Od automatyzacji wdrażania do utrzymania: Jenkins, logi, monitoring i skalowanie aplikacji

Wojciech Kowalewski

wkowalewski@egnyte.com

Gdzie jesteśmy z naszym biznesem?

Gdzie jesteśmy z naszym biznesem?

- Aplikacja została rozszerzona o nowe funkcjonalności
- Aplikacja została podzielona na serwisy
- Aplikacja działa na produkcji i klienci z niej korzystają
- Deweloperzy pracują nad dalszymi funkcjonalnościami

Gdzie jesteśmy z naszym biznesem?

- Aplikacja została rozszerzona o nowe funkcjonalności
- Aplikacja została podzielona na serwisy
- Aplikacja działa na produkcji i klienci z niej korzystają
- Deweloperzy pracują nad dalszymi funkcjonalnościami
- **Czy wiemy, że aplikacja aktualnie działa?**
- **Czy wszystkie funkcje aplikacji działają poprawnie?**
- **Czy wiemy, że użytkownicy z niej korzystają?**

Agenda

- Co to jest CI/CD
- Wprowadzenie do narzędzia Jenkins
- Wprowadzenie do obserwowalności (ang. observability)
- Narzędzia ELK
- Metryki i alerty
- Narzędzia Prometheus i Grafana
- Automatyczne skalowanie aplikacji

Budowanie/uruchamianie aplikacji i praca z kodem

- W jaki sposób kompilujecie kod źródłowy i uruchamiacie aplikacje?

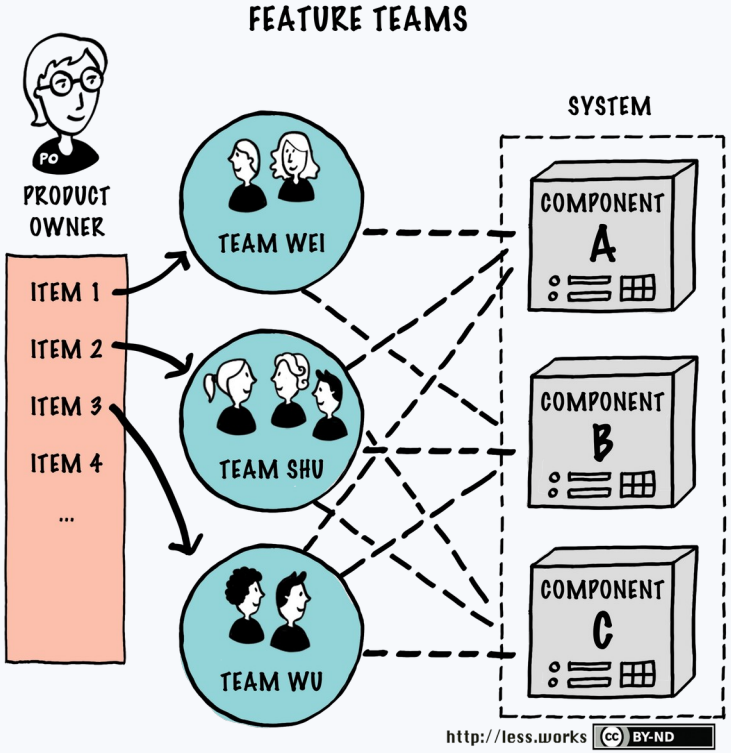
Budowanie/uruchamianie aplikacji i praca z kodem

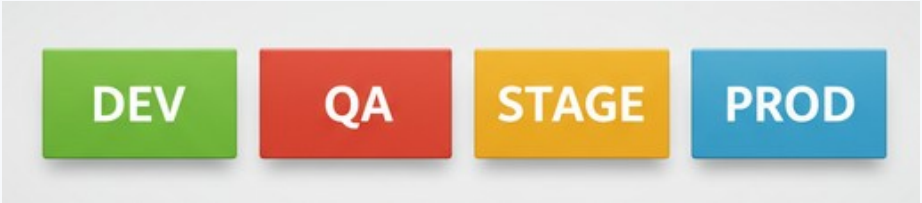
- W jaki sposób kompilujecie kod źródłowy i uruchamiacie aplikacje?
- Czy pracujecie z jednym repozytorium kodu w kilka osób?

Budowanie/uruchamianie aplikacji i praca z kodem

- W jaki sposób kompilujecie kod źródłowy i uruchamiacie aplikacje?
- Czy pracujecie z jednym repozytorium kodu w kilka osób?
- Jak upewnić się, że wprowadzona zmiana w kodzie nie wprowadziła regresji i kod dalej się kompiluje?

Inżynieria oprogramowania w firmie produktowej





CI/CD

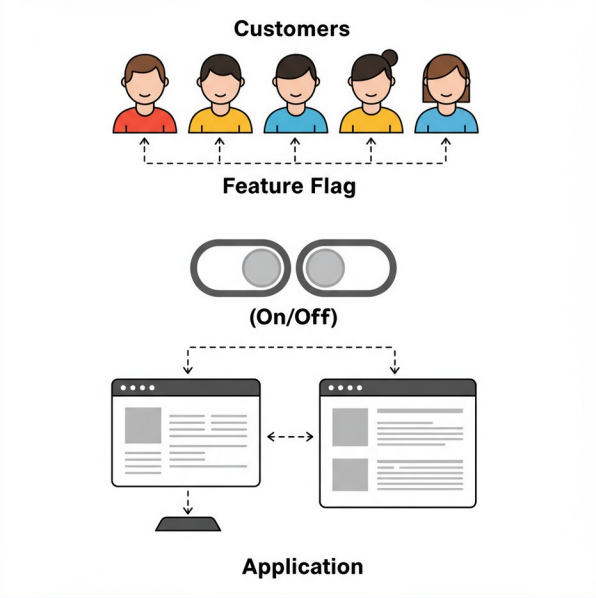
- **CI - ciągła integracja** (ang. Continuous Integration)
  - Proces automatycznej, częstej integracji kodu z głównym repozytorium oraz jego weryfikacji

CI/CD

- **CI - ciągła integracja** (ang. Continuous Integration)
  - Proces automatycznej, częstej integracji kodu z głównym repozytorium oraz jego weryfikacji
- **CD - ciągłe dostarczanie** (ang. Continuous Delivery)
  - Ciągłe dostarczanie to zautomatyzowany proces dostarczania oprogramowania do konkretnego środowiska w infrastrukturze IT

Jak deweloperzy mogą usprawić proces integracji?

- Rozbijanie dużych zdań na małe kawałki
- Częste tworzenie merge requestów - Short-Living Branches
- Feature flagi



Jenkins

- Serwer automatyzacji służący do budowania, testowania i wdrażania aplikacji

Jenkins

- Serwer automatyzacji służący do budowania, testowania i wdrażania aplikacji
- Automatyzacja poprzez tworzenie zadań budowania (ang. builds)

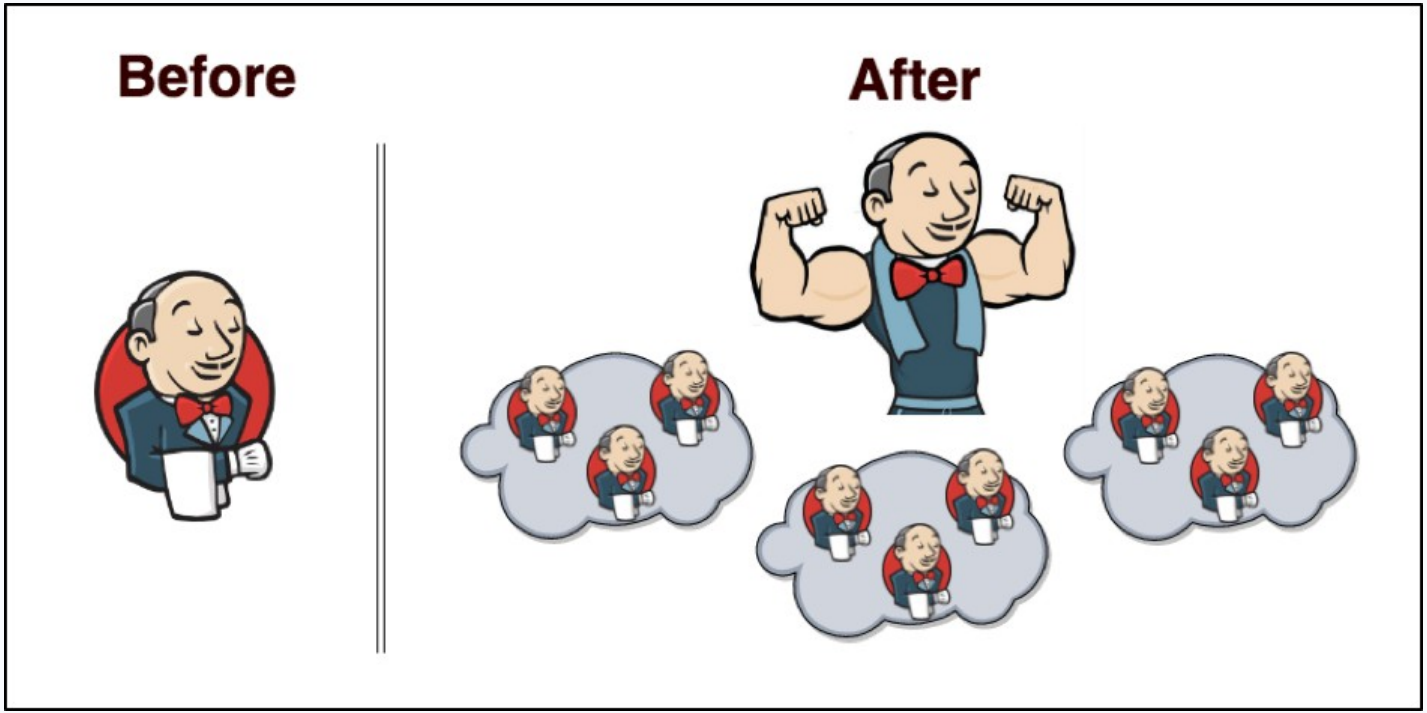
Jenkins

- Serwer automatyzacji służący do budowania, testowania i wdrażania aplikacji
- Automatyzacja poprzez tworzenie zadań budowania (ang. builds)
- Konfiguracja potoków (ang. pipelines) zapisana w repozytorium kodu projektu (Jenkinsfile)
  - Korzysta z własnego języka (DSL) opartego o język Groovy

Jenkins - demo

- Repozytorium:
  - **https://github.com/kszalast/todolist.git**
- Branch:
  - **observability_1_jenkins**

Jenkins - skalowanie



Źródło: https://dzone.com/articles/how-to-setup-scalable-jenkins-on-top-of-a-kubernet

Rozwiązywanie problemów z uruchomioną aplikacją

- Jak można śledzić, co aktualnie dzieje się z aplikacją?

Rozwiązywanie problemów z uruchomioną aplikacją

- Jak można śledzić, co aktualnie dzieje się z aplikacją?
- Jedna z aplikacji zaczyna działać wolno lub przestaje odpowiadać - jak znaleźć problem?

Rozwiązywanie problemów z uruchomioną aplikacją

- Jak można śledzić, co aktualnie dzieje się z aplikacją?
- Jedna z aplikacji zaczyna działać wolno lub przestaje odpowiadać - jak znaleźć problem?
- Musicie zdiagnozować i naprawić problem z aplikacją – jakie informacje będą pomocne?

Obserwowalność

- Obserwowalność (ang. observability) to zdolność systemu informatycznego do dostarczania wglądu w jego wewnętrzne działanie

Obserwowalność

- Obserwowalność (ang. observability) to zdolność systemu informatycznego do dostarczania wglądu w jego wewnętrzne działanie
- **Logi**
- **Metryki**
- **Traces**

Logi

- Chronologicznie uporządkowane informacje o zdarzeniach aplikacji

Logi

- Chronologicznie uporządkowane informacje o zdarzeniach aplikacji
- Różne poziomy granulacji logów, np. **TRACE - FATAL**

Logi

- Chronologicznie uporządkowane informacje o zdarzeniach aplikacji
- Różne poziomy granulacji logów, np. **TRACE - FATAL**
- Logi powinny zawierać informacje o ważnych zdarzeniach z życia aplikacji, takich jak akcje wykonane przez użytkowników, przetwarzanie w tle, wyjątki i błędy aplikacyjne

Logi

- Chronologicznie uporządkowane informacje o zdarzeniach aplikacji
- Różne poziomy granulacji logów, np. **TRACE – FATAL**
- Logi powinny zawierać informacje o ważnych zdarzeniach z życia aplikacji, takich jak akcje wykonane przez użytkowników, przetwarzanie w tle, wyjątki i błędy aplikacyjne

- Przykładowe logi:

  Aug 10 12:34:56 server1 systemd[1]: Started Apache HTTP Server.

Logowanie – dobre praktyki

- Używaj odpowiedniego poziomu logowania

Logowanie – dobre praktyki

- Używaj odpowiedniego poziomu logowania
- Wzbogać logi o identyfikatory obiektów systemowych, takich jak identyfikator klienta czy użytkownika

Logowanie – dobre praktyki

- Używaj odpowiedniego poziomu logowania
- Wzbogać logi o identyfikatory obiektów systemowych, takich jak identyfikator klienta czy użytkownika
- Korzystaj z identyfikatorów korelacji (ang. correlation ID)

Logowanie – dobre praktyki

- Używaj odpowiedniego poziomu logowania
- Wzbogać logi o identyfikatory obiektów systemowych, takich jak identyfikator klienta czy użytkownika
- Korzystaj z identyfikatorów korelacji (ang. correlation ID)
- Nie zapisuj w logach danych wrażliwych: haseł, identyfikatorów sesji użytkownika, numerów kont bankowych, numerów PESEL itp.

Logowanie – dobre praktyki

- Używaj odpowiedniego poziomu logowania
- Wzbogać logi o identyfikatory obiektów systemowych, takich jak identyfikator klienta czy użytkownika
- Korzystaj z identyfikatorów korelacji (ang. correlation ID)
- Nie zapisuj w logach danych wrażliwych: haseł, identyfikatorów sesji użytkownika, numerów kont bankowych, numerów PESEL itp.
- Retencja logów

Logowanie – dobre praktyki

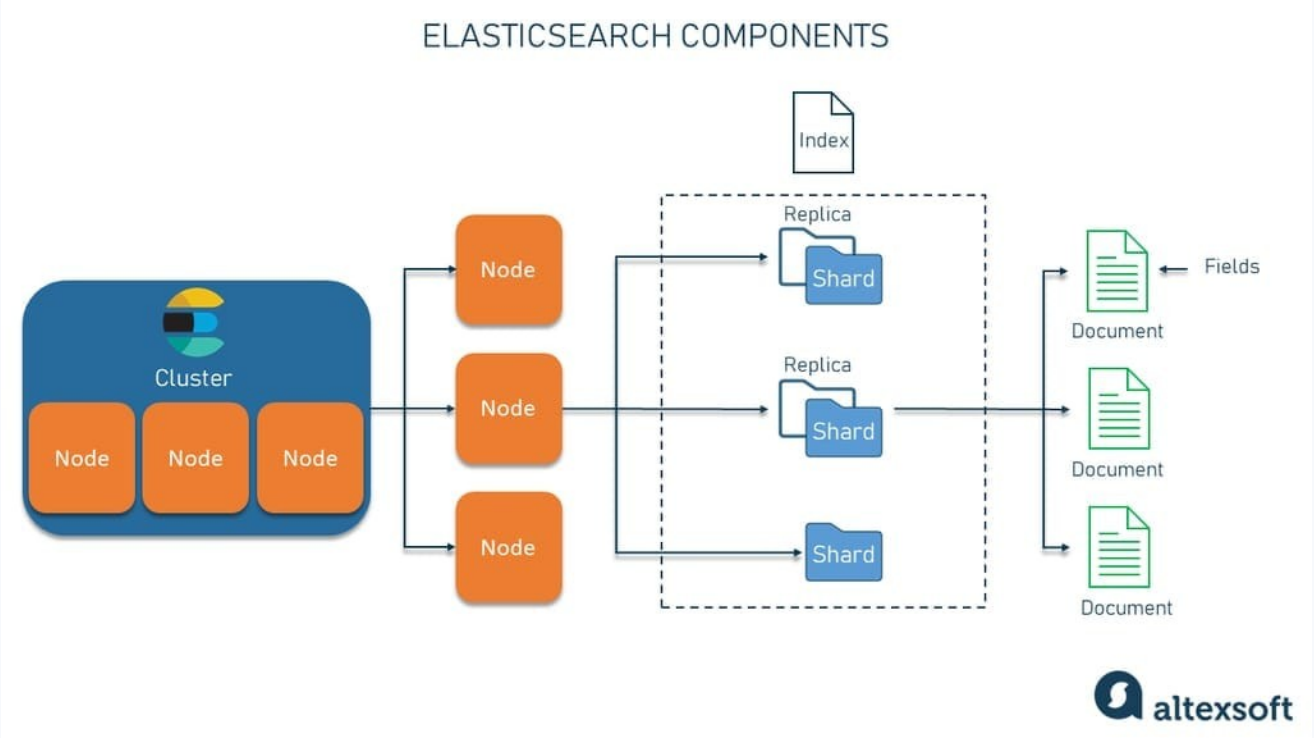
- Używaj odpowiedniego poziomu logowania
- Wzbogać logi o identyfikatory obiektów systemowych, takich jak identyfikator klienta czy użytkownika
- Korzystaj z identyfikatorów korelacji (ang. correlation ID)
- Nie zapisuj w logach danych wrażliwych: haseł, identyfikatorów sesji użytkownika, numerów kont bankowych, numerów PESEL itp.
- Retencja logów
- **https://cheatsheetseries.owasp.org/cheatsheets/Logging_Cheat_Sheet.html**

Logi – Elastic stack

- ELK: Elastic + Logstash + Kibana

Źródło: https://www.geeksforgeeks.org/what-is-elastic-stack-and-elasticsearch/

Elasticsearch - architektura

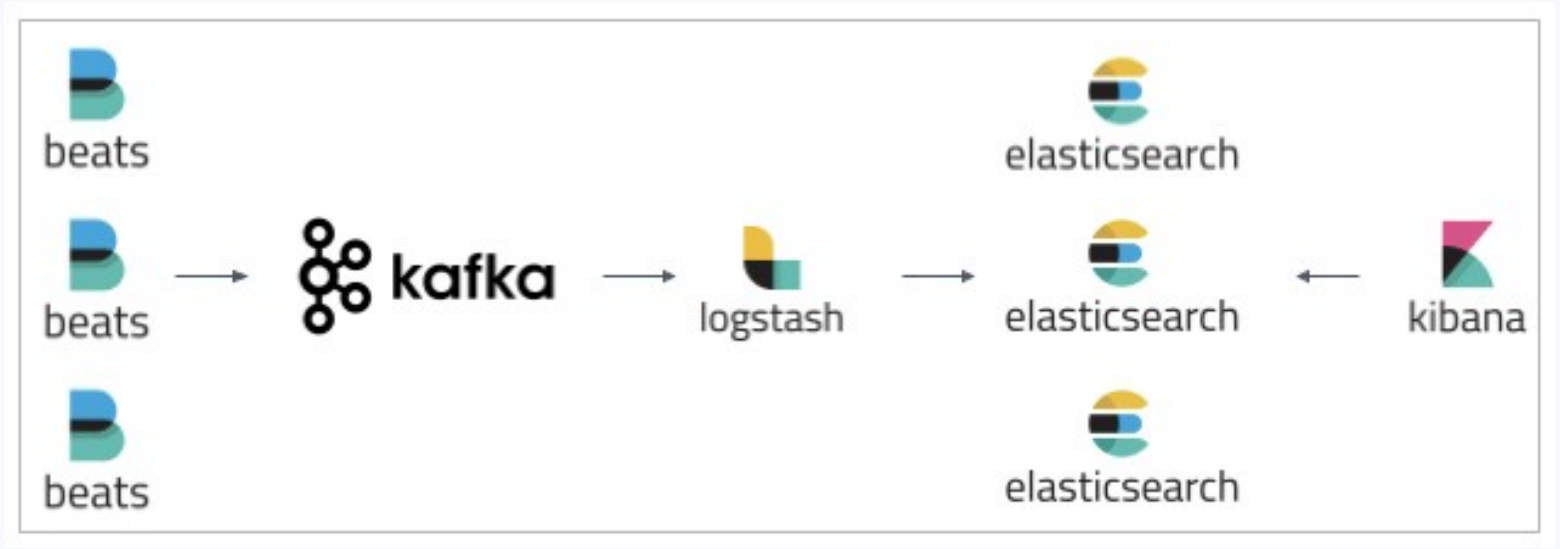


Źródło: https://www.altexsoft.com/blog/elasticsearch-pros-cons/

ELK - demo

- Repozytorium:
  - **https://github.com/kszalast/todolist.git**
- Branch:
  - **observability_2_jenkins**

ELK w środowisku produkcyjnym



Źródło: https://dzone.com/articles/deploying-kafka-with-the-elk-stack

Metryki

- Dane ilościowe, mierzalne i wyrażane liczbowo

Metryki

- Dane ilościowe, mierzalne i wyrażane liczbowo
- Pozwalają kontrolować stan systemu
  - Obserwowanie trendów i szukanie anomalii
  - Debugowanie błędów

Metryki

- Dane ilościowe, mierzalne i wyrażane liczbowo
- Pozwalają kontrolować stan systemu
  - Obserwowanie trendów i szukanie anomalii
  - Debugowanie błędów
- Co mierzyć – Kirk Pepperdine *The Box model*
  - *Ludzie*: akcje generowane przez użytkowników
  - *Aplikacje*: pule wątków, rozmiary kolejek, ilość zapytań
  - *Runtime (JVM)*: GC, użycie pamięci, wątki
  - *Infrastruktura*: użycie procesora, I/O

Wystawianie metryk przez aplikację

- Aplikacja może udostępniać metryki na wiele sposobów:
  - Logi
  - Wysyłanie bezpośrednio do systemów monitorujących
  - Endpointy HTTP

Wystawianie metryk przez aplikację

- Aplikacja może udostępniać metryki na wiele sposobów:
  - Logi
  - Wysyłanie bezpośrednio do systemów monitorujących
  - Endpointy HTTP
- Istnieje wiele gotowych bibliotek do zbierania metryk – Micrometer

Narzędzie Prometheus

- Narzędzie służące do zbierania metryk i alertowania

Narzędzie Prometheus

- Narzędzie służące do zbierania metryk i alertowania
- Baza danych szeregów czasowych

Narzędzie Prometheus

- Narzędzie służące do zbierania metryk i alertowania
- Baza danych szeregów czasowych
- Zbiera metryki z różnych źródeł przez HTTP (HTTP pull requests)

Narzędzie Prometheus

- Narzędzie służące do zbierania metryk i alertowania
- Baza danych szeregów czasowych
- Zbiera metryki z różnych źródeł przez HTTP (HTTP pull requests)
- Umożliwia integrację z różnymi frameworkami, systemami i usługami (np. Kubernetes, Spring Boot)

Narzędzie Grafana

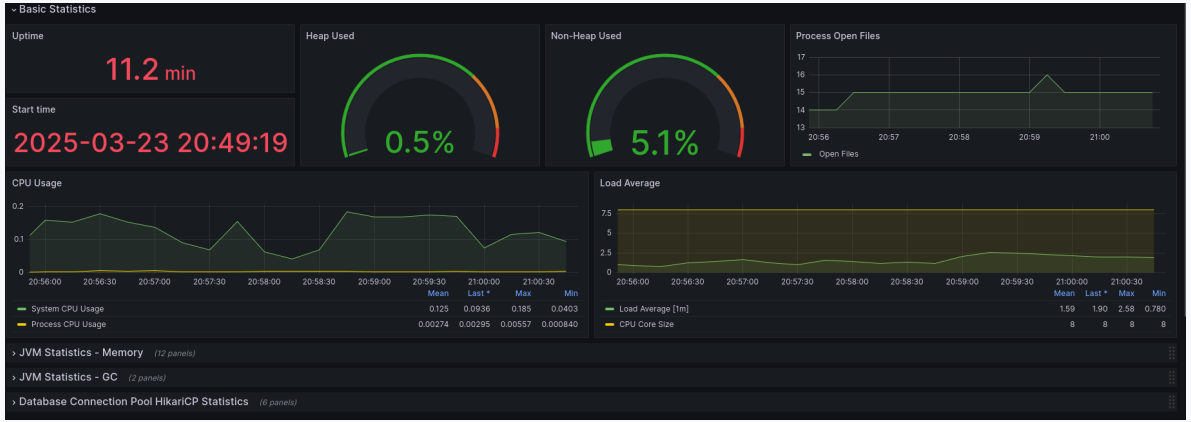
- Narzędzie do wizualizacji i analizy danych

Narzędzie Grafana

- Narzędzie do wizualizacji i analizy danych
- Tworzenie zaawansowanych wykresów, dashboardów i paneli

Narzędzie Grafana

- Narzędzie do wizualizacji i analizy danych
- Tworzenie zaawansowanych wykresów, dashboardów i paneli
- Obsługa różnych źródeł danych:
  - Prometheus
  - Elasticsearch
  - MySQL



Metryki - demo

- Repozytorium:
  - **https://github.com/kszalast/todolist.git**
- Branch:
  - **observability_3_metrics**

Metryki – dobre praktyki

- Zidentyfikuj, co chcesz osiągnąć dzięki metrykom
  - Poprawa czasu odpowiedzi
  - Zmniejszenie współczynnika błędów

Metryki – dobre praktyki

- Zidentyfikuj, co chcesz osiągnąć dzięki metrykom
  - Poprawa czasu odpowiedzi
  - Zmniejszenie współczynnika błędów
- Kluczowe metryki powinny być alertowane

Metryki – dobre praktyki

- Zidentyfikuj, co chcesz osiągnąć dzięki metrykom
  - Poprawa czasu odpowiedzi
  - Zmniejszenie współczynnika błędów
- Kluczowe metryki powinny być alertowane
- Prometheus - unikaj “cardinality bombs”

Metryki – dobre praktyki

- Zidentyfikuj, co chcesz osiągnąć dzięki metrykom
  - Poprawa czasu odpowiedzi
  - Zmniejszenie współczynnika błędów
- Kluczowe metryki powinny być alertowane
- Prometheus - unikaj “cardinality bombs”
- Metryki powinny ewoluować wraz z rozwojem aplikacji

Metryki – dobre praktyki

- Zidentyfikuj, co chcesz osiągnąć dzięki metrykom
  - Poprawa czasu odpowiedzi
  - Zmniejszenie współczynnika błędów
- Kluczowe metryki powinny być alertowane
- Prometheus - unikaj “cardinality bombs”
- Metryki powinny ewoluować wraz z rozwojem aplikacji
- Nie zakładaj, że metryki są zawsze precyzyjne

Skalowanie w oparciu o metryki - HorizontalPodAutoscaler

- Automatyczne skalowanie liczby replik podów w oparciu o metryki

Skalowanie w oparciu o metryki - HorizontalPodAutoscaler

- Automatyczne skalowanie liczby replik podów w oparciu o metryki
- Metryki zbierane są domyślnie co 15 sekund

Skalowanie w oparciu o metryki - HorizontalPodAutoscaler

- Automatyczne skalowanie liczby replik podów w oparciu o metryki
- Metryki zbierane są domyślnie co 15 sekund
- Obliczanie aktualnej ilości podów w oparciu o zebrane metryki:

$$desiredReplicas = ceil\left[currentReplicas \times \frac{currentMetricValue}{desiredMetricValue}\right]$$

Skalowanie w oparciu o metryki - HorizontalPodAutoscaler

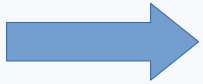
- Automatyczne skalowanie liczby replik podów w oparciu o metryki
- Metryki zbierane są domyślnie co 15 sekund
- Obliczanie aktualnej ilości podów w oparciu o zebrane metryki:

$$desiredReplicas = ceil\left[currentReplicas \times \frac{currentMetricValue}{desiredMetricValue}\right]$$

- Uwzględnienie parametrów minReplicas i maxReplicas

Skalowanie w oparciu o metryki

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: exchange-rates
  namespace: todo-app
spec:
  replicas: 1
  selector:
    matchLabels:
      app: exchange-rates
  template:
    metadata:
      labels:
        app: exchange-rates
    spec:
      containers:
        - name: exchange-rates
          image: exchange-rates:latest
          imagePullPolicy: Never
          ports:
            - containerPort: 8080
```



```yaml
apiVersion: autoscaling/v2
kind: HorizontalPodAutoscaler
metadata:
  name: exchange-rates
  namespace: todo-app
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: exchange-rates
  minReplicas: 1
  maxReplicas: 10
  metrics:
    - type: Pods
      pods:
        metric:
          name: http_requests
        target:
          type: AverageValue
          averageValue: "100"
```

Dziękuję za uwagę!

Pytania?