

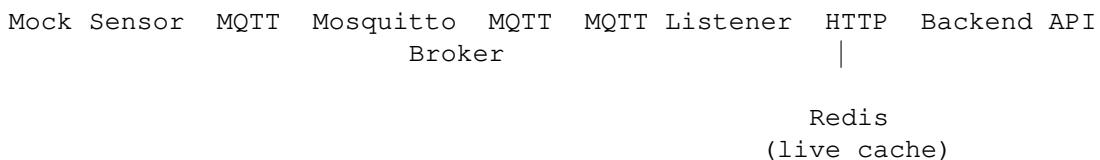
## Primus MQTT Listener

Mikroserwis nasłuchujący dane z czujników IoT dla systemu **Primus Inter Pares 2026**. Odpowiada za przetwarzanie odczytów temperatury i wagi w czasie rzeczywistym oraz generowanie alertów.

**Pełna dokumentacja projektu:** [Primus Docs](#)

### Architektura

Asynchroniczny serwis Python działający jako pomost między warstwą IoT a backendem.



### Realizacja Wymagań Specyfikacji

#### 5c. Monitoring Wagi (Nieautoryzowane Pobranie)

- **Implementacja:** [listener.py](#)
- **Logika:**
  1. Pobiera ostatnią prawidłową wagę z Redis (`Weight:{designation}:{row}:{col}`).
  2. Porównuje z aktualnym odczytem z sensora.
  3. Jeśli różnica > TOLERANCE (0.5 kg) i nie ma aktywnej transakcji (`Expected-Change:...`), generuje alert WEIGHT.

#### 5a/5b. Monitoring Temperatury

- **Logika:**
  1. Pobiera definicje regałów z backendu (zakresy `temp_min`, `temp_max`).
  2. Dla każdego odczytu sprawdza, czy temperatura mieści się w zakresie.
  3. Przy przekroczeniu generuje alert TEMP.

### Funkcjonalności

Funkcja	Opis
<b>MQTT Subscription</b>	Nasłuchuje na topiku <code>sensors/+ (wszystkie regały)</code>
<b>Redis Cache</b>	Zapisuje aktualne odczyty ( <code>rack:{id}:live</code> ) z TTL 60s
<b>Alert Throttling</b>	Nie wysyła tego samego alertu częściej niż raz na 60s
<b>Auto-Login</b>	Automatyczna autoryzacja do backendu z odświeżaniem tokena
<b>Weight Sync</b>	Przy starcie synchronizuje expected weights z backendu

## **Stos Technologiczny**

Kategoria	Technologia
<b>Język</b>	Python 3.11
<b>MQTT</b>	Paho MQTT Client
<b>Cache</b>	Redis
<b>HTTP</b>	Requests
<b>TLS</b>	Obsługa certyfikatów dla MQTT i API

## **Uruchomienie**

### **Docker (Zalecane)**

Serwis uruchamiany automatycznie jako część stosu [primus-infra](#):

### **Lokalnie**

Wymagane zmienne środowiskowe: