PBL-1

Presented by Saurabh P
Section – I
23FE10CSE00264
Manipal University Jaipur
Session – 2023-27

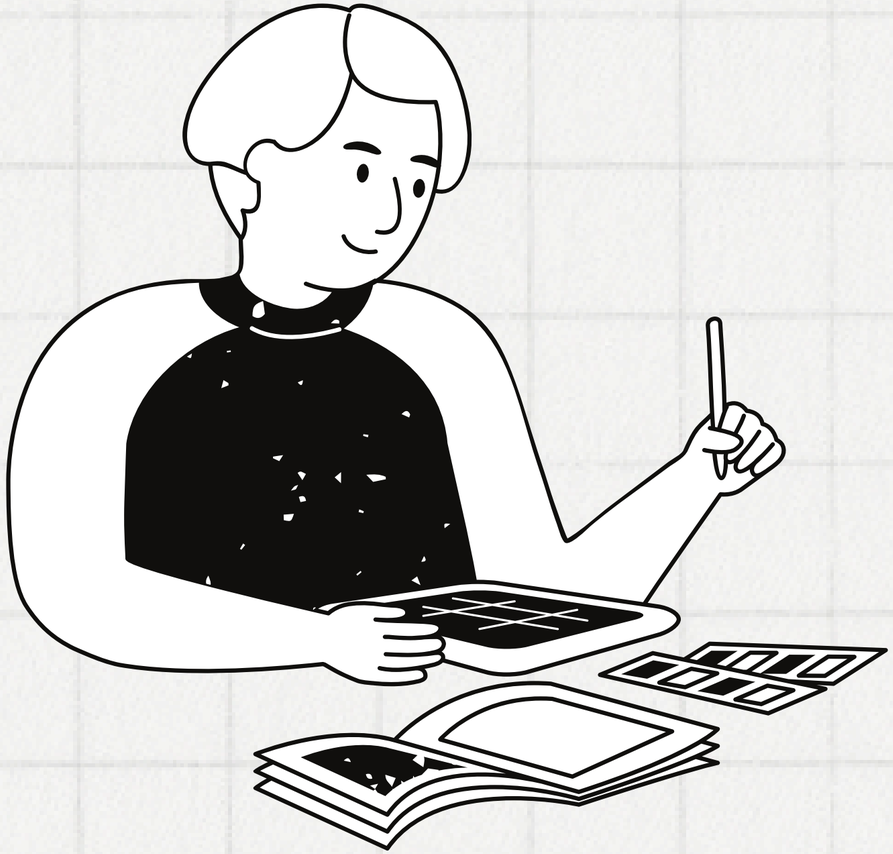
Mentor – Shikha Mundra

Table of contents:

Project Type: Software

Reasons for choosing a software based project:

- Practicality and immediate application
- Faster development cycle
- Lower costs of resources required when compared to hardware based projects



Problem Statement

Current monitoring solutions are either overly complex, lack a user-friendly interface or lack features, making it difficult to get a easy view of server performance and manage resources efficiently.

Metricly a user-friendly server monitoring tool designed to make server monitoring easy.

Literature review:

Here are some of the existing solutions: Prometheus + Grafana, Naglos and Glances

The gaps present in each solution is given below:

Prometheus + Grafana:

- **Complex setup:** Requires expertise and not easy for beginners or small scale users
- **Overhead:** Resource intensive on smaller servers and personal setup

Naglos:

- **Outdated interface:** User interface is dated compared to modern alternatives
- **Manual Configuration:** Setting up plugins for the required monitoring is challenging

Glances:

- **Basic functionality:** Glances doesn’t provide deep metrics collection and historical analysis
- **Limited Docker-specific features:** Focuses on overall system stats, not on container-specific metrics

Objectives

To develop a user friendly server system monitor that:

- Provides real-time insights into CPU, GPU, memory, and storage usage.
- Monitors Docker containers and their resource consumption on a per container basis.
- Simplified data visualization through graphs/charts

Skills And Resources Required

Skills:

- **Programming:** Python, Node.js
- **Web Development:** HTML, CSS, JavaScript (for frontend)
- **Database Management:** MongoDB/MySQL
- **Other:** Docker and system monitoring libraries (e.g., psutil, docker-api etc.)

Resources:

- **Development Environment:** IDE (VSCode), Docker
- **Documentation** and tutorials on various technologies used
- **Version Control:** git and github

Project plan

01

Research and planning:

- Research existing solutions and gather user feedback.
- Finalize gathering requirements and resources.

02

Development:

- Set up the development environment.
- Implement data collection and storage functionalities.
- Develop the frontend interface.

03

Testing:

- Conduct unit testing and user testing.
- Gather feedback and make necessary adjustments.

04

Deployment:

- Deploy the application on docker hub and provide docker-compose file..
- Provide user documentation.

05

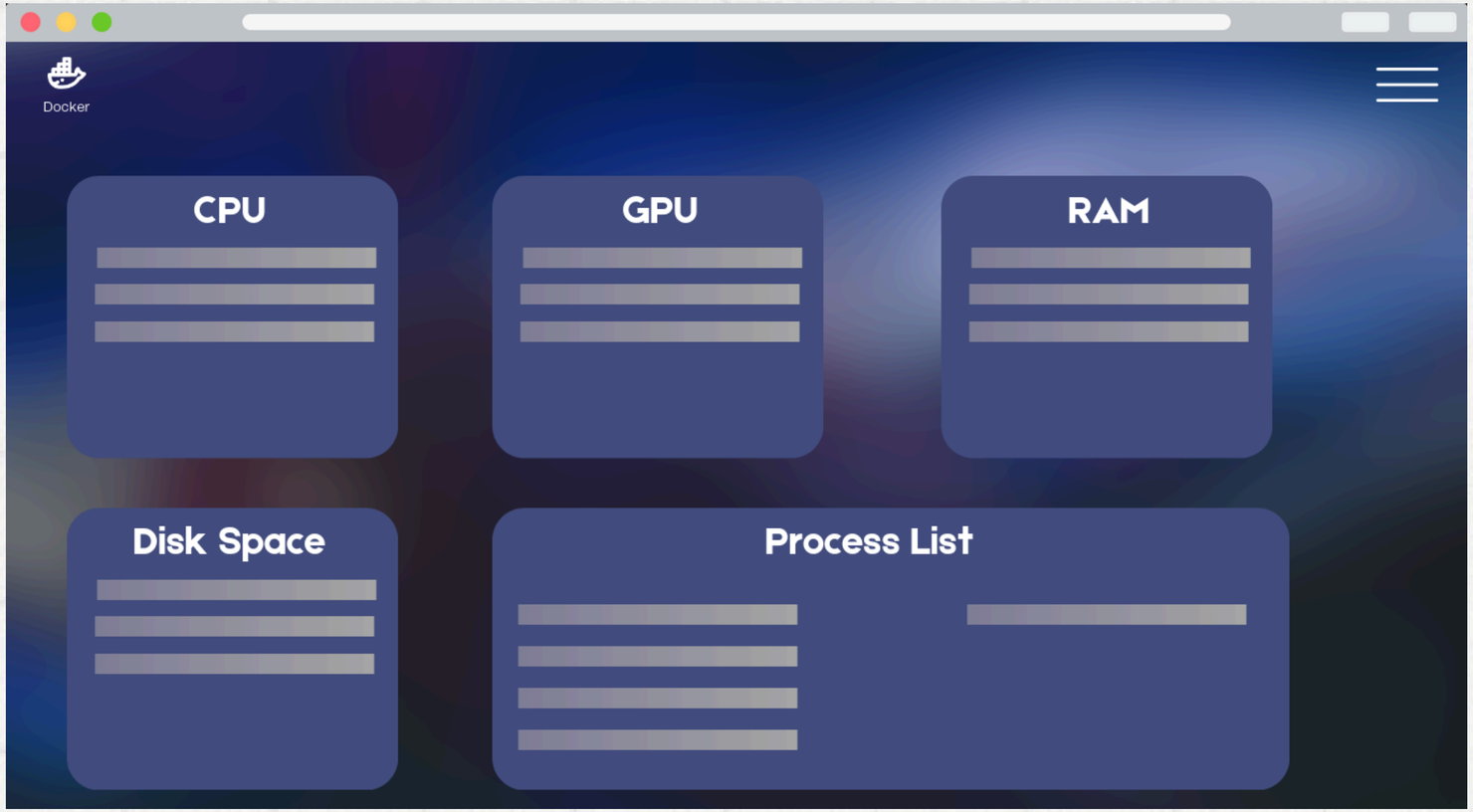
Evaluation and Feedback:

- Evaluate the project’s performance.
- Gather user feedback for future improvements.

Project Plan - Mock Website

Many more features may be present in the actual implementation such as:

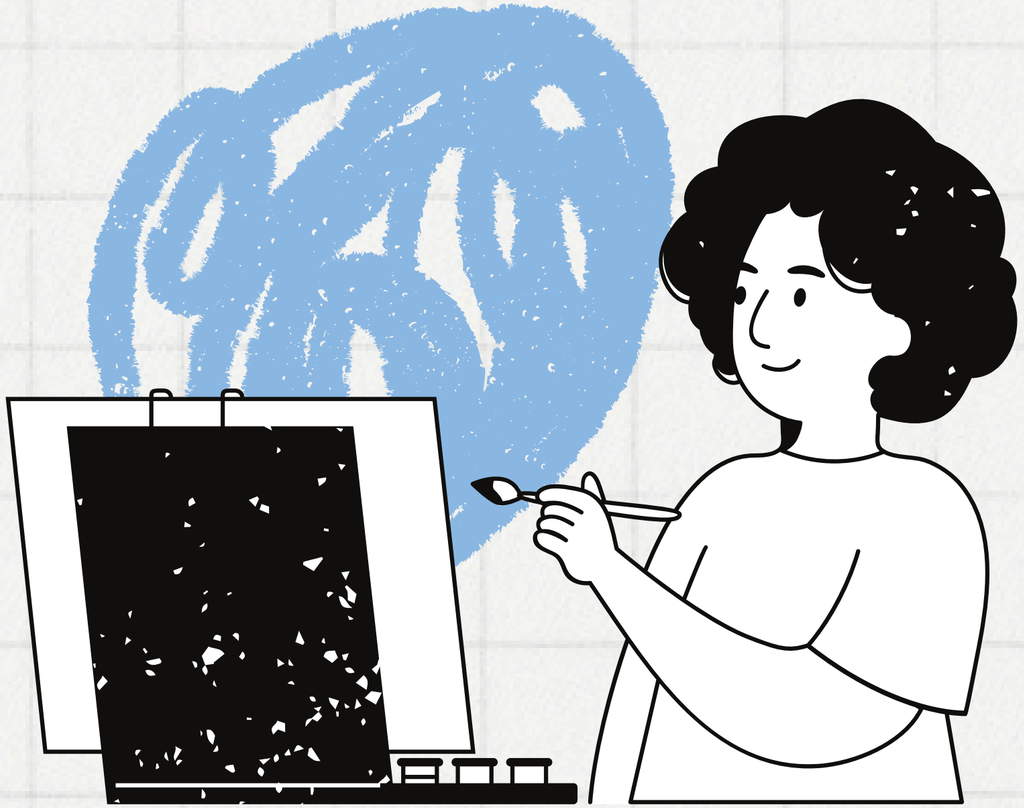
- Historical data charts
- Custom alerts and notifications
- Detailed resource pages
- Responsive design



The mock website displays server information, including CPU, GPU, memory and Docker container info offering a simple view of system health and resource management.

Future Scope:

- Implement alerting features for resource thresholds.
- Introduce historical data analysis and reporting capabilities.
- Expand the monitoring to include network performance and other metrics.
- Explore cloud-based deployment options for scalability
- Monitor the status of server backups, health checks
- Expand the system to monitor multiple servers from a single dashboard, providing a simple view of a network of servers.



Thank you!