

***Wojskowa Akademia Techniczna  
im. Jarosława Dąbrowskiego***



**Wydział Cybernetyki, kierunek informatyka - inżynieria systemów**

Realizacja zadania laboratoryjnego w ramach przedmiotu:

***Systemy Baz Danych***

Temat laboratorium:

***Bazy Danych typu wyszukiwarki***

**Opracował:** Radosław Relidzyński, **Grupa:** WCY23IX3S4

## Spis treści

Wstęp teoretyczny .....	3
Treść zadania.....	3
Środowisko.....	4
Implementacja środowiska.....	4
Definiowanie środowiska docker-compose.yml .....	4
Zarządzanie danymi (logami) logstash.conf.....	7
Przetwarzanie wstępne logów.....	8
Uruchomienie środowiska.....	9
Wizualizacja danych .....	14
Podsumowanie.....	16

## Wstęp teoretyczny

**Baza danych** – „uporządkowany zbiór danych określających wybrany fragment rzeczywistości lub problemu, które są przechowywane trwale w pamięci komputerowej do której może mieć dostęp wielu użytkowników w dowolnej chwili czasu.”

**System zarządzania bazami danych** – „zorganizowany zbiór narzędzi (programów komputerowych i bibliotek), które umożliwiają wykonanie podstawowych operacji na danych (CRUD) zawartych w jednej lub więcej bazach danych.”

System baz danych – jego definicja wyraża się wzorem:

$$SBD = \langle \{U, SO, DB, SZBD, P\}, R \rangle$$

Gdzie:

*U* – zbiór urządzeń

*SO* – system operacyjny

*BD* – baza danych (schemat, stan, ścieżki dostępu)

*SZBD* – system zarządzania bazą danych

*P* – polecenia użytkownika

*R* – relacje między obiektami *SBD* a otoczeniem

[źródło: materiały z wykładu „Temporalne bazy danych” dr inż. Jarosława Koszeli]

## Treść zadania

Wymagania:

- Implementacja środowiska
- 4 wykresy dashboard
- opisać proces tworzenia indeksu z wgrywaniem danych
- opisać skąd są logi (plik) i w jaki sposób są przetwarzane (parser, groki)
- stworzenie 3 węzłów elasticsearch
- przedstawić i opisać konfigurację z logstashem
- prezentacja liczby wstawionych dokumentów
- pokazać jak przeparsowano datę (format/mapping, dopasowanie)
- Stworzyć wizualizację danych – 4 wizualizacje (wykresy dashboard)

# Środowisko

„**Elasticsearch** jest silnikiem wyszukiwania i analizy danych pozwalającym również na składowanie danych. Cechuje się dużą szybkością oraz skalowalnością. Komunikacja z silnikiem odbywa się przy użyciu zapytań REST, dzięki czemu jest bardzo prosta w zaimplementowaniu do naszego systemu.”

[źródło: <https://ermlab.com/blog/technicznie/elasticsearch-jako-narzedzie-do-przeszukiwania-danych/>]

Środowisko zostanie utworzone lokalnie przy pomocy narzędzia docker-compose oraz różnych kontenerów, głównie dla elasticsearch oraz kibany dostarczającej narzędzia wizualizacji danych.

## Implementacja środowiska

### Definiowanie środowiska docker-compose.yml

```
version: '3.8'
services:
  es01:
    image: elasticsearch:7.16.2
    container_name: es01
    restart: always
    volumes:
      - elastic_data1:/usr/share/elasticsearch/data/
    environment:
      - node.name=es01
      - cluster.name=es-docker-cluster
      - discovery.seed_hosts=es02,es03
      - cluster.initial_master_nodes=es01,es02,es03
      - xpack.ml.enabled=false
      - xpack.monitoring.collection.enabled=true
      - bootstrap.memory_lock=true
      - "ES_JAVA_OPTS=-Xmx256m -Xms256m"
    ulimits:
      memlock:
        soft: -1
        hard: -1
    ports:
      - '9200:9200'
    networks:
      - elastic

  es02:
    image: elasticsearch:7.16.2
    container_name: es02
    restart: always
    volumes:
      - elastic_data2:/usr/share/elasticsearch/data/
    environment:
      - node.name=es02
      - cluster.name=es-docker-cluster
      - discovery.seed_hosts=es01,es03
      - cluster.initial_master_nodes=es01,es02,es03
      - xpack.ml.enabled=false
      - xpack.monitoring.collection.enabled=true
      - bootstrap.memory_lock=true
      - "ES_JAVA_OPTS=-Xmx256m -Xms256m"
    ulimits:
      memlock:
```

```

    soft: -1
    hard: -1
  ports:
    - '9201:9200'
  networks:
    - elastic

es03:
  image: elasticsearch:7.16.2
  container_name: es03
  restart: always
  volumes:
    - elastic_data3:/usr/share/elasticsearch/data/
  environment:
    - node.name=es03
    - cluster.name=es-docker-cluster
    - discovery.seed_hosts=es01,es02
    - cluster.initial_master_nodes=es01,es02,es03
    - xpack.ml.enabled=false
    - xpack.monitoring.collection.enabled=true
    - bootstrap.memory_lock=true
    - "ES_JAVA_OPTS=-Xmx256m -Xms256m"
  ulimits:
    memlock:
      soft: -1
      hard: -1
  ports:
    - '9202:9200'
  networks:
    - elastic

logstash:
  image: logstash:7.16.2
  container_name: logstash
  restart: always
  volumes:
    - ./logstash:/logstash_dir
  command: logstash -f /logstash_dir/logstash.conf
  depends_on:
    - es01
  ports:
    - '9600:9600'
  environment:
    LS_JAVA_OPTS: "-Xmx256m -Xms256m"
  networks:
    - elastic

kibana:
  image: kibana:7.16.2
  container_name: kibana
  restart: always
  ports:
    - '5601:5601'
  environment:
    ELASTICSEARCH_URL: http://es01:9200
    ELASTICSEARCH_HOSTS:
'["http://es01:9200","http://es02:9200","http://es03:9200"]'
  networks:
    - elastic

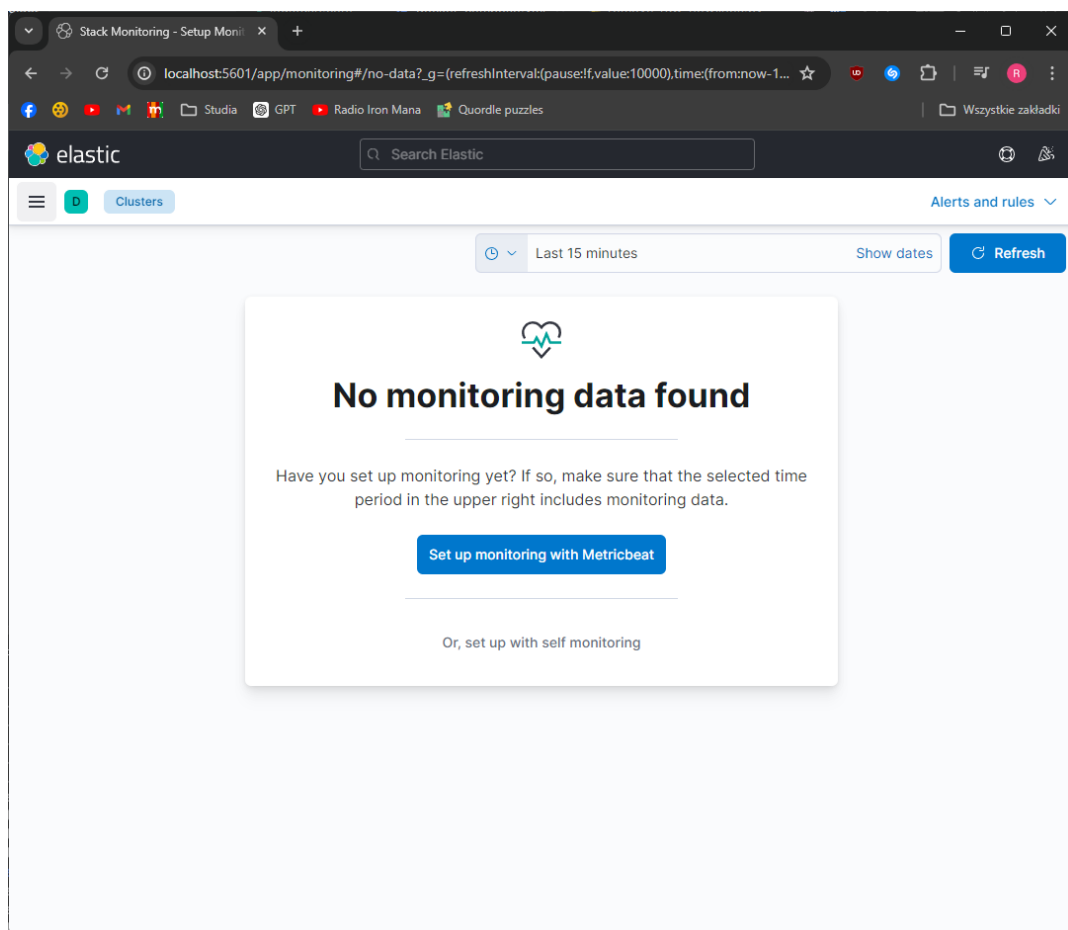
volumes:
  elastic_data1:
  elastic_data2:
  elastic_data3:

```

```
networks:
  elastic:
```

Zastosowane kontenery:

- es1, es2, es3 – 3 węzły Elasticsearch
  - obraz: elasticsearch:7.16.2
  - wykorzystywane porty: 9200, 9201, 9202 (wszystkie przekierowywane na 9200)
  - sieć: elastic (jedyna dostępna)
  - paczka kolekcji monitoringu ustawiona na true. Jest to automatyczne włączenie monitoringu danych tak, żeby przy uruchomieniu nie wyświetlała się taka wiadomość:



- logstash
  - obraz: logstash:7.16.2
  - Wykorzystywany port: 9600
  - ścieżka do folderu z plikiem logstash: ./logstash/
  - instrukcja wywołania logstash wraz z podaniem nazwy pliku: logstash -f /logstash\_dir/logstash.conf
- kibana
  - obraz: kibana:7.16.2
  - Wykorzystywany port: 5601
  - adres url do elasticsearch: <http://es01:9200>

- o adresy dla hostów elasticsearch:  
'["http://es01:9200","http://es02:9200","http://es03:9200"]'

Uruchomienie polecenia „docker-compose up -d” tworzy 5 kontenerów, 3 węzły elasticsearch, interfejs do wizualizacji kibana oraz narzędzie przetwarzania danych logstash.

## Zarządzanie danymi (logami) logstash.conf

```
input {
  file {
    path => "/logstash_dir/logs_source.csv"
    start_position => "beginning"
    sincedb_path => "/dev/null"
  }
}

filter {
  csv {
    separator => ","
    columns => [
      "error_level",
      "error_message",
      "raw",
      "remote_host",
      "remote_logname",
      "remote_user",
      "request_header_referer",
      "request_header_user_agent",
      "request_header_user_agent_browser_family",
      "request_header_user_agent_os_family",
      "request_header_user_agent_os_version_string",
      "request_http_ver",
      "request_method",
      "request_url",
      "response_bytes_clf",
      "status",
      "time_received"
    ]
    skip_header => true
  }

  date {
    match => ["time_received", "[dd/MMM/yyyy:HH:mm:ss Z]", "[EEE MMM dd HH:mm:ss
yyyy]", "[dd/MMM/yyyy:HH:mm:ss]", "[dd/MMM/yyyy HH:mm:ss]"]
    target => "@timestamp"
    timezone => "UTC"
  }
}

output {
  stdout {
    codec => rubydebug
  }

  elasticsearch {
    hosts => ["http://es01:9200","http://es02:9200","http://es03:9200"]
    index => "new_logs"
  }

  file {
    path => "/logstash_dir/log_output.log"
  }
}
```

### 1. Sekcja 'input':

- file – definiowanie źródła w postaci ścieżki do pliku, miejsca z którego ma zacząć przetwarzać (w tym przypadku od początku) oraz określa ścieżkę do pliku „sincedb” śledzącą progres przetwarzania pliku. Przekierowywanie do „/dev/null” oznacza, że nie będzie on śledzony, wszystkie przekierowywane tam dane zostaną utracone.

### 2. Sekcja „filter”:

- csv – deklaracja separatora i listy kolumn do przetworzenia pliku csv, wraz z pominięciem pierwszego wiersza (nagłówków)
- date – definicja formatów dat do dopasowania, strefy czasowej oraz tego, która data ma zostać dodana do „@timestamp”. Są one różne w zależności od rekordu, stąd wymagane jest stworzenie różnych formatów.

### 3. Sekcja „output”:

- stdout – przetwarzanie standardowego wyjścia w formacie „rubydebug”
- elasticsearch – definicja hostów wyjściowych dla klastra elasticsearch oraz nazwy indeksu, do którego będą zapisywane przetworzone dane
- file – podanie ścieżki i nazwy pliku, w którym zapisane mają być przetworzone logi.

## Przetwarzanie wstępne logów

Analiza logów prezentuje się następująco:

- Pierwszy wiersz to zbiór kolumn
- Logi są zapisane w jednej lub dwóch liniach
- Logi posiadające „error\_level” oraz „error\_message” są zapisane w jednej linii
- Logi nie posiadające „error\_level” ani „error\_message” są zapisane w dwóch liniach

Aby ułatwić ich przetwarzanie, logi zapisane w dwóch liniach przetworzone zostały tak, aby być w jednej linii. Dzięki temu logstash nie wymaga konfiguracji przy pomocy instrukcji „multiline”, będzie mógł bez tego trafnie je przeparsować

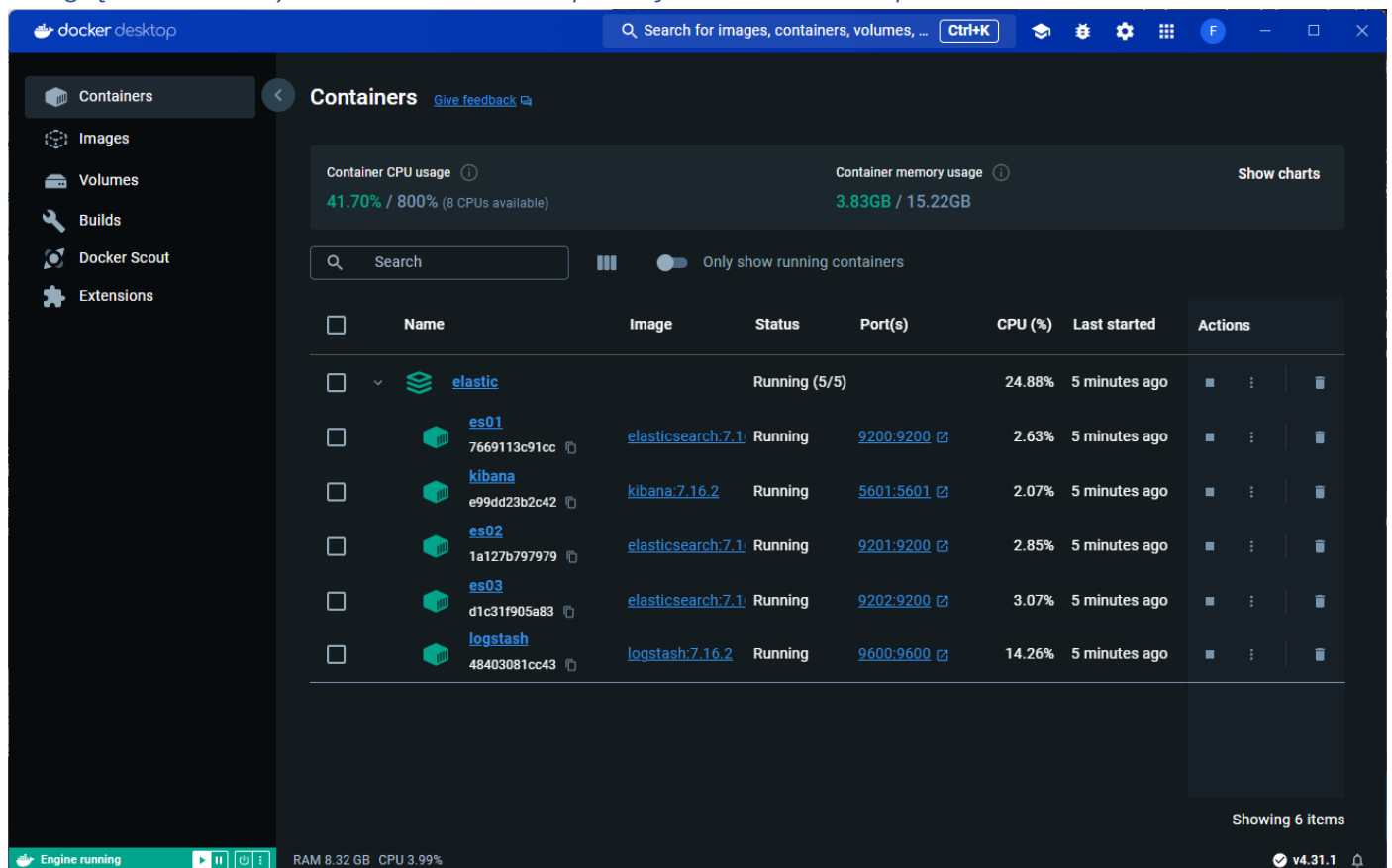


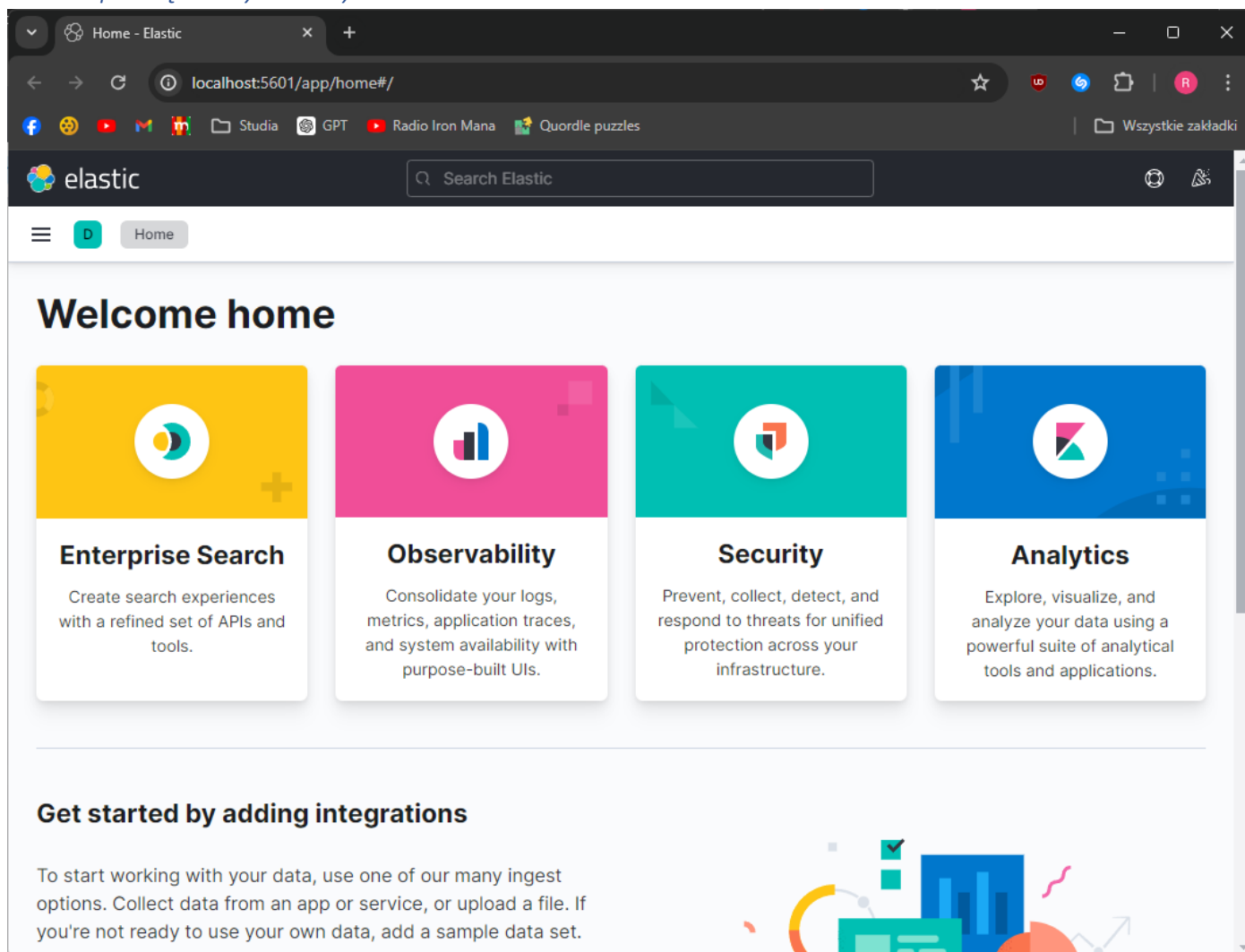
# Uruchomienie środowiska

Wywołanie polecenia „docker-compose up -d”

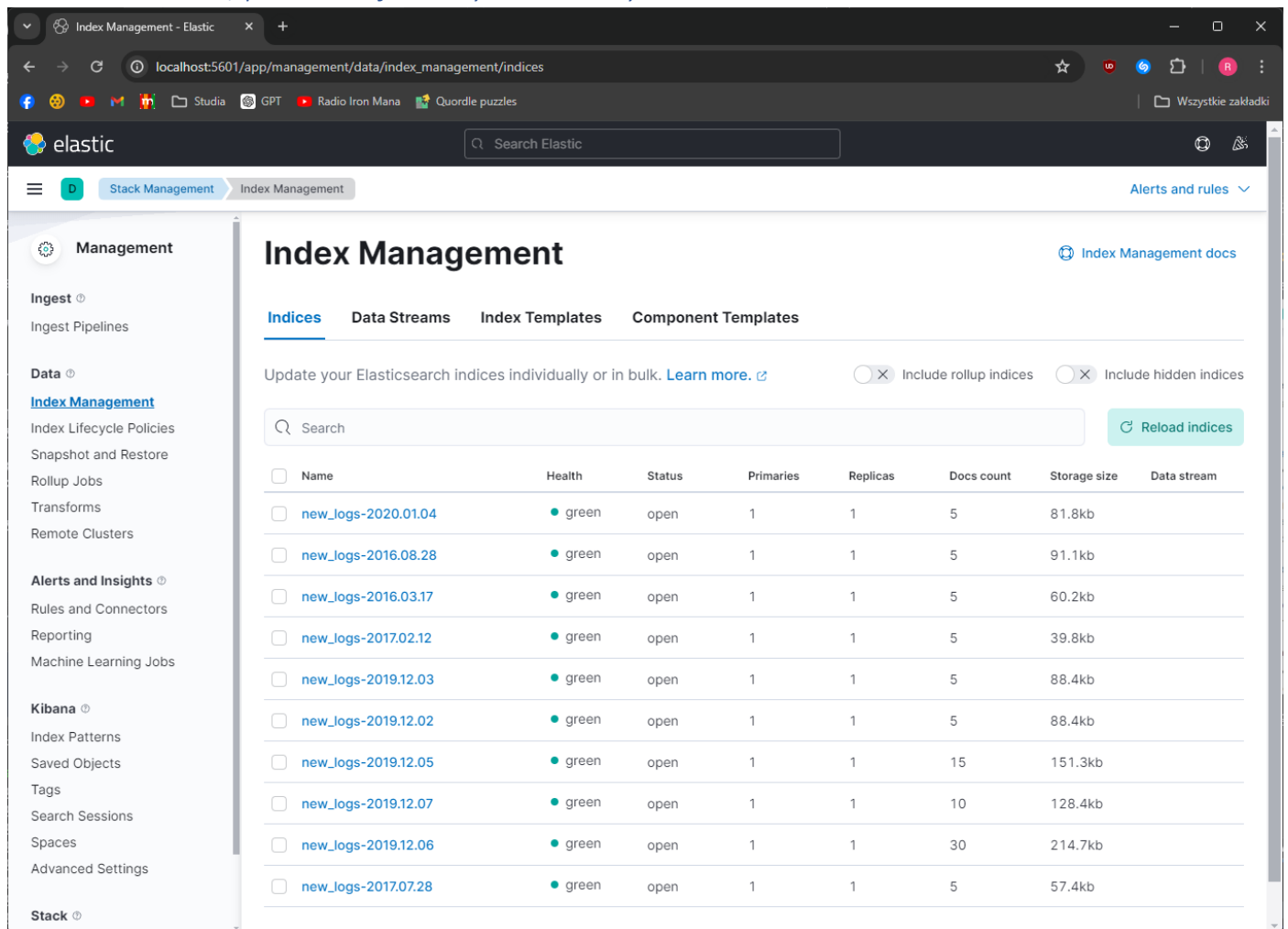
```
radek@Radoslaw: ~/elastic
radek@Radoslaw:~/elastic$ docker-compose up -d
Creating network "elastic_elastic" with the default driver
Creating volume "elastic_elastic_data1" with default driver
Creating volume "elastic_elastic_data2" with default driver
Creating volume "elastic_elastic_data3" with default driver
Creating kibana ... done
Creating es02 ... done
Creating es01 ... done
Creating es03 ... done
Creating logstash ... done
radek@Radoslaw:~/elastic$
```

Podgląd utworzonych kontenerów w aplikacji Docker Desktop





## Widok indeksów, prezentacja liczby wstawionych dokumentów



The screenshot displays the Elastic Index Management interface in a web browser. The browser's address bar shows the URL `localhost:5601/app/management/data/index_management/indices`. The Elastic logo and a search bar are at the top. The left sidebar contains navigation links for Management, Ingest, Data, Alerts and Insights, Kibana, and Stack. The main content area is titled "Index Management" and includes tabs for Indices, Data Streams, Index Templates, and Component Templates. Below the tabs, there is a text prompt to update indices, a search bar, and two toggle switches for "Include rollout indices" and "Include hidden indices". A "Reload indices" button is also present. The central table lists various indices, all with a "green" health status and "open" status. The "Docs count" column shows the number of documents for each index.

<input type="checkbox"/>	Name	Health	Status	Primaries	Replicas	Docs count	Storage size	Data stream
<input type="checkbox"/>	<a href="#">new_logs-2020.01.04</a>	green	open	1	1	5	81.8kb	
<input type="checkbox"/>	<a href="#">new_logs-2016.08.28</a>	green	open	1	1	5	91.1kb	
<input type="checkbox"/>	<a href="#">new_logs-2016.03.17</a>	green	open	1	1	5	60.2kb	
<input type="checkbox"/>	<a href="#">new_logs-2017.02.12</a>	green	open	1	1	5	39.8kb	
<input type="checkbox"/>	<a href="#">new_logs-2019.12.03</a>	green	open	1	1	5	88.4kb	
<input type="checkbox"/>	<a href="#">new_logs-2019.12.02</a>	green	open	1	1	5	88.4kb	
<input type="checkbox"/>	<a href="#">new_logs-2019.12.05</a>	green	open	1	1	15	151.3kb	
<input type="checkbox"/>	<a href="#">new_logs-2019.12.07</a>	green	open	1	1	10	128.4kb	
<input type="checkbox"/>	<a href="#">new_logs-2019.12.06</a>	green	open	1	1	30	214.7kb	
<input type="checkbox"/>	<a href="#">new_logs-2017.07.28</a>	green	open	1	1	5	57.4kb	

## Tworzenie patternu dla indeksów

The screenshot shows the Kibana interface for managing index patterns. The browser address bar indicates the URL: `localhost:5601/app/management/kibana/indexPatterns/patterns/4025a700-29ba-11ef-94fc-253175731d73#/?_a=(tab:indexedFields)`. The page title is **new\_logs-\***. Below the title, there's a section for field mappings with a search bar and a table of indexed fields.

**Time field:** `@timestamp` (Default)

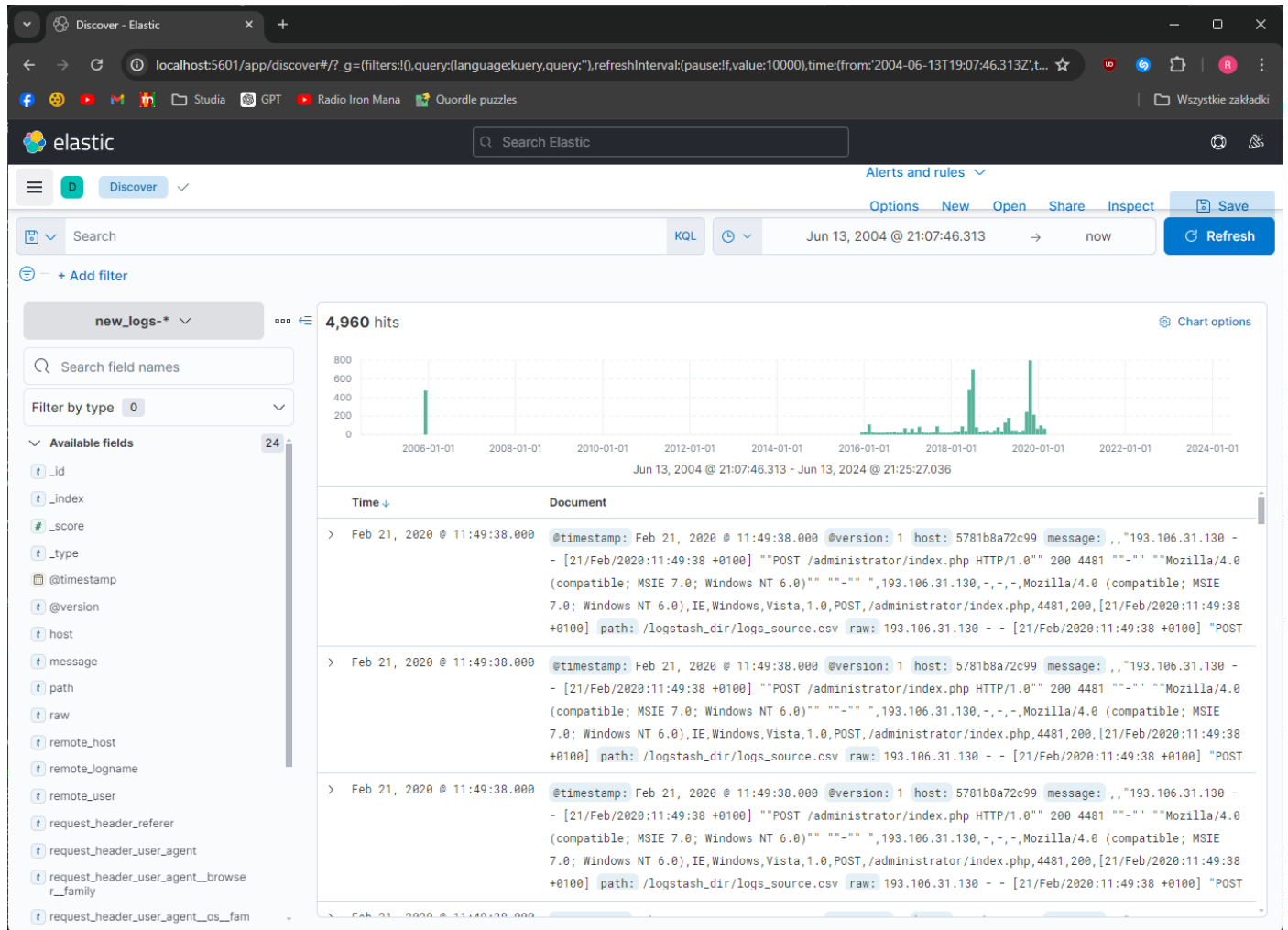
View and edit fields in **new\_logs-\***. Field attributes, such as type and searchability, are based on [field mappings](#) in Elasticsearch.

**Fields (48)**   **Scripted fields (0)**   **Field filters (0)**

Search:  All field types

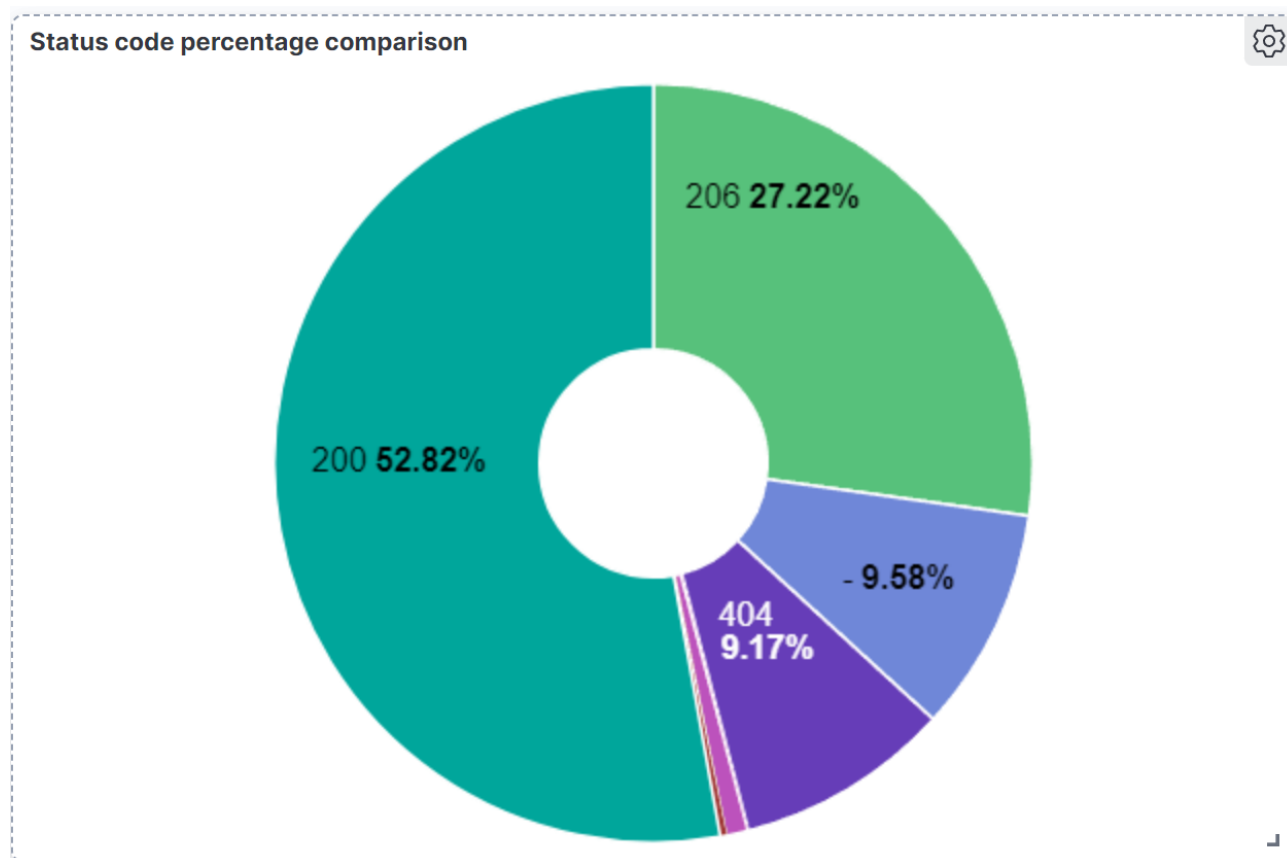
Name ↑	Type	Format	Searchable	Aggregatable	Excluded
<code>@timestamp</code>	date		•	•	
<code>@version</code>	text		•		
<code>@version.keyword</code>	keyword		•	•	
<code>_id</code>	<code>_id</code>		•	•	
<code>_index</code>	<code>_index</code>		•	•	
<code>_score</code>					
<code>_source</code>	<code>_source</code>				
<code>_type</code>	<code>_type</code>		•	•	
<code>error_level</code>	text		•		
<code>error_level.keyword</code>	keyword		•	•	

## Przegląd wszystkich logów

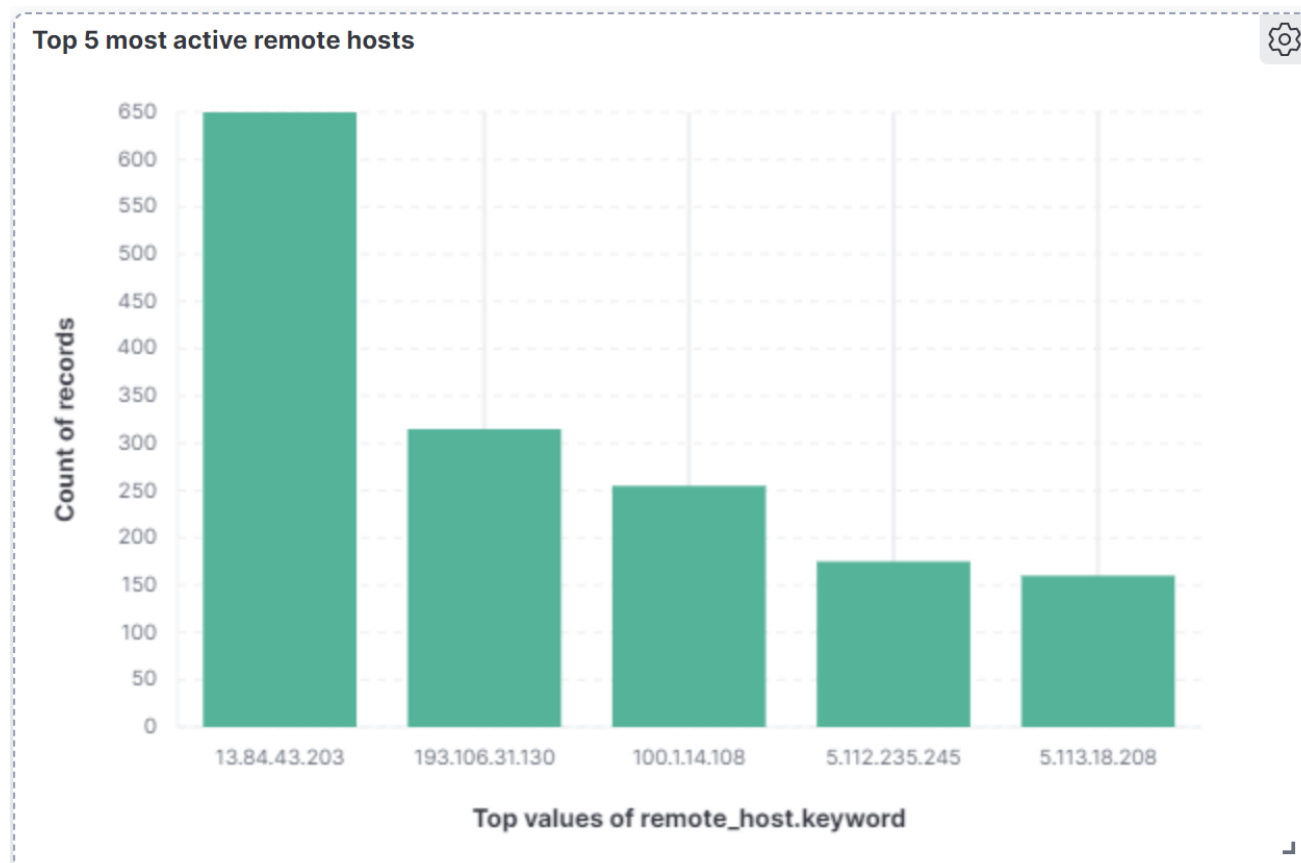


# Wizualizacja danych

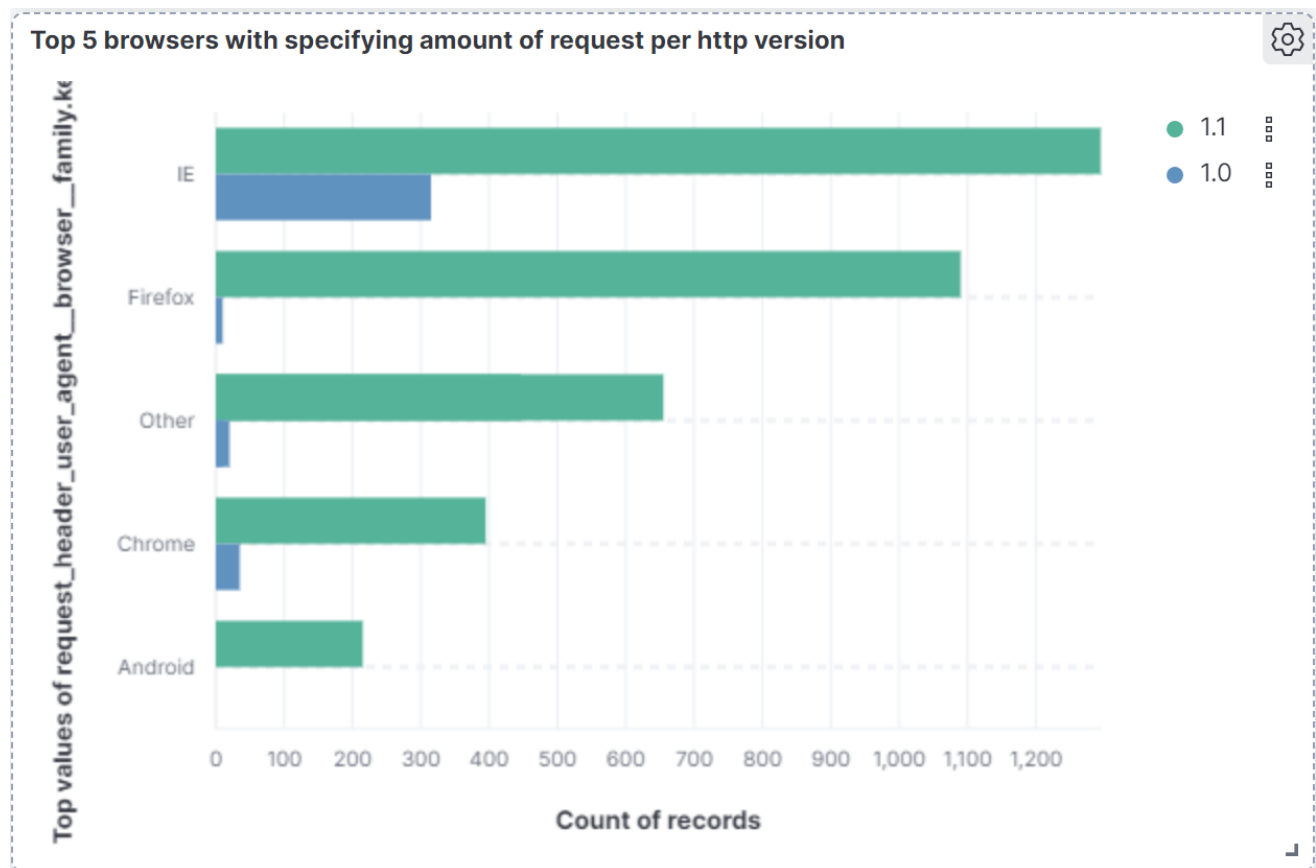
## Procentowy udział kodów odpowiedzi w logach



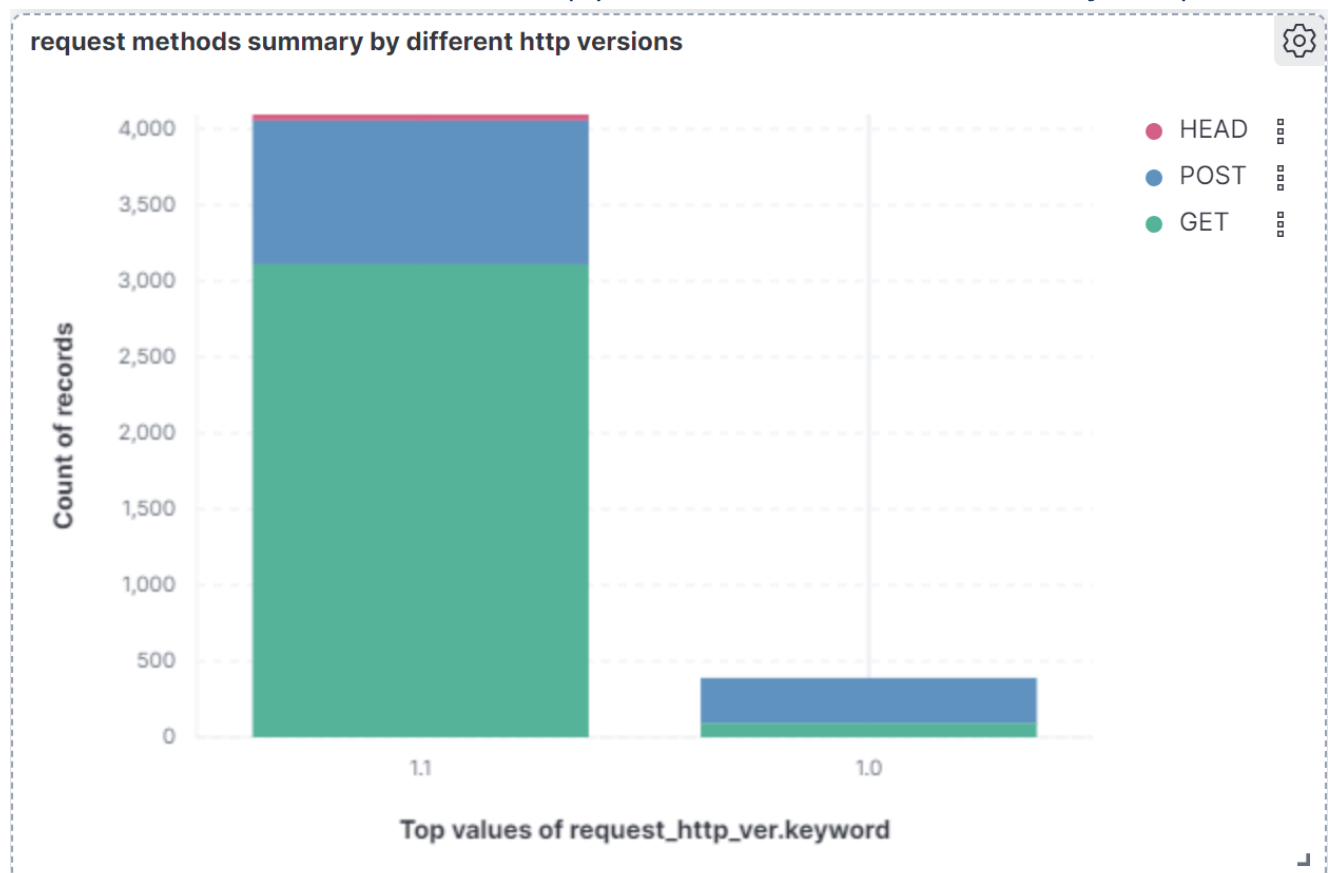
## Top 5 najbardziej aktywnych zdalnych hostów



## Top 5 najczęściej wykorzystywanych przeglądarek z wyszczególnieniem udziału wersji http



## Podsumowanie metod dla zapytań w zależności od wersji http



## Podsumowanie

W ramach przeprowadzonego zadania wykonany został pełny proces implementacji środowiska baz danych typu wyszukiwarki w ramach technologii „Elasticsearch”, obejmujący konfigurację oraz uruchomienie kontenerów Elasticsearch, Logstash oraz Kibana. Udało się przetworzyć logi. Wizualizacja wyników w Kibana pozwoliła na wizualizację wybranych aspektów logów.

Elasticsearch to bardzo efektywne narzędzie do przetwarzania i przeglądania logów. Jak pokazało przygotowywanie wizualizacji potrafi przetworzyć bardzo duże ilości rekordów w bardzo krótkim czasie. Możliwości narzędzia w zakresie zarządzania formatami oraz sposoby przechowywania rekordów wskazują na duży jego potencjał przy bardzo dużych systemach.