La plateforme

CHALLENGE AAA

Michaël et Alexis NOIRET

La plateforme

SOMMAIRE

- introduction
- prerequis
- outils utilisés
- démo
- difficulté/ameliorations
- conclusion

La plateforme

INTRODUCTION

développer un outil simple de monitoring avec un dashboard web qui affiche en temps réel les statistiques d'une machine virtuelle Linux.

La plateforme

PREREQUIS

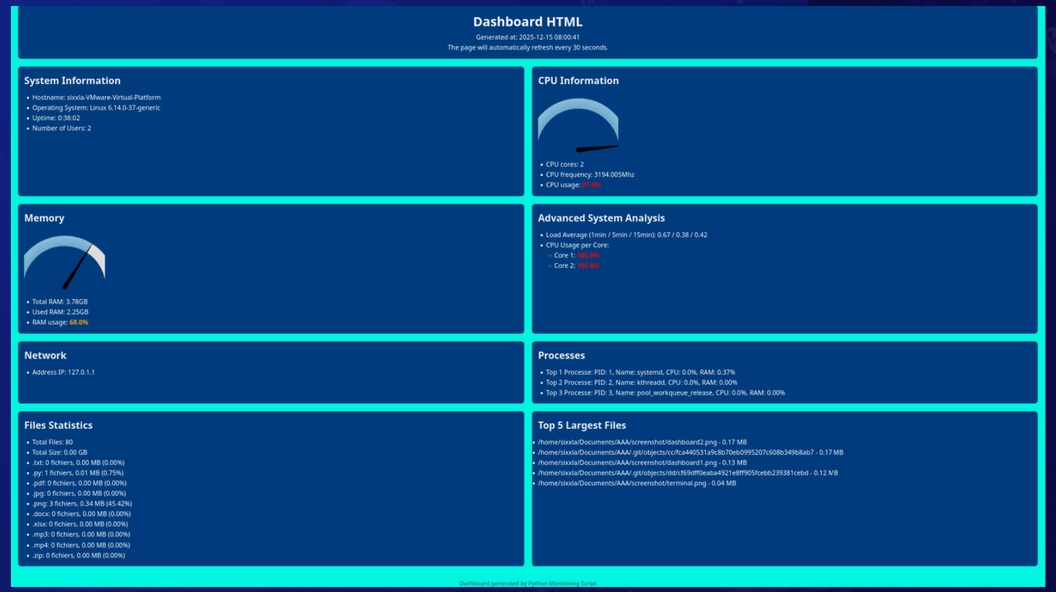
- Machine Virtuelle Ubuntu
- 2GB de RAM min
- 15GB stockage
- Accés internet
- compte admin avec droit sudo
- python et pip install

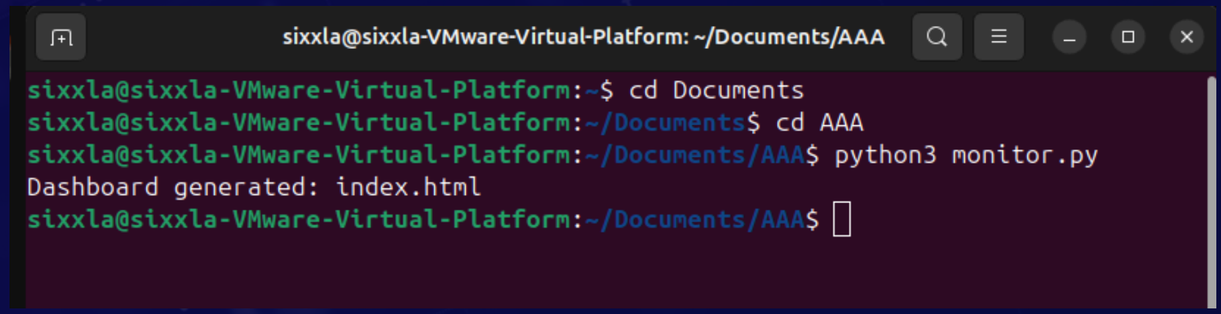
OUTILS UTILISÉS

- navigateur web
- python et psutil
- Github
- VScode
- VMware

La plateforme

DEMO





- dashboard generer
- refresh toute les 30s
- jauge JS

La plateforme

DIFFICULTÉ

nous avons rencontrer des difficulté pour les push sur github ansi que pour mettre en place les jauge gauje.js.

github

gauje.js

La plateforme

AMELIORATIONS

- Graphique historique
- alerte en cas de surcharge CPU/Memoire
- amelioration graphique ou icones

pour ameliorer le dashboard on pourrais mettre en place une alerte lorsque les CPU ou la RAM sature ainsi qu'un historique graphique mais aussi ameliorer la mise en page et le visuel avec des icones

```python
import psutil
import socket
import platform
import os
from datetime import datetime, timedelta
```

import des bibliotheques

definition des couleurs du pourcentage

```python
def get_usage_class(usage_percent):
    """Returns CSS class based on usage percentage."""
    if usage_percent <= 50:
        return "usage-vert"
    elif usage_percent <= 80:
        return "usage-orange"
    else:
        return "usage-rouge"
```

La plateforme

```python
# Load HTML template

with open("template.html", "r", encoding="utf-8") as f:
    html = f.read()
```

```python
#  Replace variables in HTML {{}}

ariables = {
    "{{ timestamp }}": timestamp,
    "{{ hostname }}": hostname,
    "{{ os_name }}": os_name,
    "{{ boot_time }}": str(boot_time),
    "{{ uptime }}": uptime,
    "{{ user_count }}": str(user_count),
    "{{ ip_address }}": ip_address,
    "{{ cpu_cores }}": str(cpu_cores),
    "{{ cpu_frequency }}": str(cpu_frequency),
    "{{ cpu_usage }}": cpu_usage_html,
    "{{ total_ram }}": str(total_ram),
    "{{ used_ram }}": str(used_ram),
    "{{ ram_usage }}": ram_usage_html,
    "{{ load_avg }}": f"{load1:.2f} / {load5:.2f} / {load15:.2f}",
    "{{ cpu_per_core }}": cpu_per_core_html,
    "{{ cpu_usage_value }}": str(cpu_usage_value),
    "{{ ram_usage_value }}": str(ram_usage_value),
    "{{ top_process_1 }}": format_process(top_process_1),
    "{{ top_process_2 }}": format_process(top_process_2),
    "{{ top_process_3 }}": format_process(top_process_3),
    "{{ text_files }}": str(file_counts[".txt"]),
    "{{ py_files }}": str(file_counts[".py"]),
    "{{ pdf_files }}": str(file_counts[".pdf"]),
    "{{ jpg_files }}": str(file_counts[".jpg"]),
    "{{ file_stats }}": file_stats_html,
    "{{ largest_files }}": largest_files_html,
    "{{ total_files }}": str(total_files),
    "{{ total_size }}": f"{total_size / (1024 ** 3):.2f} GB",
}
```

```python
# Replace variables in HTML

for key, value in variables.items():
    html = html.replace(key, value)


# Save the final HTML

output = "index.html"
with open(output, "w", encoding="utf-8") as f:
    f.write(html)

print(f"Dashboard generated: {output}")
```

generer le dashboard avec les variable charger dans le template

```
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="refresh" content="30">
    <title>system monitoring dashboard</title>
    <link rel="stylesheet" href="template.css">
    <script src="https://bernii.github.io/gauge.js/dist/gauge.min.js"></script>
```

```
<script>
    // Options de la gauge
    const CPU_USAGE = Number("{{ cpu_usage_value }}"
    const RAM_USAGE = Number("{{ ram_usage_value }}"
    var opts = {
```

JAUGE GAUGE.JS

```
// CPU Gauge
    var cpuTarget = document.getElementById('cpuGauge');
    var cpuGauge = new Gauge(cpuTarget).setOptions(opts);
    cpuGauge.maxValue = 100;
    cpuGauge.setMinValue(0);
    cpuGauge.animationSpeed = 32;
    cpuGauge.set(CPU_USAGE);

// RAM Gauge
    var ramTarget = document.getElementById('ramGauge');
    var ramGauge = new Gauge(ramTarget).setOptions(opts);
    ramGauge.maxValue = 100;
    ramGauge.setMinValue(0);
    ramGauge.animationSpeed = 32;
    ramGauge.set(RAM_USAGE);
</script>
```

CONCLUSION

Ce projet nous à permis de mieux comprendre le fonctionnement d'un outil de supervision système et de mettre en pratique plusieurs compétences techniques en même temps

La plateforme

THANK YOU