Projekt - Programowanie systemów rozproszonych

Wykonano przez: Marcin Konwiak, Michał Konwiak

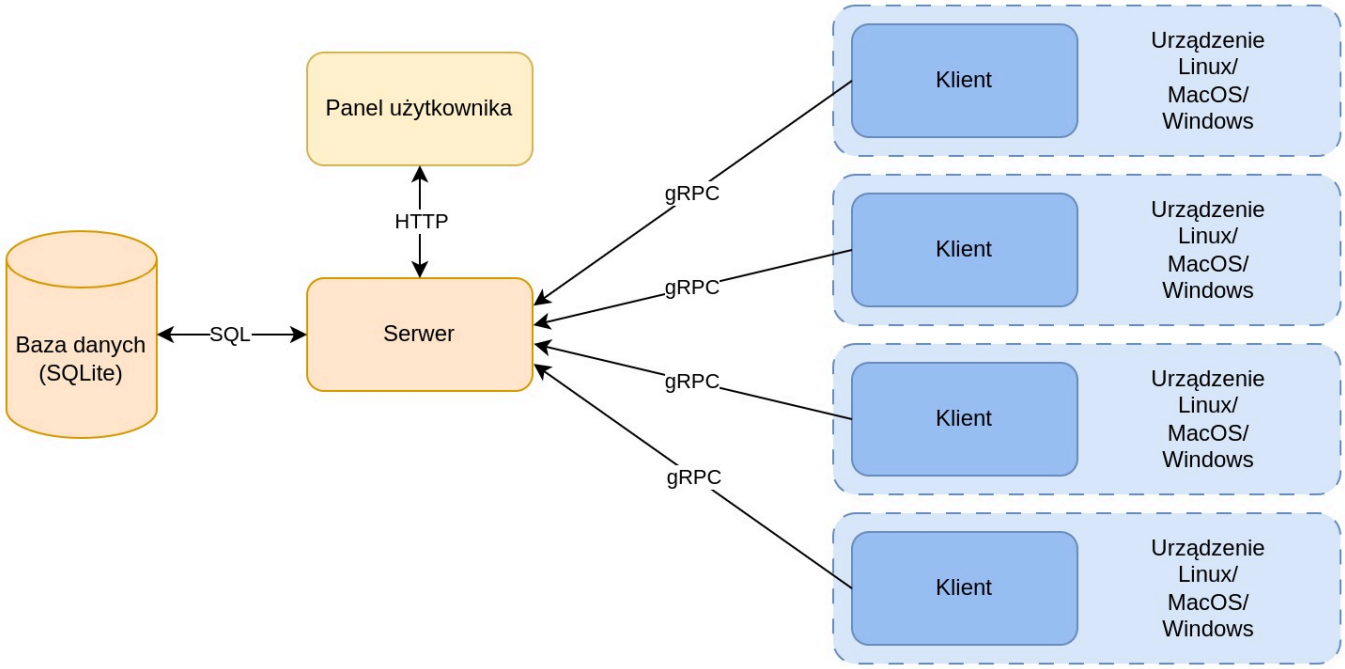
Aplikacja składa się z klienta oraz serwera. Celem serwera jest zbieranie danych zawierających informacje o systemie na którym uruchomiony jest klient. Użytkownik ma możliwość zalogowania się do aplikacji i przeglądania danych zebranych przez każdego klienta.

Przykładowy serwer dostępny jest pod http://165.227.167.195/. Dane logowania:

- Login: `projekt_psr_login`
- Hasło: `projekt_psr_haslo`

Architektura aplikacji

Na poniższym diagramie przedstawiono architekturę aplikacji.



Komunikacja między klientem a serwerem

Do komunikacji między klientem a serwerem wykorzystywany jest protokół **gRPC**. Struktura danych i metody komunikacji zostały zdefiniowane w plikach `.proto` .

Przesyłane dane

1. **Czas zebrania danych**
2. **Zużycie zasobów systemowych**:
    - Zużycie CPU (w procentach)
    - Zużycie RAM:
        - Całkowita ilość
        - Dostępna ilość
        - Użyta ilość
3. **Informacje o systemie operacyjnym**:
    - ID hosta
    - Nazwa systemu
    - Platforma
    - Wersja platformy
    - Liczba procesów
4. **Informacje o procesach**:
    - PID
    - Nazwa
    - Zużycie CPU
    - Zużycie RAM
5. **Informacje o kontenerach Docker**:
    - ID
    - Nazwa
    - Obraz

Zapis danych w bazie

Dane przesyłane przez klienta są zapisywane przez serwer w bazie danych **SQLite** za pomocą **SQL**.

Struktura zapisywanych danych

1. **Informacje o hostach** (tabela `Host`):
    - **host_id**: Unikalny identyfikator hosta (primary key).
    - **os**: Nazwa systemu operacyjnego.
    - **created_at**: Data i czas pierwszego zgłoszenia hosta.
    - **last_seen**: Data i czas ostatniego zgłoszenia hosta.
2. **Statystyki systemowe hosta** (tabela `HostBaseStats`):
    - **id**: Unikalny identyfikator wpisu (primary key).
    - **host**: Klucz obcy do tabeli `Host`.
    - **time**: Czas zebrania danych.
    - **cpu_percent**: Zużycie CPU w procentach.
    - **ram_total**: Całkowita pamięć RAM.
    - **ram_available**: Dostępna pamięć RAM.
    - **ram_used**: Użyta pamięć RAM.
    - **os**: Nazwa systemu operacyjnego.
    - **platform**: Platforma systemu operacyjnego.
    - **platform_version**: Wersja platformy.
    - **processes**: Liczba procesów działających na hoście.

3. **Informacje o procesach** (tabela `HostProcesses`):
    - **id**: Unikalny identyfikator wpisu (primary key).
    - **pid**: Identyfikator procesu (PID).
    - **name**: Nazwa procesu.
    - **cpu**: Zużycie CPU przez proces.
    - **mem**: Zużycie pamięci przez proces.
    - **host**: Klucz obcy do tabeli `Host`.
4. **Informacje o kontenerach Docker** (tabela `HostContainers`):
    - **id**: Unikalny identyfikator wpisu (primary key).
    - **name**: Nazwa kontenera.
    - **image**: Obraz kontenera.
    - **host**: Klucz obcy do tabeli `Host`.

Dostęp do danych

Dane zgromadzone w bazie danych są dostępne poprzez panel użytkownika w przeglądarce, który komunikuje się z serwerem za pomocą protokołu **HTTP**. Aby uzyskać dostęp do panelu użytkownika, należy podać login i hasło administratora.

Uruchomienie klienta

Klienta można uruchomić pobierając gotowy plik wykonywalny z sekcji **Releases** w repozytorium w serwisie GitHub lub budując aplikację z kodu źródłowego.

Do wybudowania aplikacji wymagane jest środowisko **Go** w wersji 1.23 lub nowszej.

```
go build main.go
```

Uruchomienie klienta:

```
./main start -s <adres_serwera> -i <interwał_zbierania_danych>
```

Uruchomienie serwera

Serwer można uruchomić za pomocą narzędzia **Docker Compose** lub korzystając z lokalnego środowiska python.

Uruchomienie serwera za pomocą **Docker Compose**:

```
docker-compose up
```

Uruchomienie panelu użytkownika lokalnie (wymagane jest narzędzie **uv**):

```
uv run manage.py runserver
```

Uruchomienie serwera gRPC lokalnie:

```
uv run manage.py grpcrunaioserver
```

Uruchomienie migracji bazy danych:

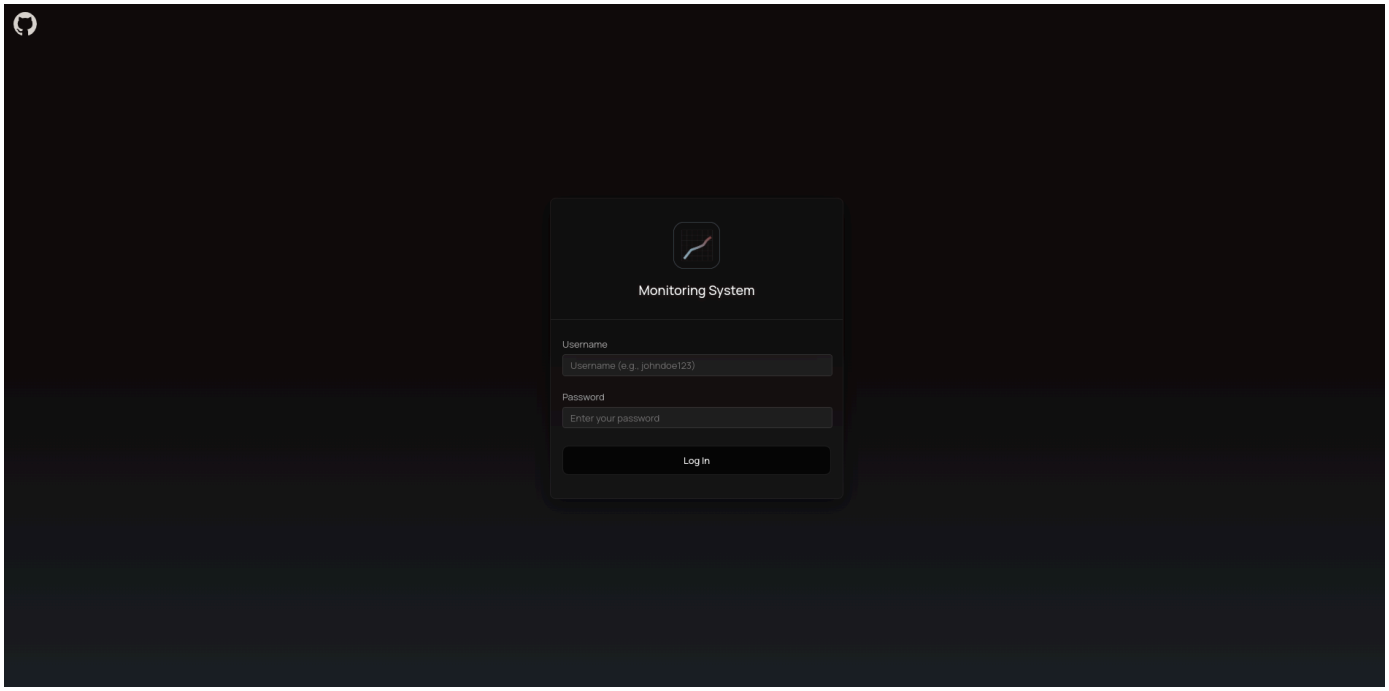
```
uv run manage.py migrate
```

Dodanie użytkownika:

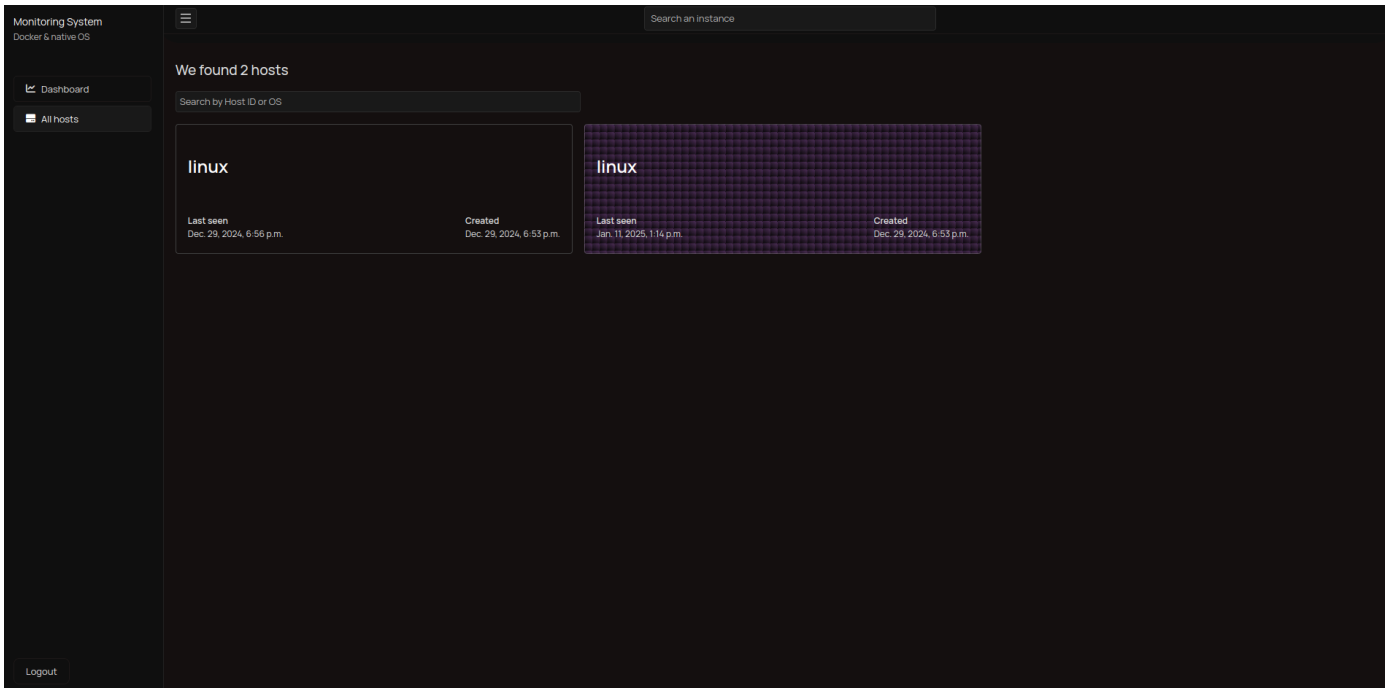
```
uv run manage.py createsuperuser
```

Panel użytkownika

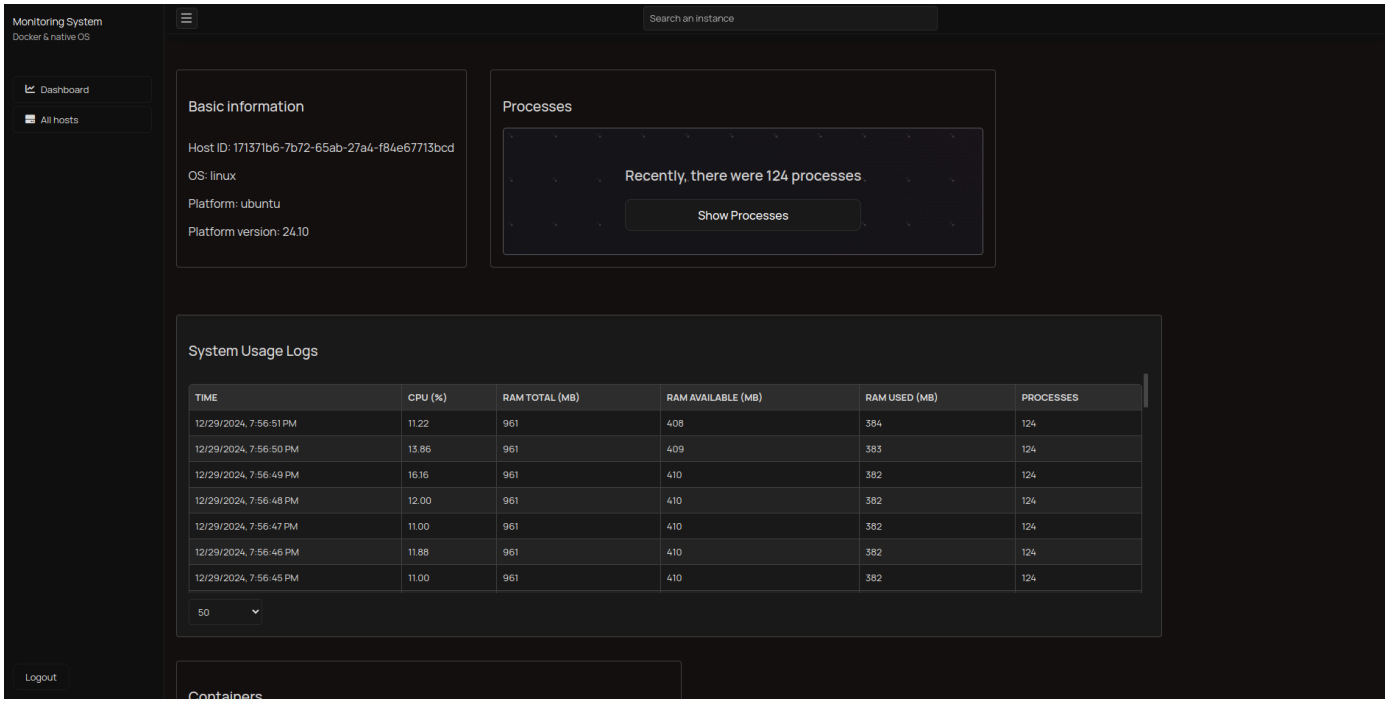
Panel logowania:

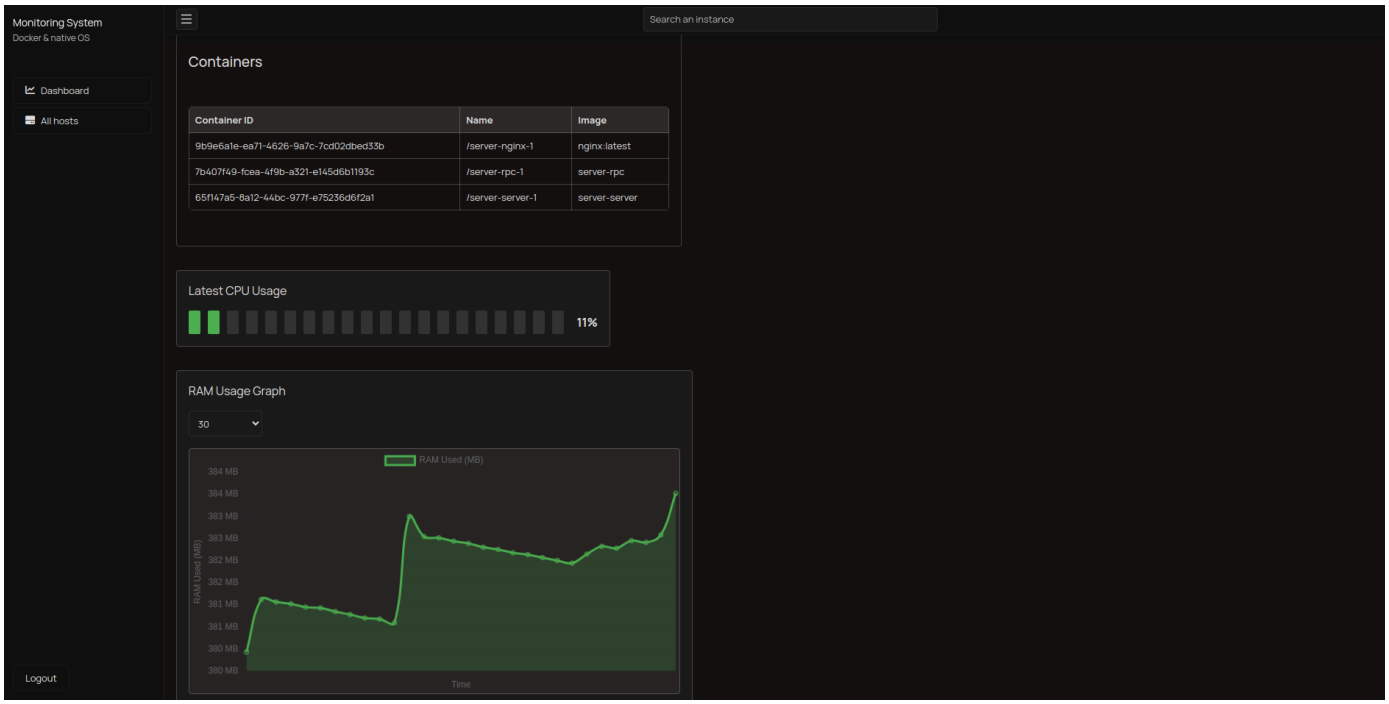


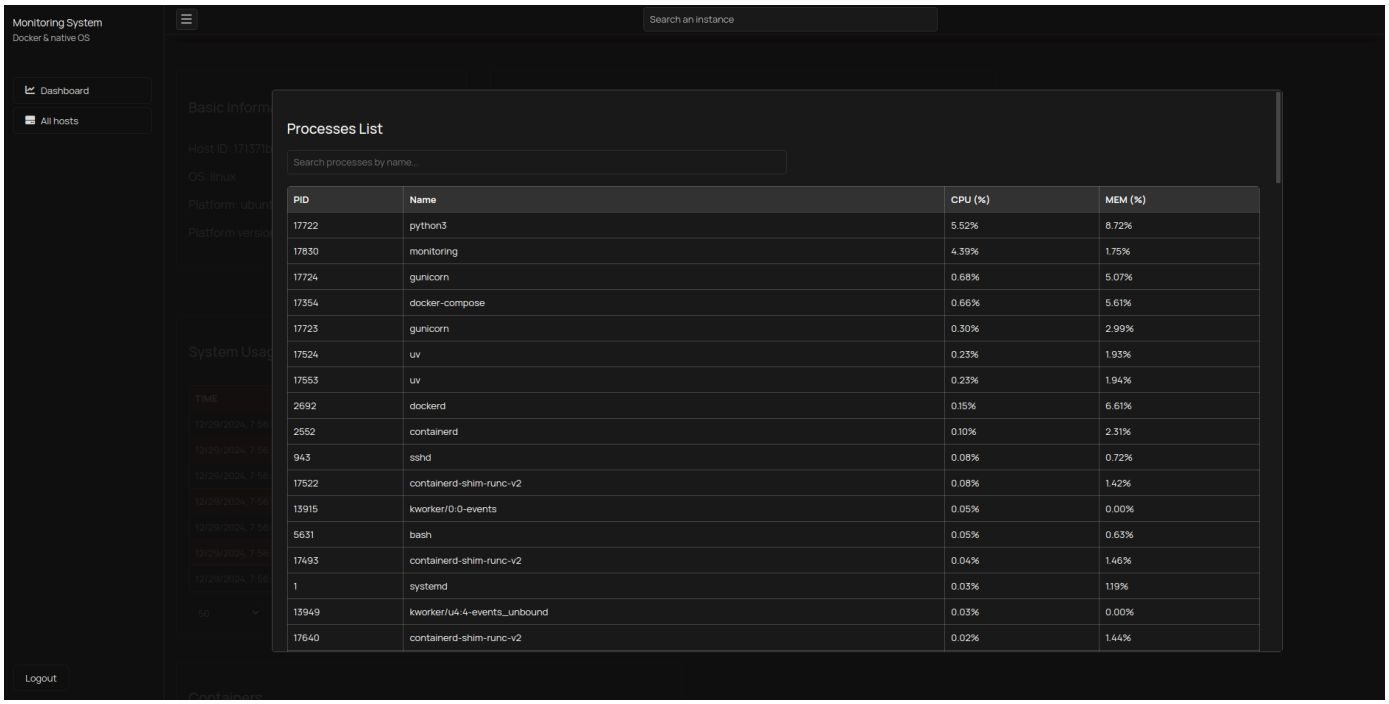
Lista klientów:



Szczegóły klienta:




Monitoring System
Docker & native OS
Dashboard
All hosts
Search an instance
Containers
Container ID
Name
Image
9b9e6a1e-ea71-4626-9a7c-7cd02dbed33b
/server-nginx-1
nginx:latest
7b407f49-fcea-4f9b-a321-e145d6b1193c
/server-rpc-1
server-rpc
65f147a5-8a12-44bc-977f-e75236d6f2a1
/server-server-1
server-server
Latest CPU Usage
11%
RAM Usage Graph
30
RAM Used (MB)
384 MB
384 MB
383 MB
383 MB
382 MB
382 MB
381 MB
381 MB
380 MB
380 MB
RAM Used (MB)
Time
Logout


Monitoring System
Docker & native OS
Dashboard
All hosts
Search an instance
Processes List
Search processes by name...
PID
Name
CPU (%)
MEM (%)
17722
python3
5.52%
8.72%
17830
monitoring
4.39%
1.75%
17724
gunicorn
0.68%
5.07%
17354
docker-compose
0.66%
5.61%
17723
gunicorn
0.30%
2.99%
17524
uv
0.23%
1.93%
17553
uv
0.23%
1.94%
2692
dockerd
0.15%
6.61%
2552
containerd
0.10%
2.31%
943
sshd
0.08%
0.72%
17522
containerd-shim-runc-v2
0.08%
1.42%
13915
kworker/0:0-events
0.05%
0.00%
5631
bash
0.05%
0.63%
17493
containerd-shim-runc-v2
0.04%
1.46%
1
systemd
0.03%
1.19%
13949
kworker/u4:4-events_unbound
0.03%
0.00%
17640
containerd-shim-runc-v2
0.02%
1.44%
Logout