Infra Monitoring

Team Rocket

Dhawal Arora
Arin Kothari
Ritish Tejpal
Akshaj

Problem Statement

BACKGROUND

In modern cloud environments, monitoring and observability are crucial for keeping applications running smoothly. In this hackathon, your challenge is to create a basic cloud monitoring tool. It will collect, process, and display logs and metrics from a Kubernetes cluster (control plane as well as pods).

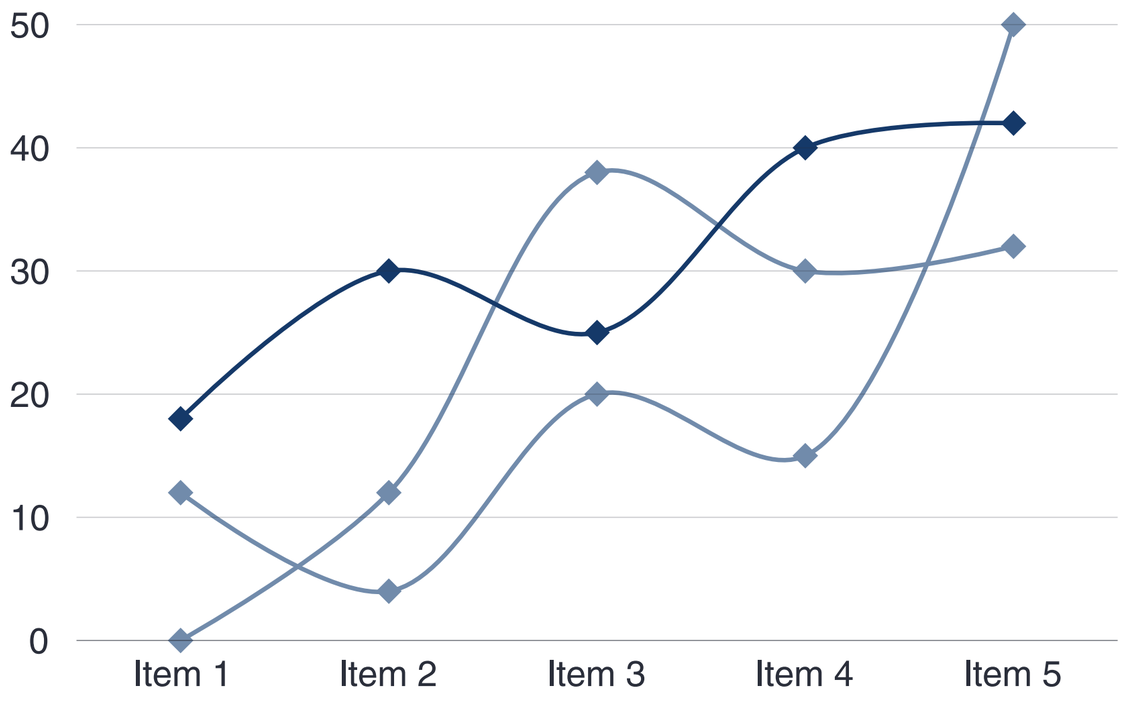
CHALLENGE

You will set up a Kubernetes cluster (using MiniKube) with two applications, a PostgreSQL and a MySQL database, running on your local machine. Your task is to build a monitoring solution that collects logs from the applications running in the Minikube cluster, authenticates them, stores them in a time series database, and visualizes them in a custom dashboard.

Logging Importance in Infrastructure

- Logging is essential for **maintaining the health and stability of any infrastructure**, especially in cloud-native or containerized environments where microservices are common.
- **Troubleshooting & Debugging**: When something goes wrong (e.g., an application crash), logs provide the necessary information to debug and trace issues back to the root cause.
- **Auditing & Compliance**: Logs are crucial for security and compliance, helping organizations track actions, detect anomalies, and maintain audit trails.

Observability vs. Monitoring:



MONITORING:

- It is the collection of predefined metrics (like CPU usage, memory consumption, and error rates) to track system health and ensure the infrastructure is working as expected.
- Monitoring tools can raise alerts when thresholds are crossed, indicating issues that need attention.

OBSERVABILITY:

- A broader concept that goes beyond monitoring. It focuses on the ability to infer the internal state of the system based on external outputs (logs, metrics, traces).
- While monitoring answers "what is happening?", observability answers "why is it happening?"

Observability Frameworks

Fluent Bit

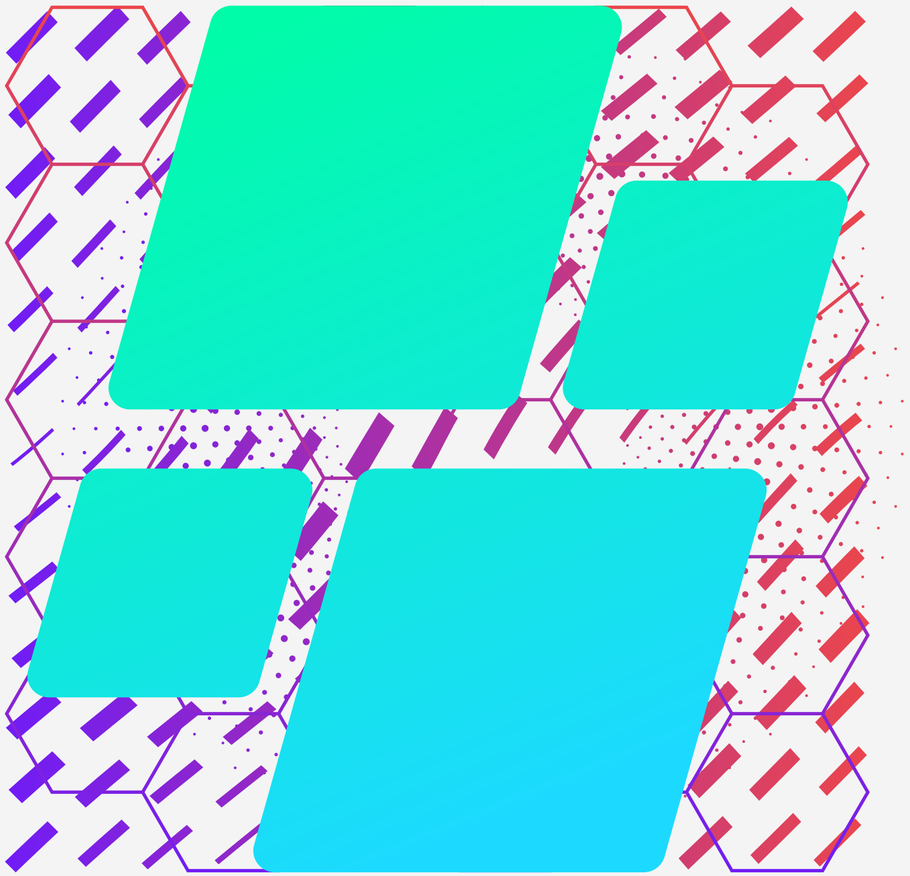
Loki

Prometheus

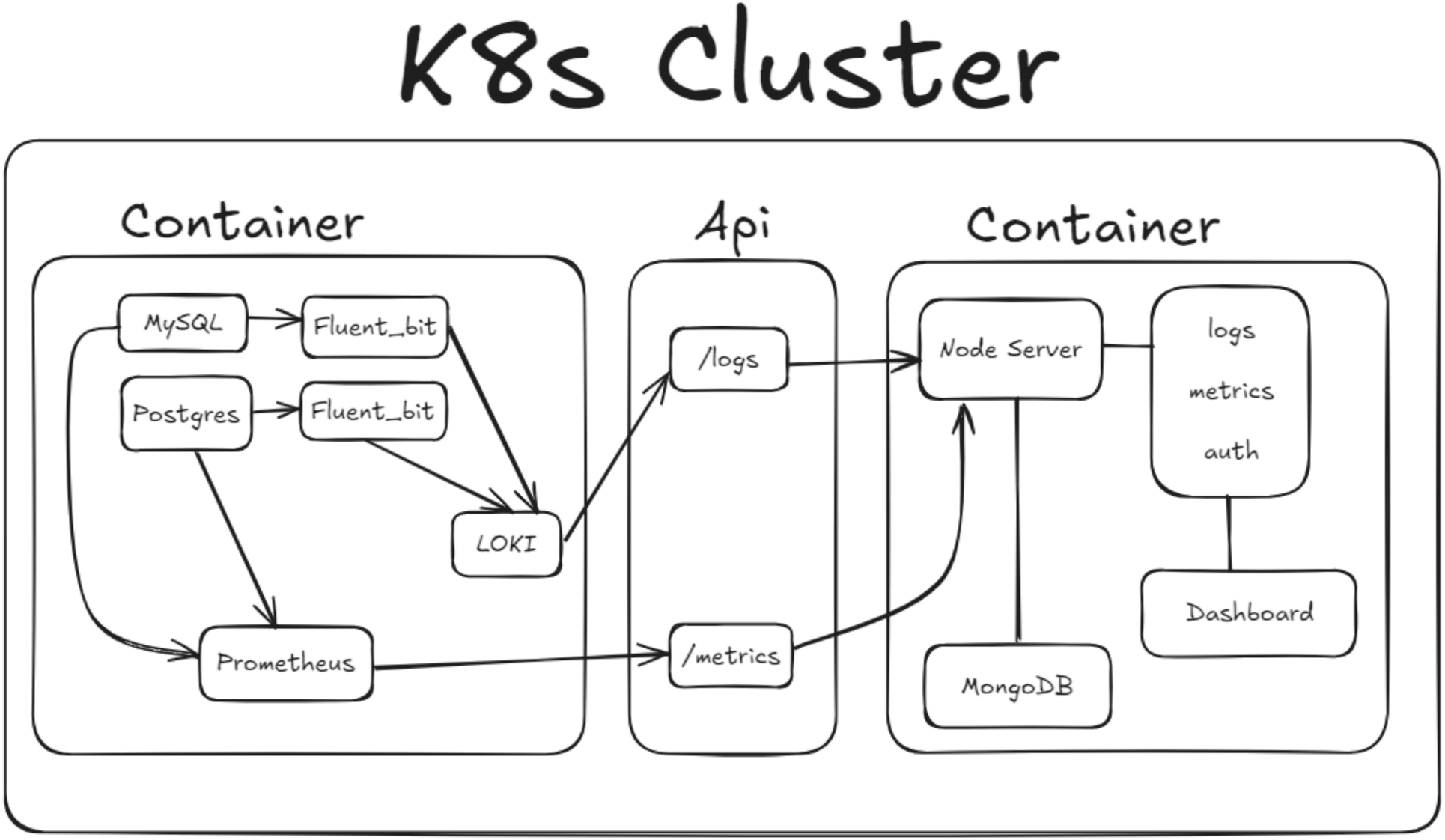
DataDog

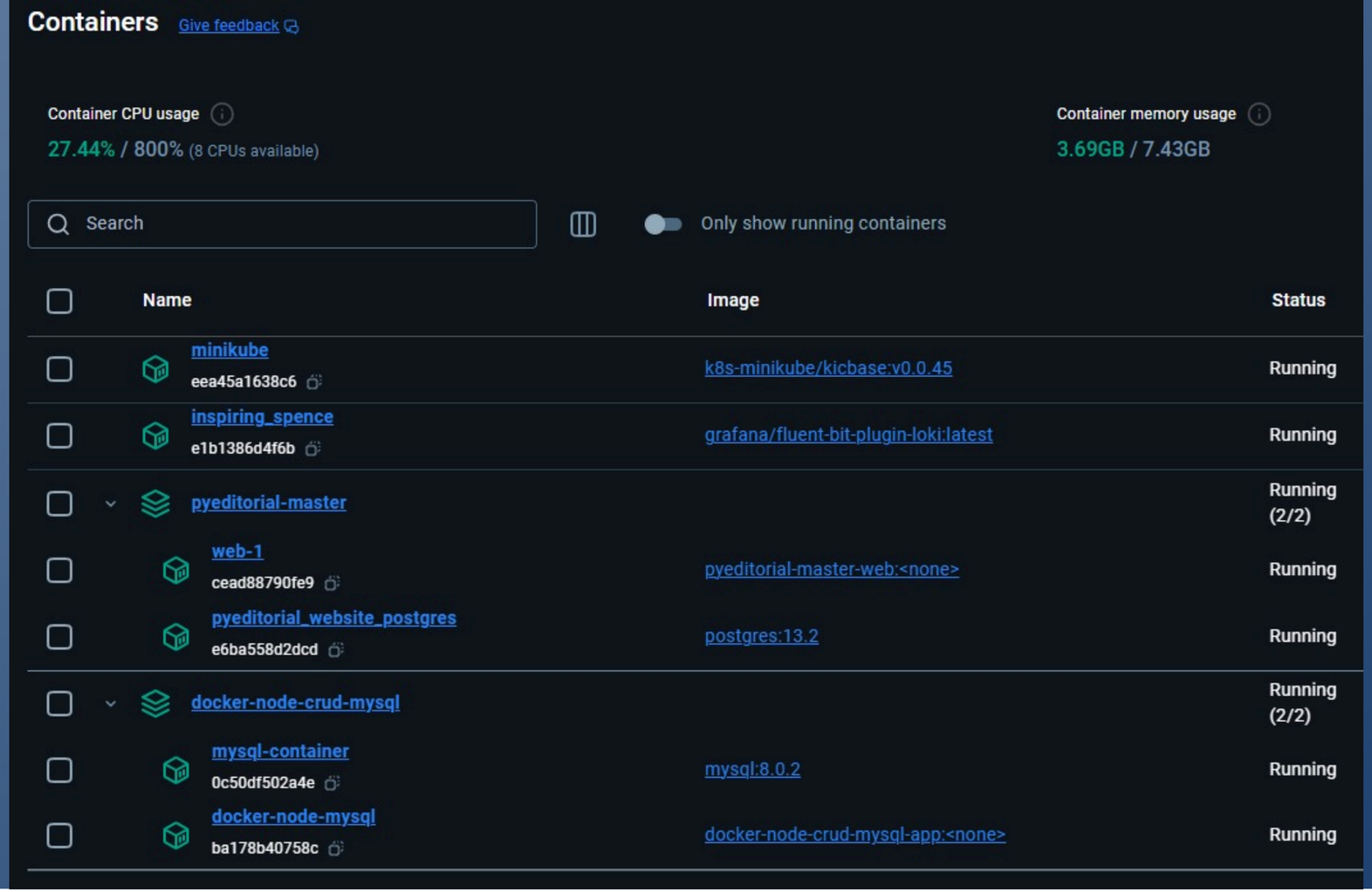
Implementation

- Used Fluent Bit for Logging
- Used Prometheus for Metric Collection
- Used Loki for Time Series data storage
- Integrated with MySQL and PostgreSQL servers running on Kubernetes Clusters
- Backend Service in Node.js and Custom Dashboard in React.js

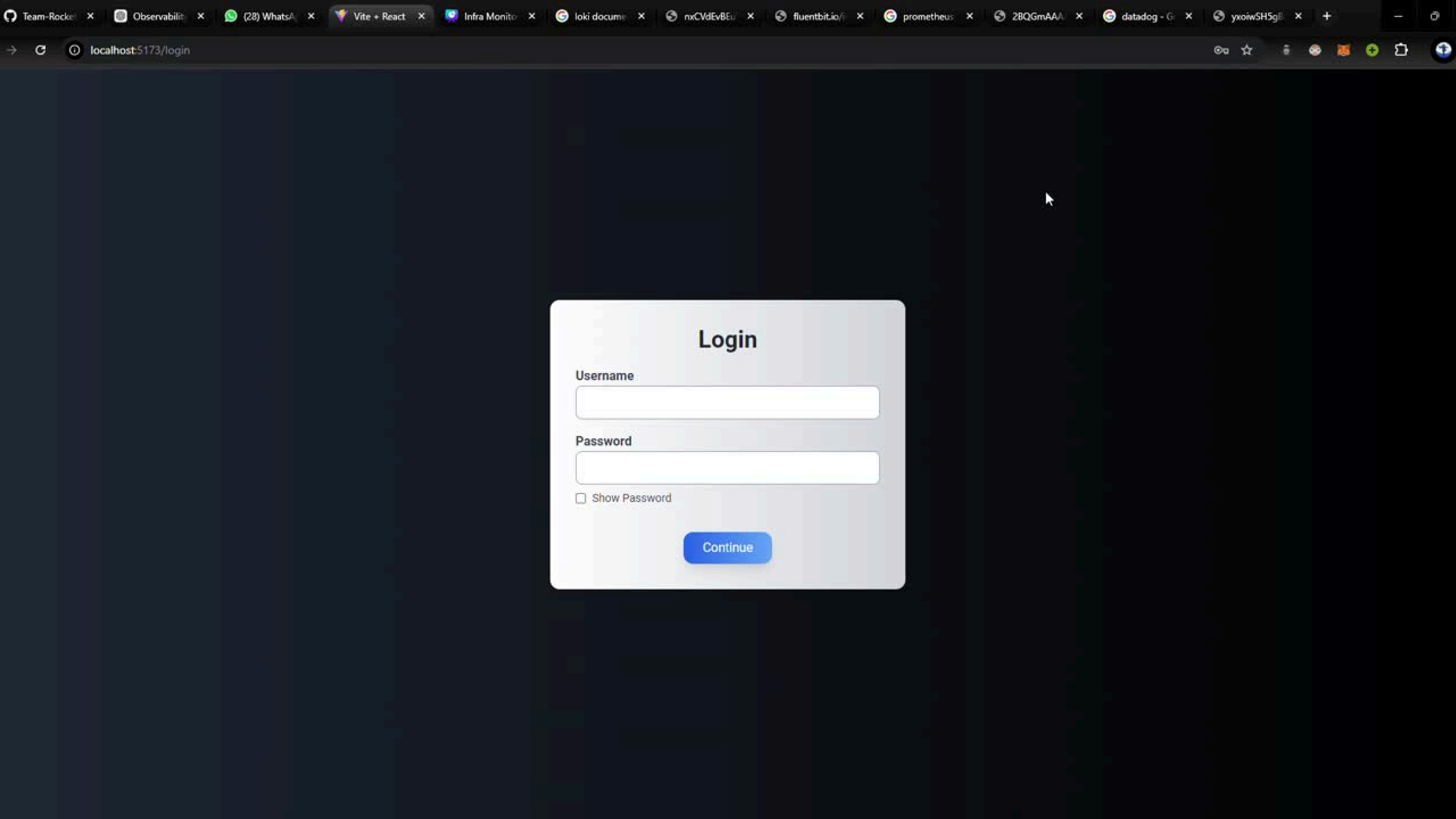


System Architecture





RUNNING ALL THE CLUSTERS ON MINIKUBE

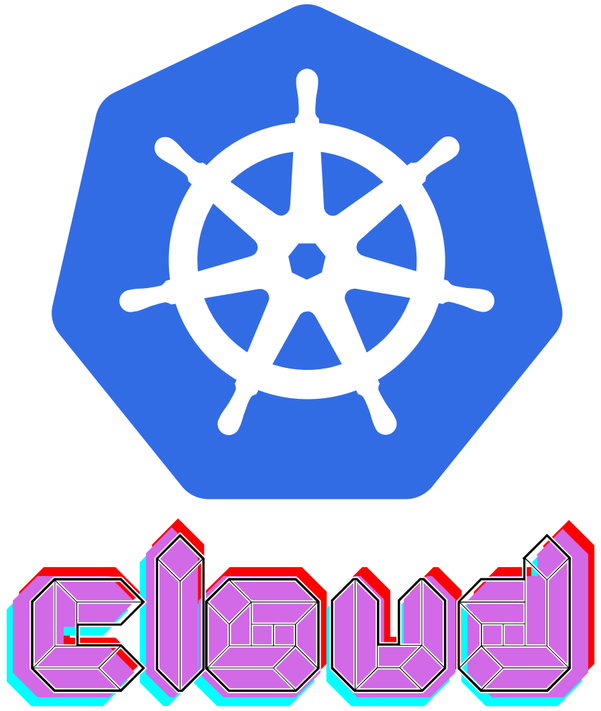

Team-Rocke
Observabilit
(28) WhatsA
Vite + React
Infra Monito
loki docum
nxCVdEvBEu
fluentbit.io/
prometheus
2BQGmAAA
datadog - G
yxoiwSH5gB
localhost:5173/login
Login
Username
Password
Show Password
Continue

Learnings

- Explored industry standard deployment tools like **Docker**, **K8s**.
- Learnings about the importance of observability and monitoring in SDLC.
- Integrating all the different technologies and understanding how they all work in tandem.
- How to configure **Fluent Bit**, **Loki**, and **Grafana** in a Kubernetes environment.
- Handling and manipulation of Time Series data.

Challenges Faced

- Configuring Kubernetes, Docker and all the different frameworks was a novel experience.
- Debugging through all of that in a short amount of time.
- Couldn’t properly configure InfluxDB so had to switch to Loki.
- Handling CrashLoopBackOff Errors: Understanding and debugging Kubernetes pod failures was essential to stabilizing the deployments.



FUTURE SCOPE

- **Scaling**: Moving the solution from Minikube to a production-grade Kubernetes cluster (like EKS or GKE) with more nodes and higher traffic.
- **Alerting**: Integrating an alerting system based on log patterns or metrics thresholds (e.g., using Prometheus).
- **Distributed Tracing**: Introducing tracing capabilities to track request flows across multiple services, adding another layer of observability.