

## Contents

Instrukcja Uruchomienia Serwera — System Primus 2026 .....	1
Konfiguracja Srodowiska .....	1
1. Tryb Produkcyjny (Zalecany) .....	1
2. Tryb Deweloperski (Lokalne Budowanie) .....	1
Struktura Projektu .....	2
Konfiguracja NVIDIA (GPU) .....	2
Konfiguracja AMD (GPU) .....	3
Tryb Hybrydowy (WSL2) .....	3
Tryb Natywny (Linux ROCm) .....	3
Konfiguracja z zewnętrznym Ollama/OpenAI (CPU) .....	3
Certyfikaty SSL .....	3
Własne Certyfikaty .....	3
Dokumentacja Projektu .....	3

## Instrukcja Uruchomienia Serwera — System Primus 2026

To repozytorium odpowiada za orkiestrację usług, zarządzanie siecią i bezpieczeństwo danych (TLS/Szyfrowanie).

### Konfiguracja Srodowiska

#### 1. Tryb Produkcyjny (Zalecany)

W tym trybie pobierasz tylko repozytorium infrastruktury i uruchamiasz gotowe obrazy z Docker Hub.

```
git clone https://github.com/Primus2026/primus-infra.git
cd primus-infra
cp .env.example .env
# Edytuj .env i ustaw odpowiednie wartości
docker compose up -d
```

#### 2. Tryb Deweloperski (Lokalne Budowanie)

Wymaga pobrania wszystkich repozytoriów, aby zbudować obrazy z kodu źródłowego.

### Struktura Katalogów

Utwórz katalog główny i wejdź do niego:

```
mkdir Primus2026
cd Primus2026
```

## Klonowanie Repozytoriów

Wykonaj poniższe polecenia, aby pobrać wszystkie moduły systemu:

```
# Infrastruktura (Docker Compose)
git clone https://github.com/Primus2026/primus-infra.git

# Pozostałe moduły
git clone https://github.com/Primus2026/primus-backend.git
git clone https://github.com/Primus2026/primus-web-frontend.git
git clone https://github.com/Primus2026/primus-mobile.git
git clone https://github.com/Primus2026/primus-docs.git
git clone https://github.com/Primus2026/primus-mqtt-listener.git
git clone https://github.com/Primus2026/primus-mock-sensor.git
```

## Uruchomienie

```
cd primus-infra/dev
cp .env.example .env
# Edytuj .env i ustaw odpowiednie wartości
docker compose up --build -d
```

Flaga `--build` wymusi zbudowanie obrazów z kodu źródłowego (`../primus-backend` itd.).

## Struktura Projektu

- `dev/` [DOMYŚLNE]: Zawiera konfigurację deweloperską (buduje obrazy lokalnie z `context: .`).
- `docker-compose.yml`: Konfiguracja produkcyjna (używa gotowych obrazów z Docker Hub).
- `docker-compose-nvidia.yml`: Dodatek produkcyjny dla NVIDIA GPU.
- `docker-compose-amd.yml`: Dodatek produkcyjny dla AMD GPU.

## Konfiguracja NVIDIA (GPU)

Zalecana dla systemów z kartami graficznymi NVIDIA.

### Instalacja NVIDIA Container Toolkit:

```
curl -fsSL https://nvidia.github.io/libnvidia-container/gpgkey | sudo gpg --
dearmor -o /usr/share/keyrings/nvidia-container-toolkit-keyring.gpg \
&& curl -s -L https://nvidia.github.io/libnvidia-container/stable/deb/nvidia-
container-toolkit.list | \
    sed 's#deb https://#deb [signed-by=/usr/share/keyrings/nvidia-container-
toolkit-keyring.gpg] https://#g' | \
    sudo tee /etc/apt/sources.list.d/nvidia-container-toolkit.list

sudo apt-get update && sudo apt-get install -y nvidia-container-toolkit
sudo nvidia-ctl runtime configure --runtime=docker
sudo systemctl restart docker
```

### Uruchomienie:

```
docker compose -f docker-compose-nvidia.yml up -d
```

## Konfiguracja AMD (GPU)

### Tryb Hybrydowy (WSL2)

1. **Ollama Windows:** `$env:OLLAMA_VULKAN="1"; ollama serve`
2. **Konfiguracja .env:** `OLLAMA_URL=http://host.docker.internal:11434/api/generate`
3. **Uruchomienie:** `docker compose up --build -d`

### Tryb Natywny (Linux ROCm)

```
docker compose -f docker-compose-amd.yml up -d
```

## Konfiguracja z zewnętrznym Ollama/OpenAI (CPU)

Zalecana dla systemów bez dedykowanej karty graficznej, z niekompatybilną kartą graficzną lub do szybkich testów.

### Uruchomienie:

```
docker compose up -d
```

[!IMPORTANT] W pliku `.env` należy ustawić wszystkie wymagane zmienne, w tym adresy serwerów i ewentualne klucze API.

## Certyfikaty SSL

System generuje certyfikaty self-signed przy pierwszym uruchomieniu (`scripts/generate_certs.sh`).

### Własne Certyfikaty

Podmień pliki w `nginx/certs/`: `nginx.crt`, `nginx.key`, `rootCA.pem`. Następnie: `docker compose restart nginx redis mosquitto`

## Dokumentacja Projektu

Pełna dokumentacja architektury, modelu danych i modułów znajduje się w osobnym repozytorium: **primus-docs**