.

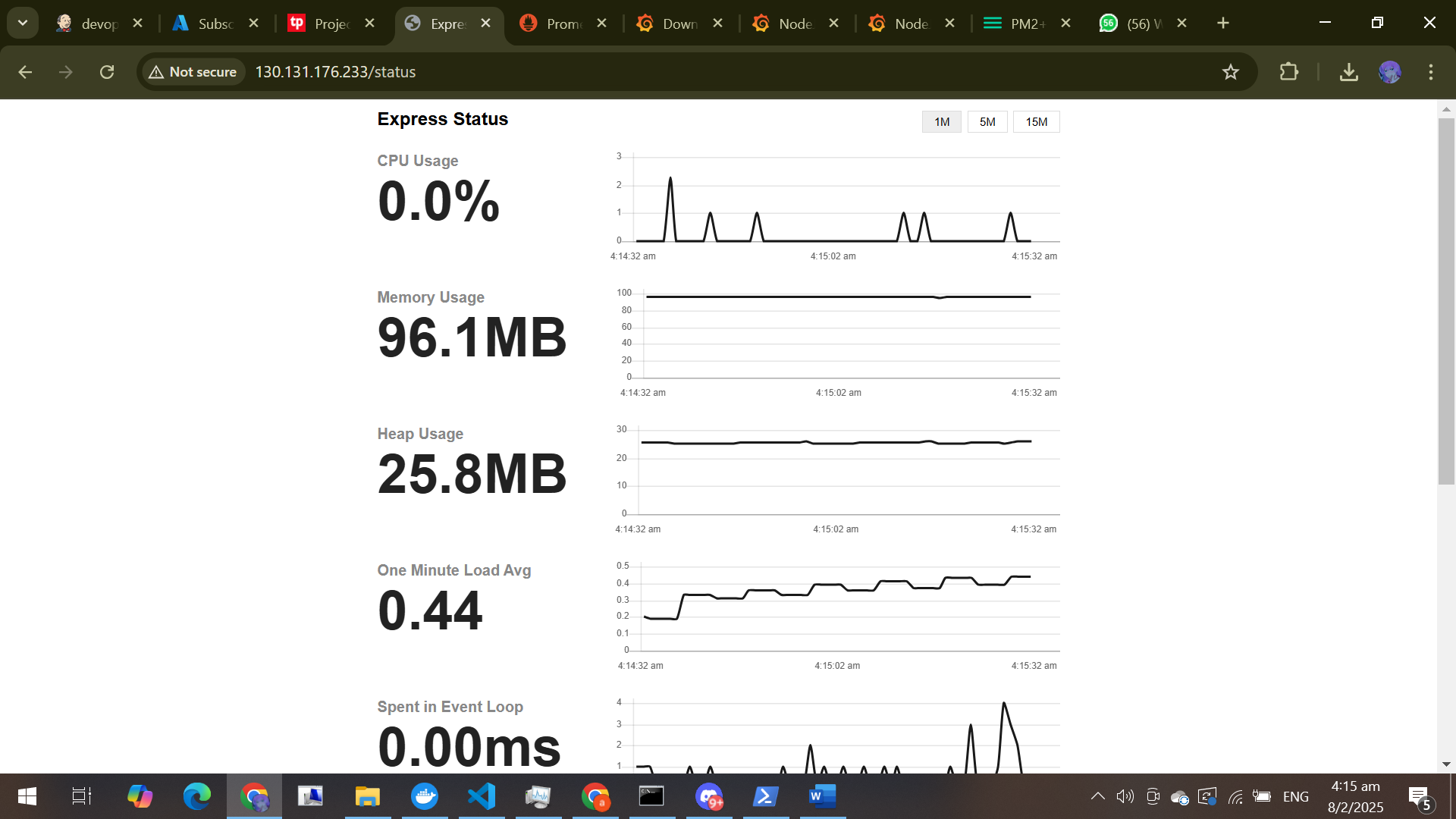
Please remove all prompting questions and guidelines in each section before submitting.

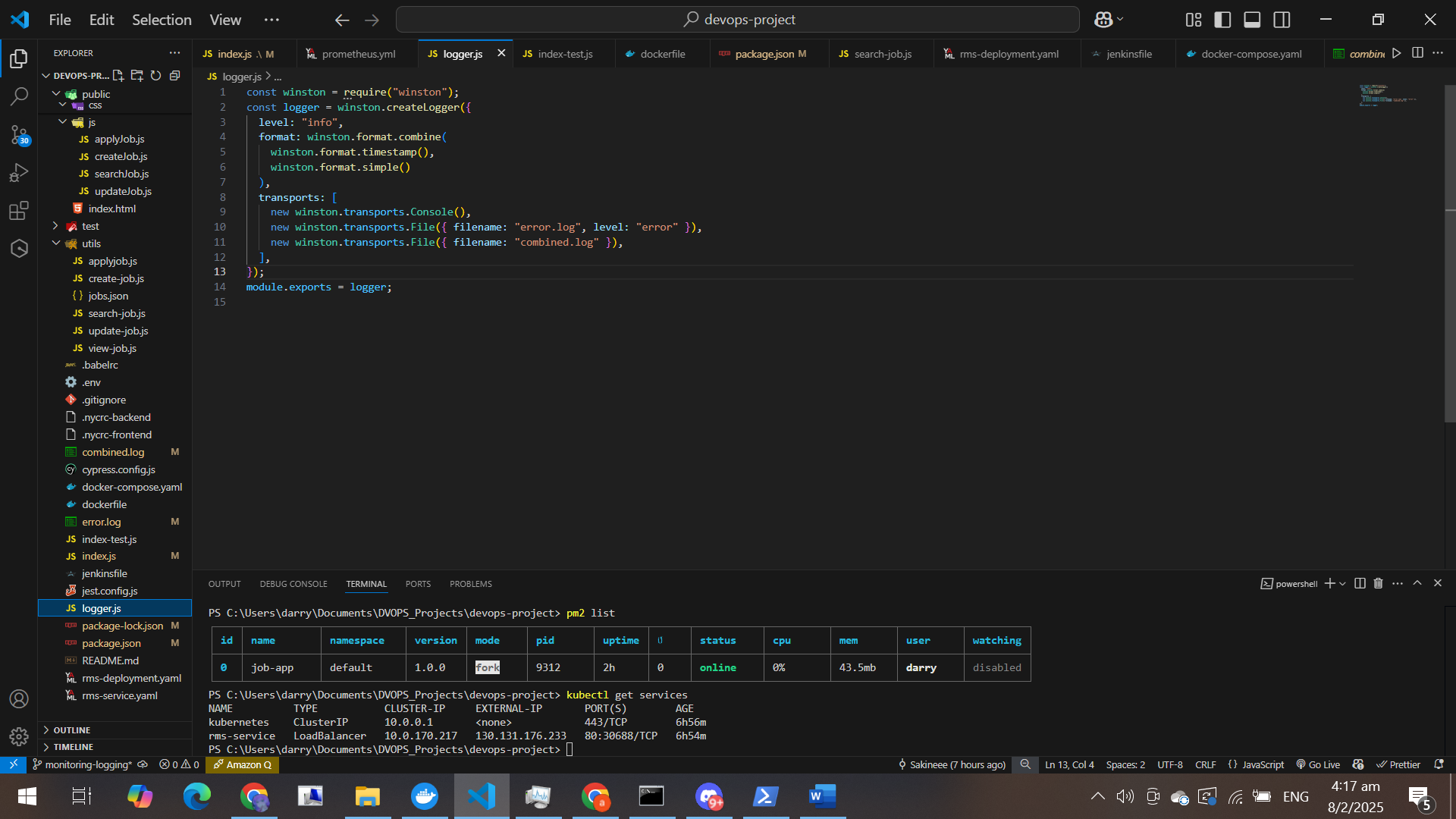
Report should NOT exceed 8 pages.

# Monitoring Implementation

Include evidence(s) such as screenshots, configuration files, and/or sample monitoring data showcasing your monitoring setup.







# Additional Features

**Implementation of Prometheus + Grafana**

**Introduction**

To further enhance observability, I implemented Prometheus + Grafana as a monitoring and visualization stack. Prometheus is an open-source monitoring system that collects and stores metrics as time-series data, while Grafana is a powerful visualization tool that enables me to create interactive dashboards based on these metrics.

1. **Implementation Steps**

**1. Installing Prometheus on Windows**

I started by downloading Prometheus from the official site:

1. Navigated to Prometheus Downloads
2. Downloaded the Windows zip file
3. Extracted the contents and navigated to the folder
4. Opened PowerShell in the extracted folder and ran:
5. .\prometheus.exe --config.file=prometheus.yml

**2. Configuring Prometheus to Scrape Application Metrics**

I updated the prometheus.yml configuration file to scrape metrics from my application:

scrape\_configs:

- job\_name: 'job-app'

static\_configs:

- targets: ['localhost:3000']

This ensured that Prometheus periodically pulled metrics from my Node.js application.

**3. Setting Up Grafana on Windows**

To install and start Grafana:

1. Downloaded Grafana from Grafana Downloads
2. Extracted the zip file and navigated to the bin folder
3. Ran grafana-server.exe
4. Accessed Grafana at http://localhost:3000 and logged in with the default credentials (admin/admin)
5. Added Prometheus as a data source

**4. Creating Grafana Dashboards**

I created custom dashboards in Grafana to visualize:

* API response times
* Server uptime
* Memory and CPU usage
* Active HTTP requests

These dashboards allowed for real-time tracking and historical trend analysis.

1. **Evidence of Implementation**

**1. Prometheus Metrics Endpoint**

To confirm Prometheus was collecting metrics, I accessed:

http://localhost:9090/metrics

This displayed raw metric data, confirming that Prometheus was pulling system metrics.

**2. Grafana Dashboard Screenshots**

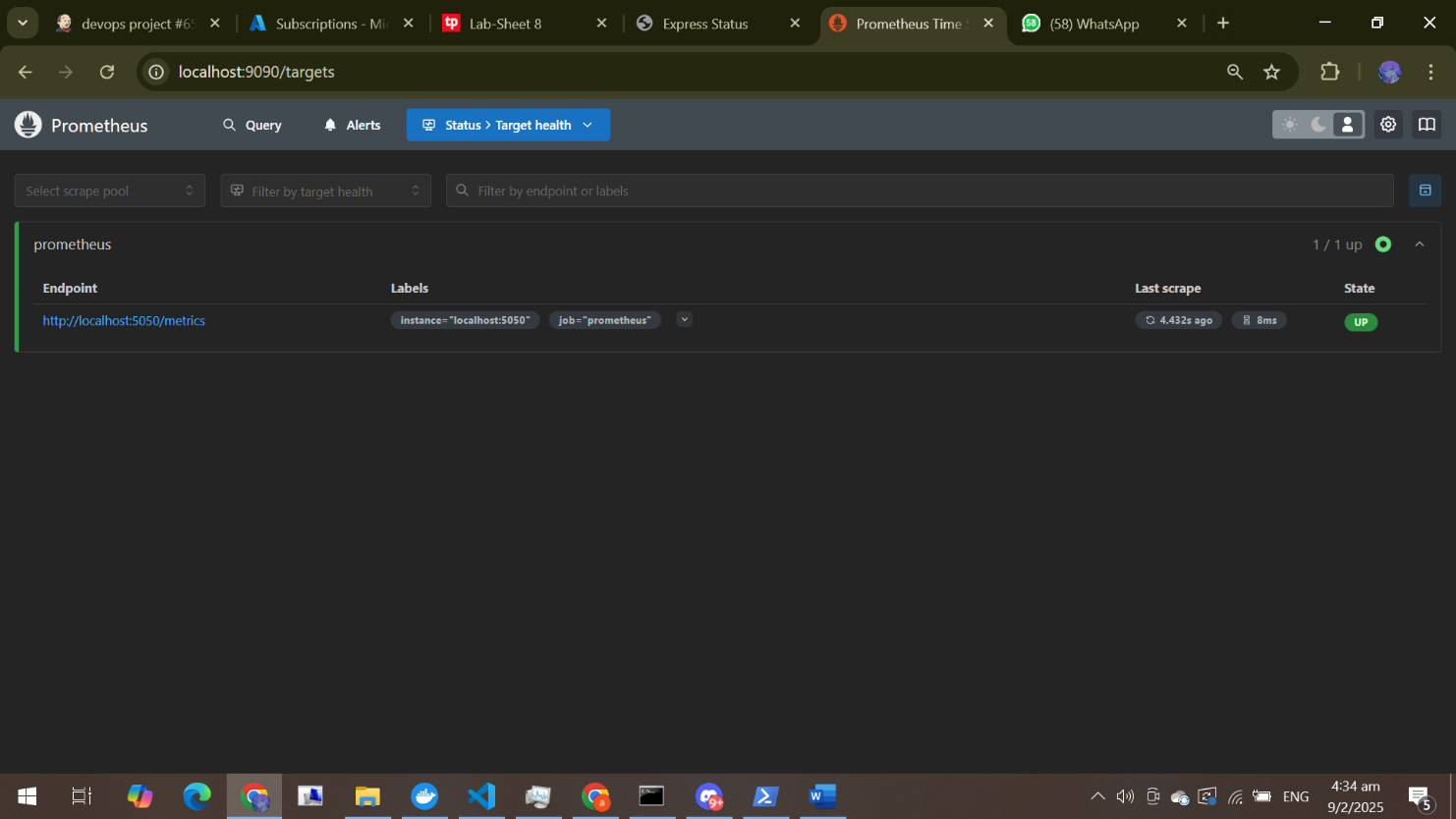
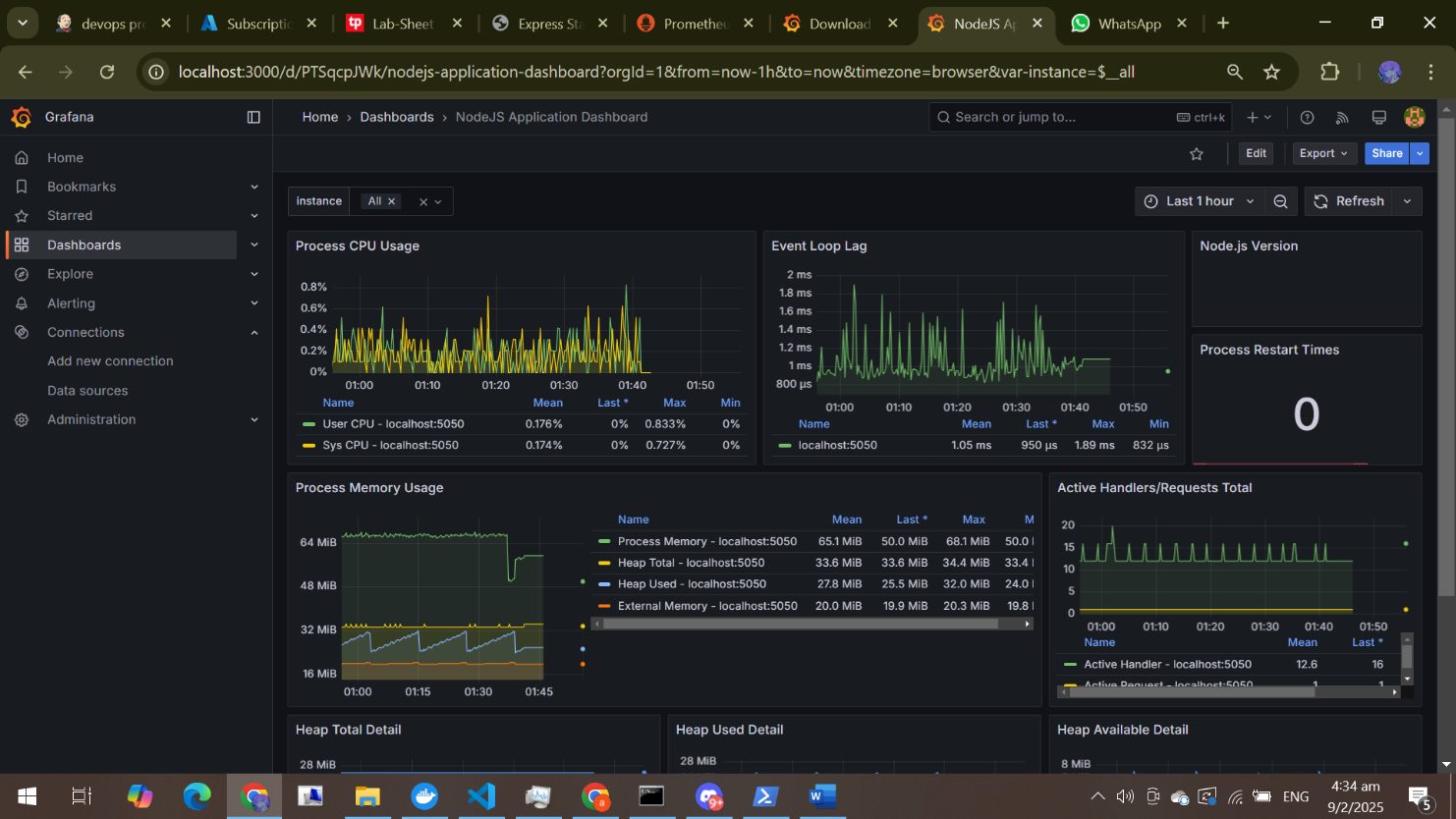
I captured screenshots of Grafana dashboards displaying real-time data from my application.

1. **References**

To implement Prometheus + Grafana, I referred to the following resources:

1. [Prometheus Official Documentation](https://prometheus.io/docs/)
2. [Grafana Official Documentation](https://grafana.com/docs/)
3. [Prometheus + Grafana Setup Guide](https://dev.to/techworld_with_nana/monitoring-with-prometheus-and-grafana-a-beginners-guide-2ehm)
4. **Conclusion**

The implementation of Prometheus + Grafana alongside PM2 + PM2 Monitoring significantly enhanced my application’s monitoring and observability. Prometheus enabled efficient metric collection, while Grafana provided an intuitive interface for data visualization, resulting in improved performance tracking and quicker debugging capabilities.



# 

# Plan-Perform-Monitor-Reflect (PPMR) Journal

** PLAN**

|  |  |
| --- | --- |
| **Monitoring Tool**: *Prometheus + Grafana* | |
| **Ideas/Issues (Topic knowledge)** | |
| What I know   * Prometheus is an open-source monitoring system used to collect and store time-series metrics. * Grafana is a visualization tool that creates dashboards using Prometheus data. * I have already set up Prometheus and Grafana on my system. * I understand how to configure Prometheus to scrape application metrics. * I know how to create basic Grafana dashboards for monitoring system performance. | What I don’t know, or  What I need to find out/work on   * *How to set up* ***alerting mechanisms*** *using Prometheus Alertmanager.* * *How to optimize Prometheus queries (****PromQL****) for better insights.* * *How to integrate* ***Grafana with external services*** *like Slack or Email for notifications.* * *How to ensure* ***scalability*** *when monitoring a large number of services.* |
| **Action (Identify resources/information)** | |
| Get help from   * Online communities such as **Reddit, Stack Overflow, and DevOps forums**. * My classmates or teachers who have experience with monitoring tools. * YouTube tutorials from DevOps experts*.* | Resources I should use   * **Prometheus Documentation**: <https://prometheus.io/docs/> * **Grafana Documentation**: <https://grafana.com/docs/> * **YouTube DevOps Channels** |
| **Time Management** | |
| By when must I complete  10/02/2025   * **Day 1-2:** Research and install **Prometheus and Grafana**. * **Day 3-4:** Configure **Prometheus to scrape application metrics**. * **Day 5-6:** Create **Grafana dashboards** to visualize metrics. | |

**A yellow bus with blue and yellow windows

Description automatically generated with medium confidence PERFORM**

|  |
| --- |
| **Strategies that I am using to** |
| Monitor progress/keep on task (is the task taking longer than expected?)   1. Using **Trello** or a **task list** to track progress. 2. Setting daily/weekly **checkpoints** to ensure steady completion. 3. Keeping notes of any errors or issues faced while implementing. |
| Locate resource/get help (where & how)   1. Search Google and Stack Overflow for troubleshooting. 2. Watch **tutorial videos** when I need a **visual demonstration**. |
| Check understanding (understand everything I need to understand)   1. Apply what I learn immediately to a test environment. 2. Confirm that **Prometheus is collecting metrics** by accessing http://localhost:9090/metrics. 3. Verify **Grafana dashboards display real-time data correctly**. |
| 1. Motivate myself   Remind myself of the importance of monitoring for system reliability.  Take **short breaks** to stay focused.  Reward myself after achieving a milestone. |
| **Resources/Information I have gathered** |
| * ***Prometheus Official Docs****:* [*https://prometheus.io/docs/*](https://prometheus.io/docs/) * ***Grafana Tutorials****:* [*https://grafana.com/tutorials/*](https://grafana.com/tutorials/) * ***YouTube Videos****: DevOps channels focusing on Prometheus and Grafana.* |

**MONITOR**

|  |
| --- |
| **Evaluating Time Management & Potential Information Sources** |
| **Tasks Monitoring**   * *Regularly check if I am meeting my milestones.* * *Keep a* ***log of difficulties and how I solved them****.* * *Adjust my timeline if tasks take longer than expected.* |
| **Resources Monitoring**   * *Ensure that sources are* ***updated and reputable****.* * *Compare information from multiple sources to verify accuracy.* * *Only follow* ***official documentation and industry-best blogs****.* |

**A blackboard with white text

Description automatically generated REFLECT**

|  |
| --- |
| **Write a reflection on your learning** |
| **Articulated Learning**   * *I have learned how to install and configure* ***Prometheus and Grafana from scratch****.* * *I have a better understanding of* ***monitoring application performance****.* * *I successfully created* ***custom Grafana dashboards*** *to visualize system metrics.* * *I improved my ability to troubleshoot* ***monitoring-related issues****.*   **Self Assessment:**   * I believe I progressed well in setting up **a fully functional monitoring system**. * I need to improve my **alerting setup** in Prometheus. * In a future project, I would **dedicate more time to optimizing queries in PromQL**. * As a **Self-Directed Learner**, I rate myself **8/10** – I managed my learning well but could improve efficiency.   **Final Thoughts:**  Installing and implementing Prometheus and Grafana has enhanced my ability to **monitor, visualize, and troubleshoot application performance effectively**. This process helped me develop a **structured approach to self-learning** and applying monitoring best practices. |
| **Evaluation**  The Plan-Perform-Monitor-Reflect (PPMR) process has helped me to challenge myself towards achieving the desired standards I set for my assignment.  Strongly Agree |