SPA-Assignment-2

**Streaming Log Analytics**

**Submission By**

Group 255 Students

Amogh Singhal

Abhay Thappan

Akshit Agarwal

AUGUST 28, 2022

**Birla Institute of Technology & Science, Pilani**

Streaming Log Analytics Solution

# Assignment details

Log analytics is a common big data use case that allows you to analyze log data from websites, mobile devices, servers, sensors, and more for a wide variety of applications such as digital marketing, application monitoring, fraud detection, ad tech, games, and IoT. In this assignment, you use **Open Source tools of your choice** to build an end-to-end log analytics solution that collects, ingests, processes, and loads both batch data and streaming data, and makes the processed data available to your users in analytics systems they are already using and in near real-time.

# Our Approach

As part of this assignment, we tried to explore multiple tools/resources that can help us achieve the goal. We would like to share some below

* Graylog - an Elasticsearch-based log management and analysis tool.

* Fluentd - It enables users to unify logs from various components and easily analyzes them.

* GoAccess - terminal-based open-source log analyzer that analyze and observe web server statistics in real-time.

* Octopussy - It sends alerts to networking devices about their applications and services supporting the Syslog protocol

* Apache Flume - one of the best open-source log analysis tools to use extensible data models for online analytic applications

* ELK Stack - enables users to aggregate logs from all connected systems and applications. It helps analyze logs and create visualizations for applications and infrastructure monitoring

* LOGalyze - centralized open-source management and network monitoring tool

* syslog-ng - It processes the gathered log data and transfers them to a preferred log analysis tool * NewRelic - a web application performance service designed to work in real-time with your live

web app

*Splunk - Splunk is a log search engine with native capability for building dashboards and alerts and out of the box capability to data insights using knowledge objects

# Final Tool for implementation

We investigated pros/cons as well as practicality of using above sources without incurring cost on cloud and limiting scope due to our personal system resources. We were already aware of python faker

through the classes and AWS kinesis solution, so log generation was not a problem. We were looking at a solution that can provide us with the required details as per assignment. We decided to go with ELK stack, which is a mélange of Elasticsearch, Logstash and Kibana plus using the faker generated logs for analysis. Let us share more about ELK, why we choose this and how it helps in Log analytics world.

## Elasticsearch, Logstash and Kibana

## Logo, company name Description automatically generated

* As its name suggests, **Elasticsearch** is designed to help users find matches within datasets using a wide range of query languages and types. Speed is this tool's number one advantage. It can be expanded into clusters of hundreds of server nodes to handle petabytes of data with ease.
* **Kibana** is a visualization tool that runs alongside Elasticsearch to allow users to analyze their data and build powerful reports. When you first install the Kibana engine on your server cluster, you will gain access to an interface that shows statistics, graphs, and even animations of your data.
* The final piece of ELK Stack is **Logstash**, which acts as a purely server-side pipeline into the Elasticsearch database. You can integrate Logstash with a variety of coding languages and APIs so that information from your websites and mobile applications will be fed directly into your

powerful Elastic Stalk search engine.

## Why Use ELK Stack?

In a data-driven world, databases must constantly handle increasingly larger amounts of data. Typically, analytic processes slow down as the amount of data a system handles continues to increase. The ELK stack can help increase these analytic processes. A brief overview of the benefits of the ELK stack

includes:

* ELK is a total log-analysis platform for search, analyses and visualization of log-generated data from different machines.
* ELK can securely pull, analyze and visualize data, in real time, from any source and format.
* ELK can perform centralized logging to help identify any server and application-related issues across multiple servers and correlate the logs in a particular time frame.
* ELK is geared to handle big data to provide crucial business insights.
* ELK is simple to use, set up and is user friendly.
* As an open-source program, Elk is highly cost effective.

## Log Analytics

Dozens of servers running multiple applications results in a lot of data to analyze. Logs are one of the most critical, but often overlooked, data sources. Within a company’s web-server logs, each individual

log file holds mostly unstructured information that is difficult, or sometimes, impossible to interpret. ELK is able to quickly analyze the log data and to identify opportunities as well as possible vulnerabilities.

It is critical to understand how the system works whenever problems arise. Having the ability to quickly locate the needed information will help expedite operations-related tasks and resolve problems.

Additionally, adding metrics to correlate logs provides increased visibility to help see the log history, what is currently happening and predict where a trend is headed.

DBAs typical must log on to multiple machines and comb through numerous files when an error occurs. The larger the system is the bigger a nightmare this becomes. ELK being able to turn this migraine-sized problem into a minor annoyance is a major reason why use ELK stack.

# Streaming data pipeline Architecture

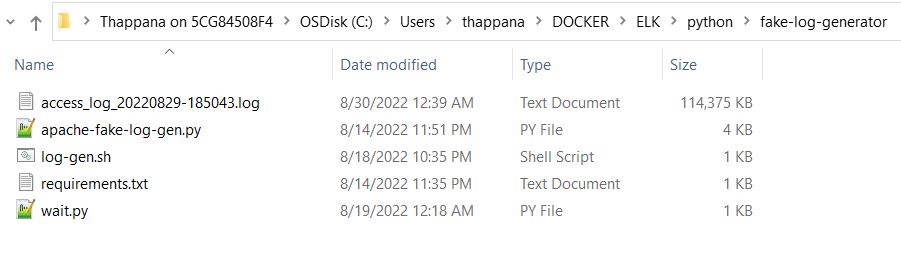
The proposed architectural diagram that we have created can be summarized below



Fig1. Explains ELK (Elasticsearch, Logstash, Kibana) integration with python Faker utility that generates random logs continuously (Apache access logs in or case)

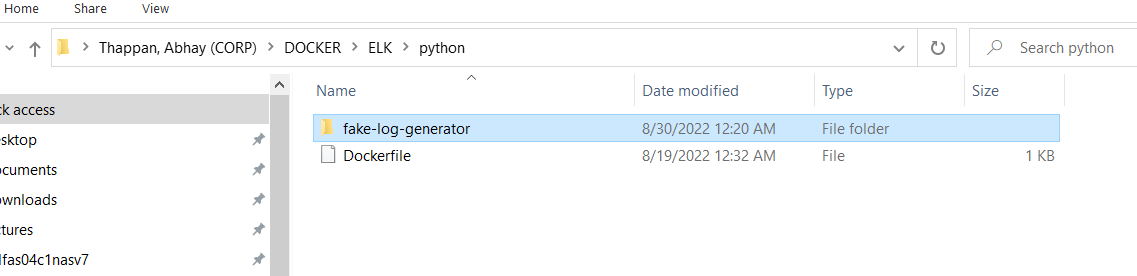
# Technical details

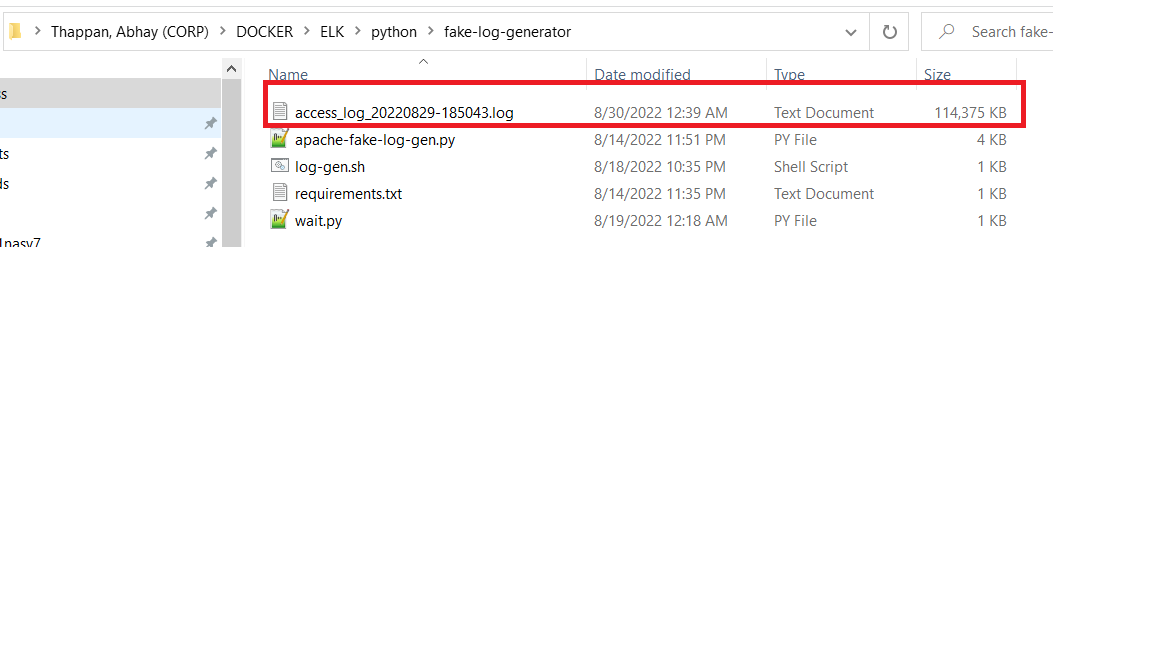
**Log generation:** We followed code from faker-apache-logs-generation repository that has code from open sources with personal updates.



This script generates a boatload of fake Apache logs very quickly. It is useful for generating fake workloads for [data ingest](http://github.com/streamsets/datacollector) and/or analytics applications. It can write log lines to console, to log files or directly to zip files. It utilizes the excellent [Faker](https://github.com/joke2k/faker/) library to generate realistic IP’s, URI's etc.

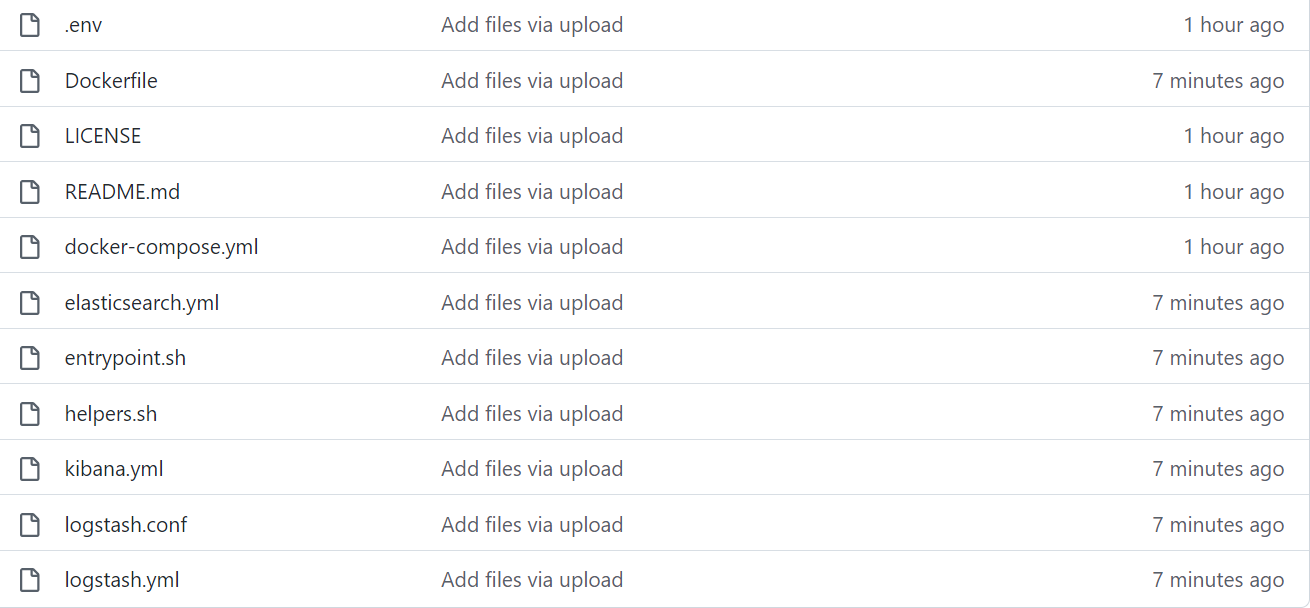
The Docker file listed below will help us smoothly deploy the code and generate logs with a wait time of 4000 milliseconds. Run the docker compose file and we will see logs getting generated. All the logs will be stored in file path given. Here is a sample path and file below:



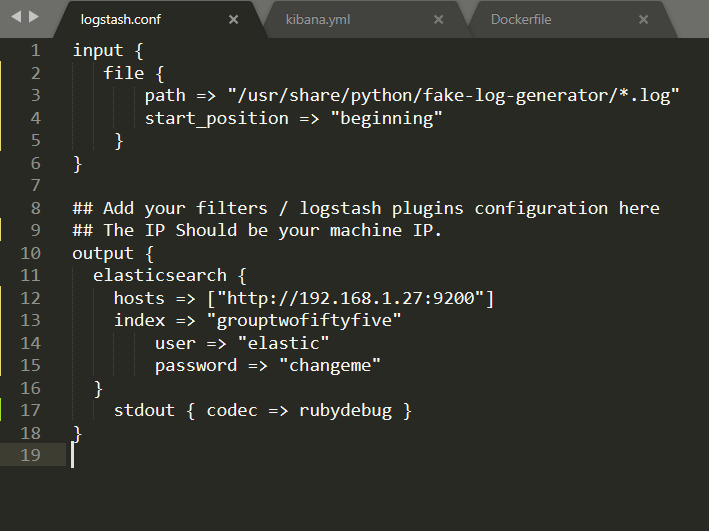


**ELK Installation**

To install Elastic stack, we need to run the latest version of the [Elastic stack](https://www.elastic.co/what-is/elk-stack) with Docker and Docker Compose. We followed open-source repository with individual updates to .yml files and can be seen here Github code- **ELK-installation-using-docker**

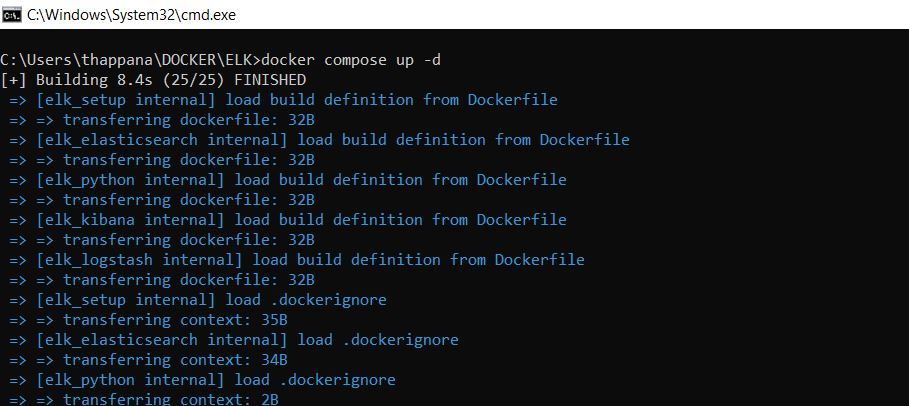


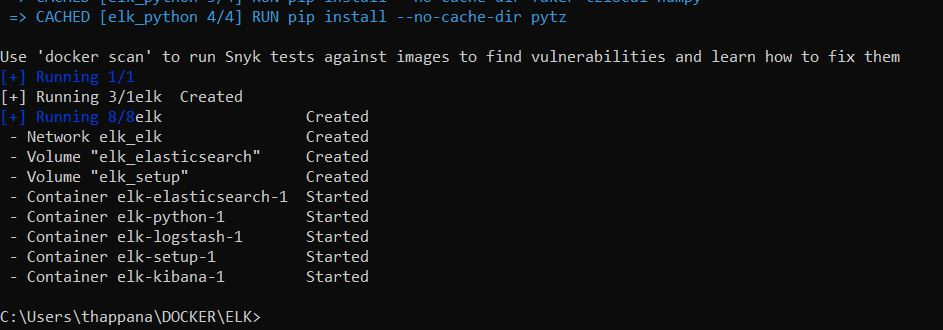
We had to make few changes to Logstash configuration files like below to provide system IP address and log file location. This will allow the components to interact and make logs accessible.



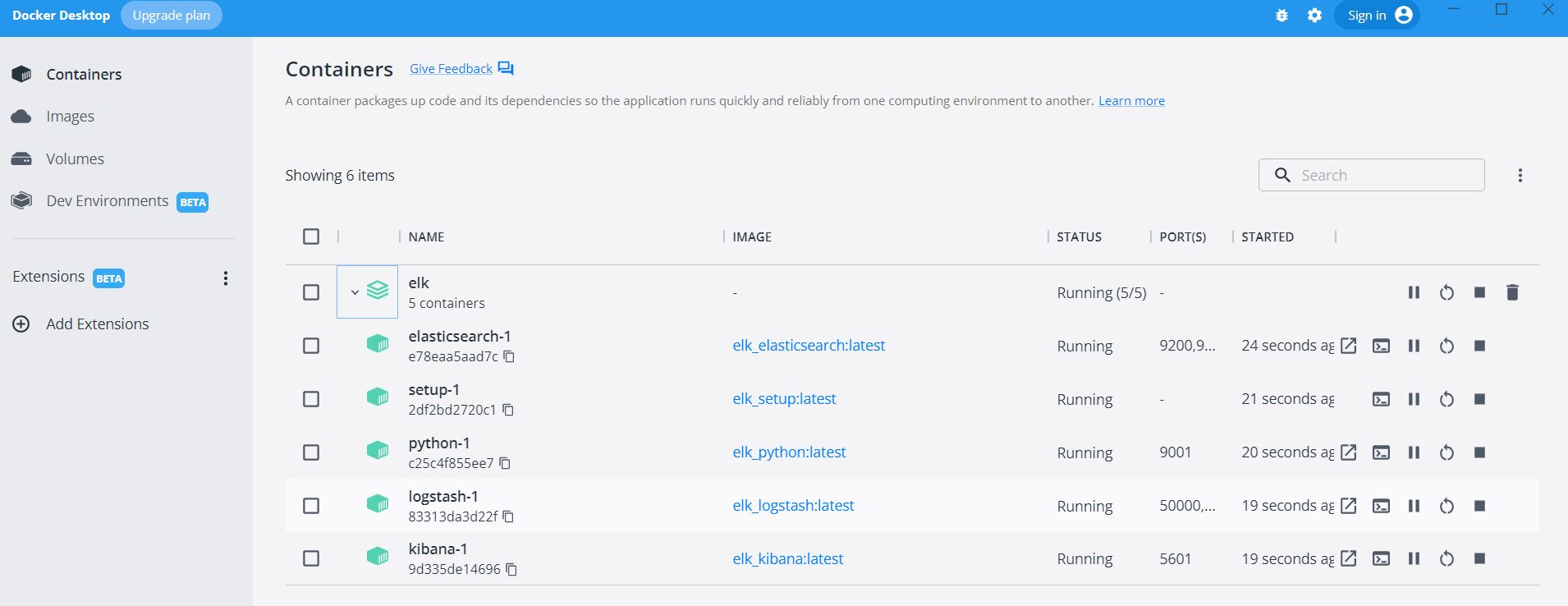
We just need to start the docker. Attaching the screenshot of executing docker installation file on windows 11.

1. Typing the cmd command from our docker file path location will open windows prompt on that path
2. We can then start docker with command ***docker-compose up -d***

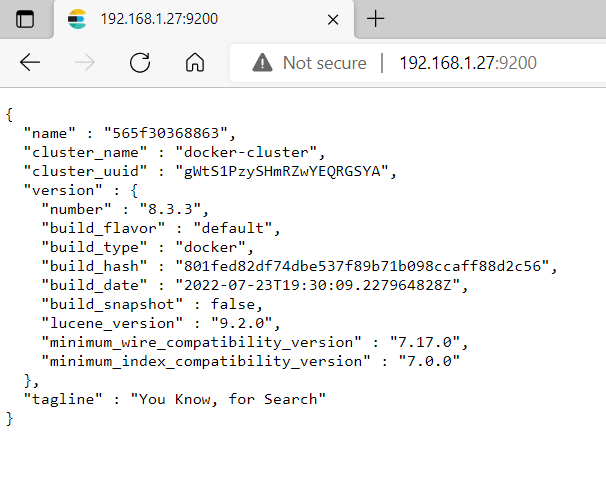


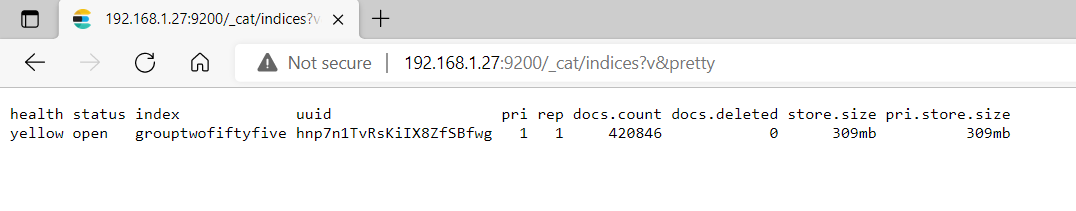


1. Once the ELK stack is up, we can see them in docker hub and explore it via console/UI ports too Docker hub

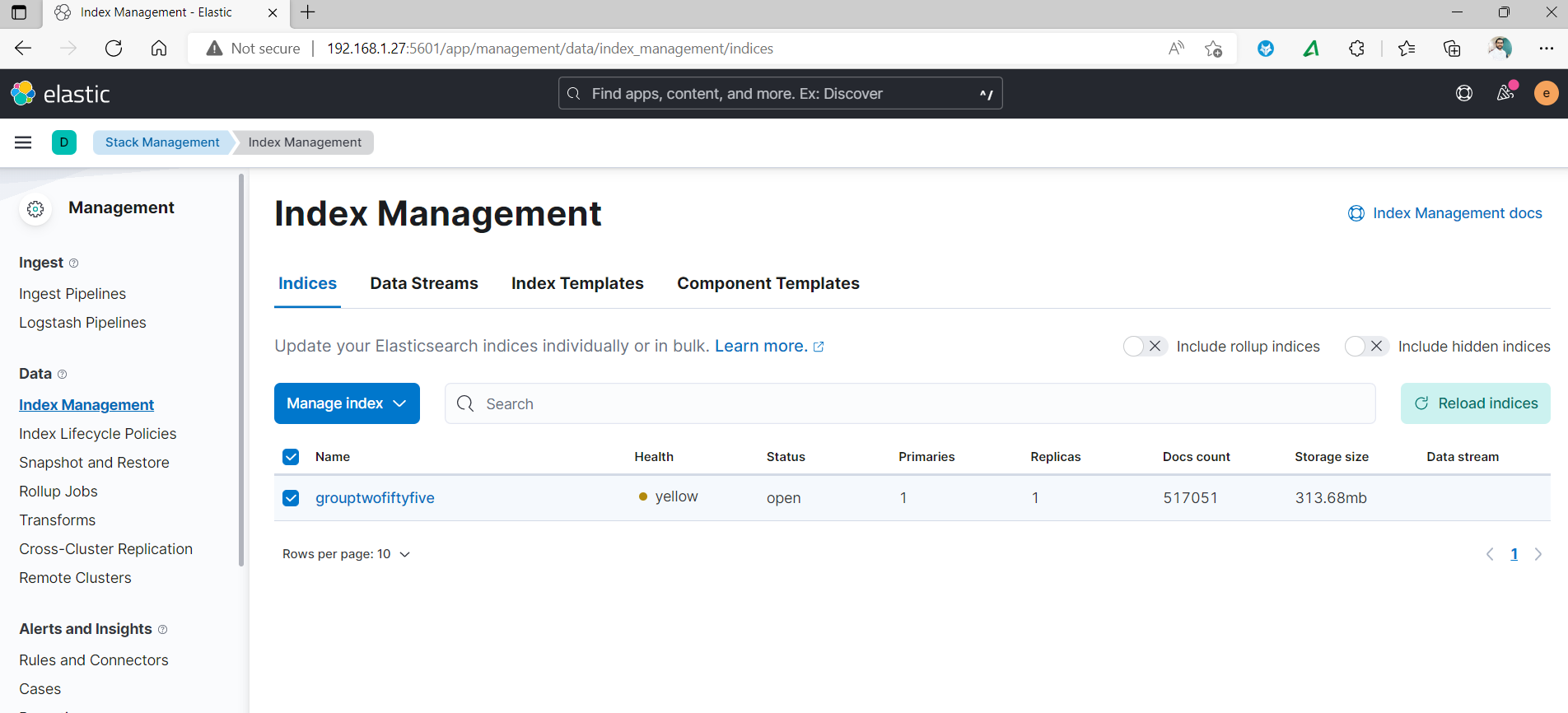


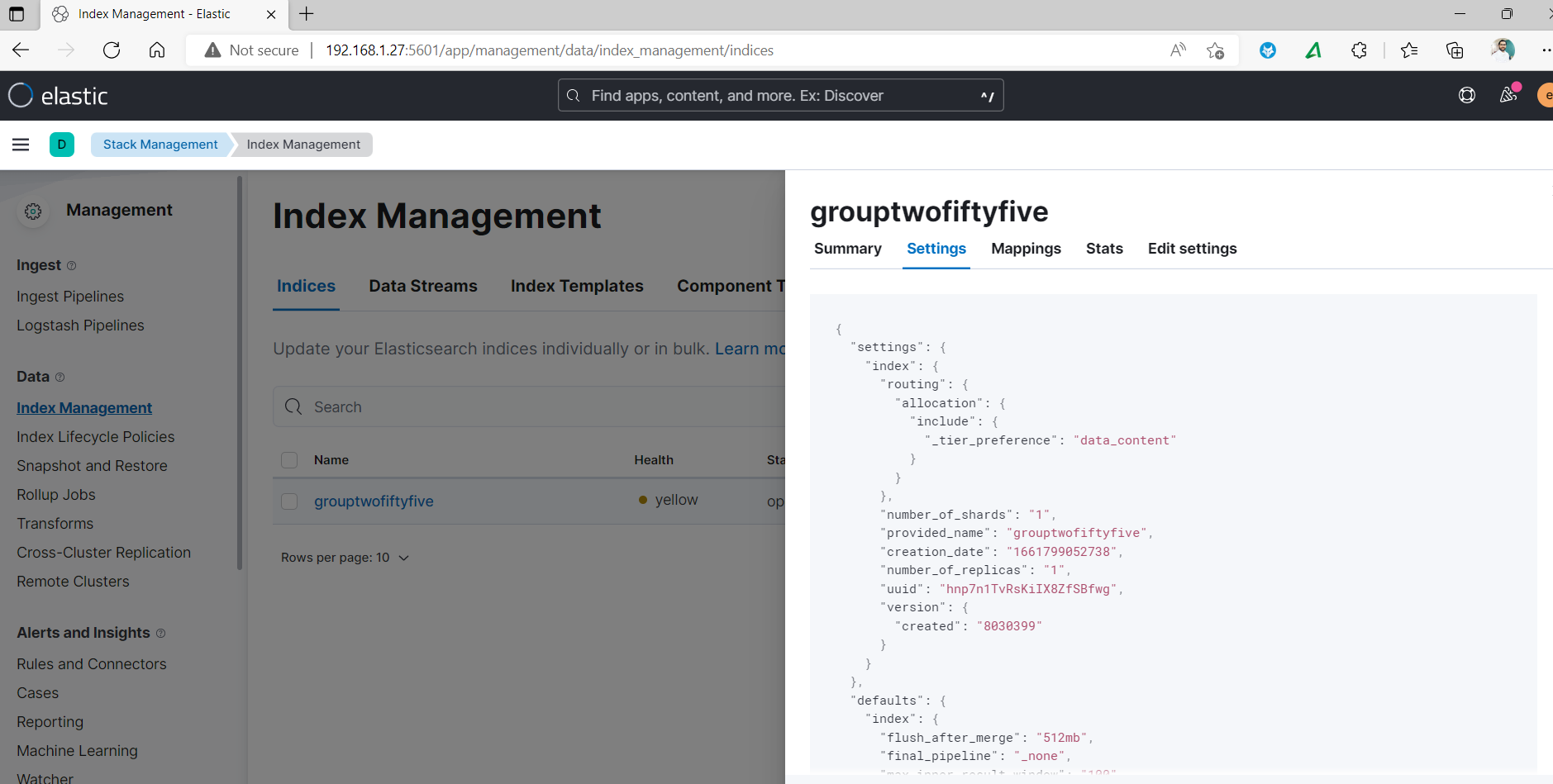
We will use the default username (elastic) and password (changeme) for Elasticsearch/Kibana while accessing it on 9200/5601 ports respectively. We can see Elasticsearch running and also the index “**grouptwofiftyfive**” getting created.





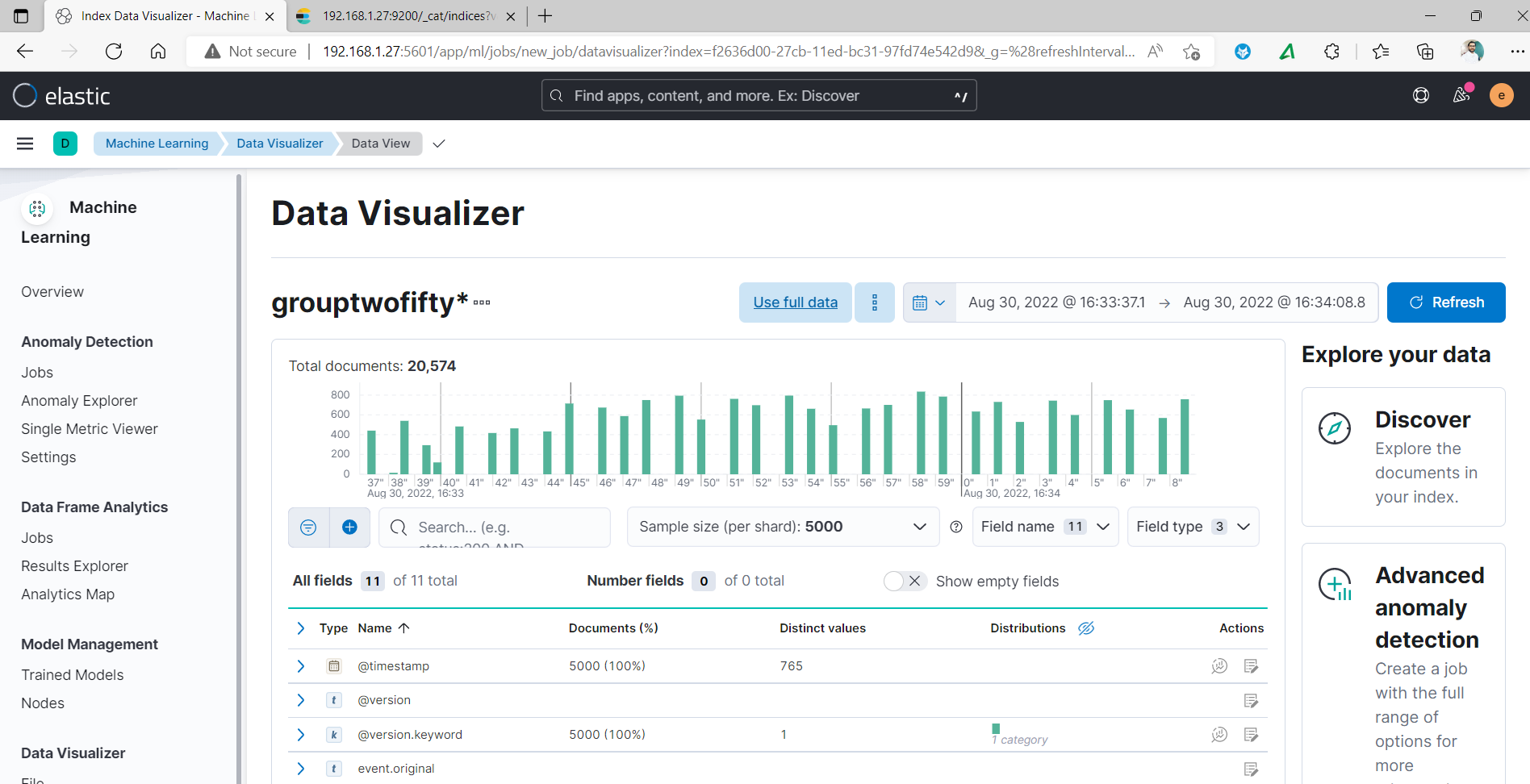
1. In Kibana, search for Index Management and we will see the entries for “**grouptwofiftyfive**” index that we generated using faker python file even before ELK installation.



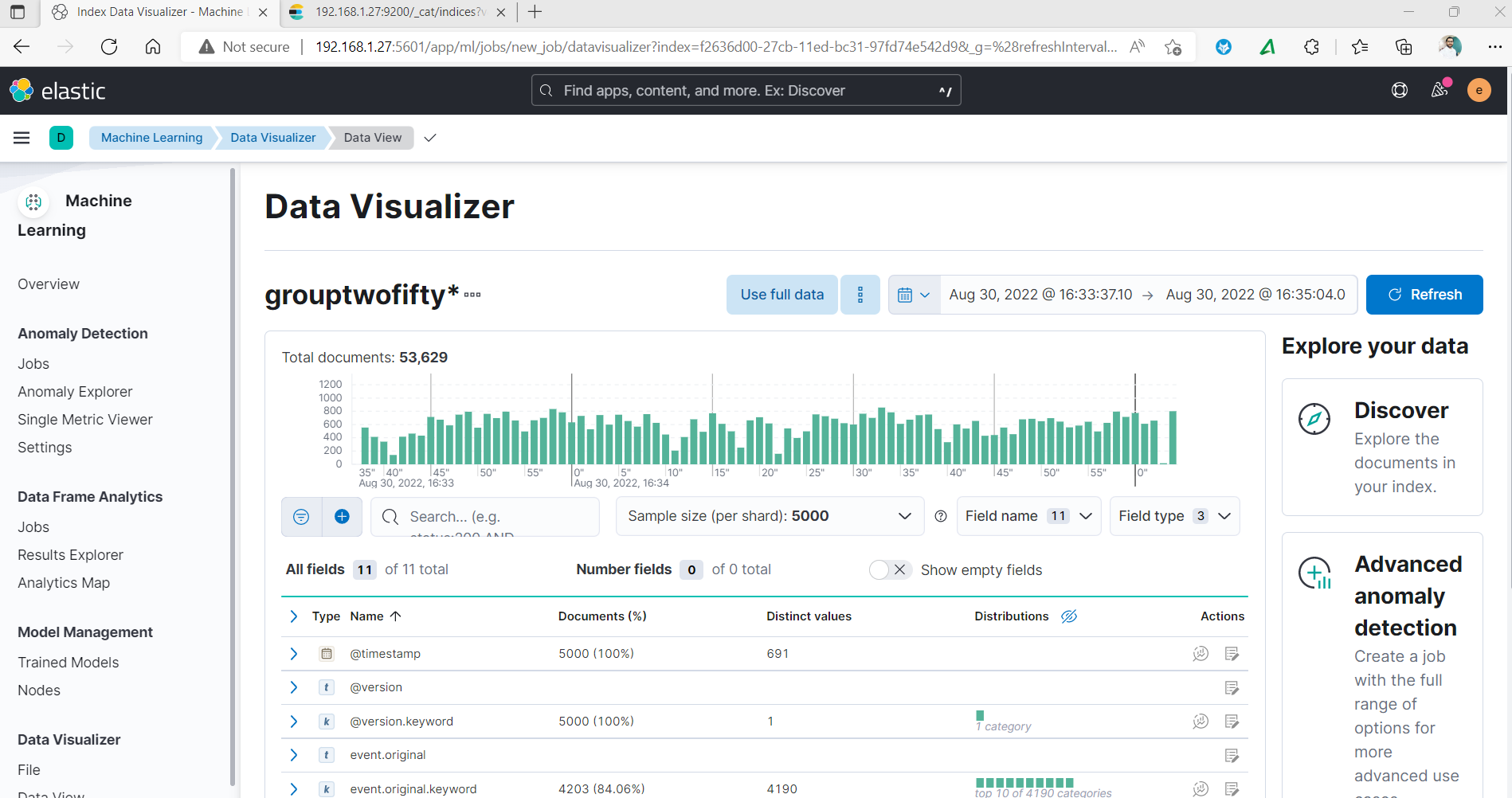


1. We will go to data Visualization and see how our logs data looks now.

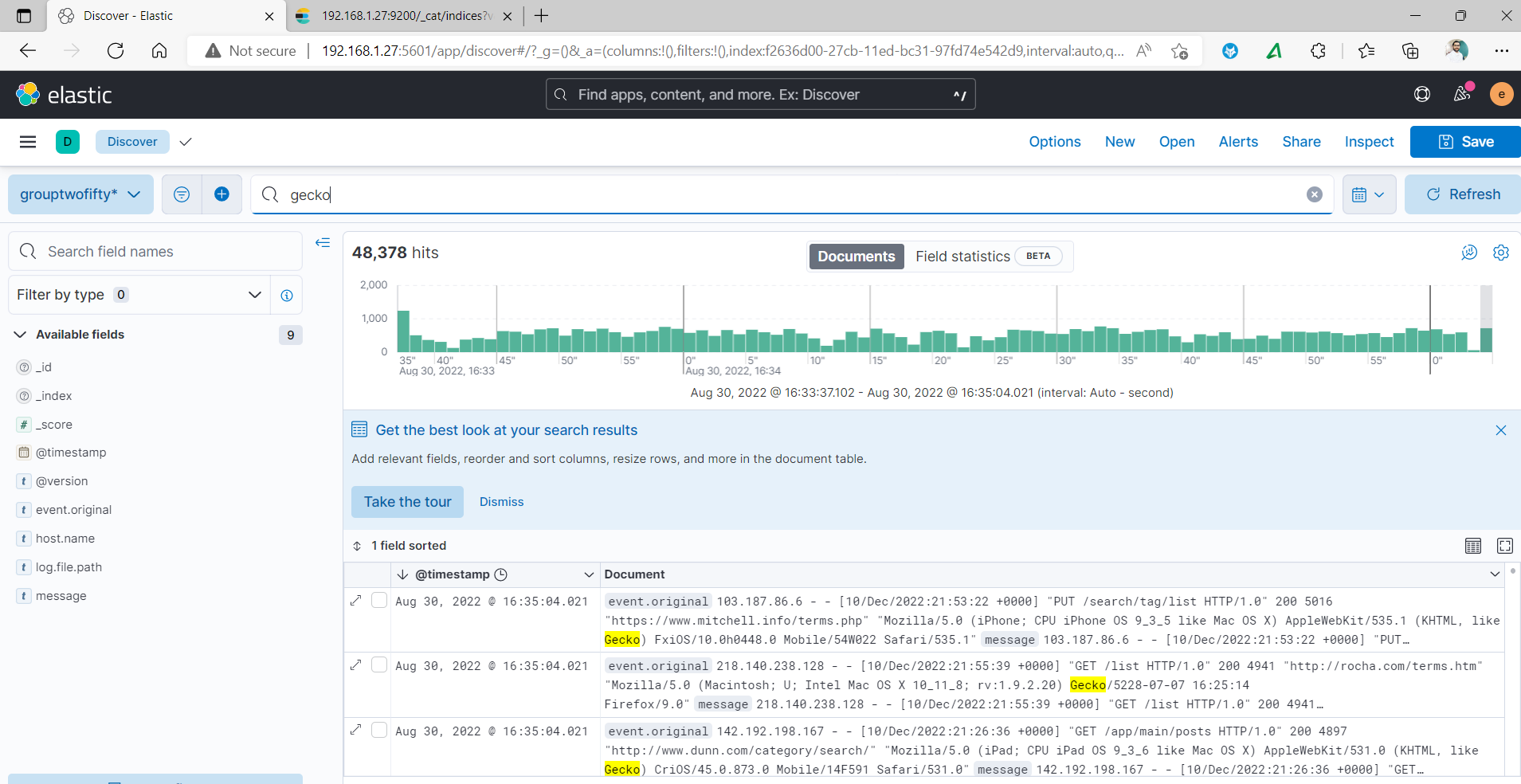
We can now see initial view of our data in Kibana as well.



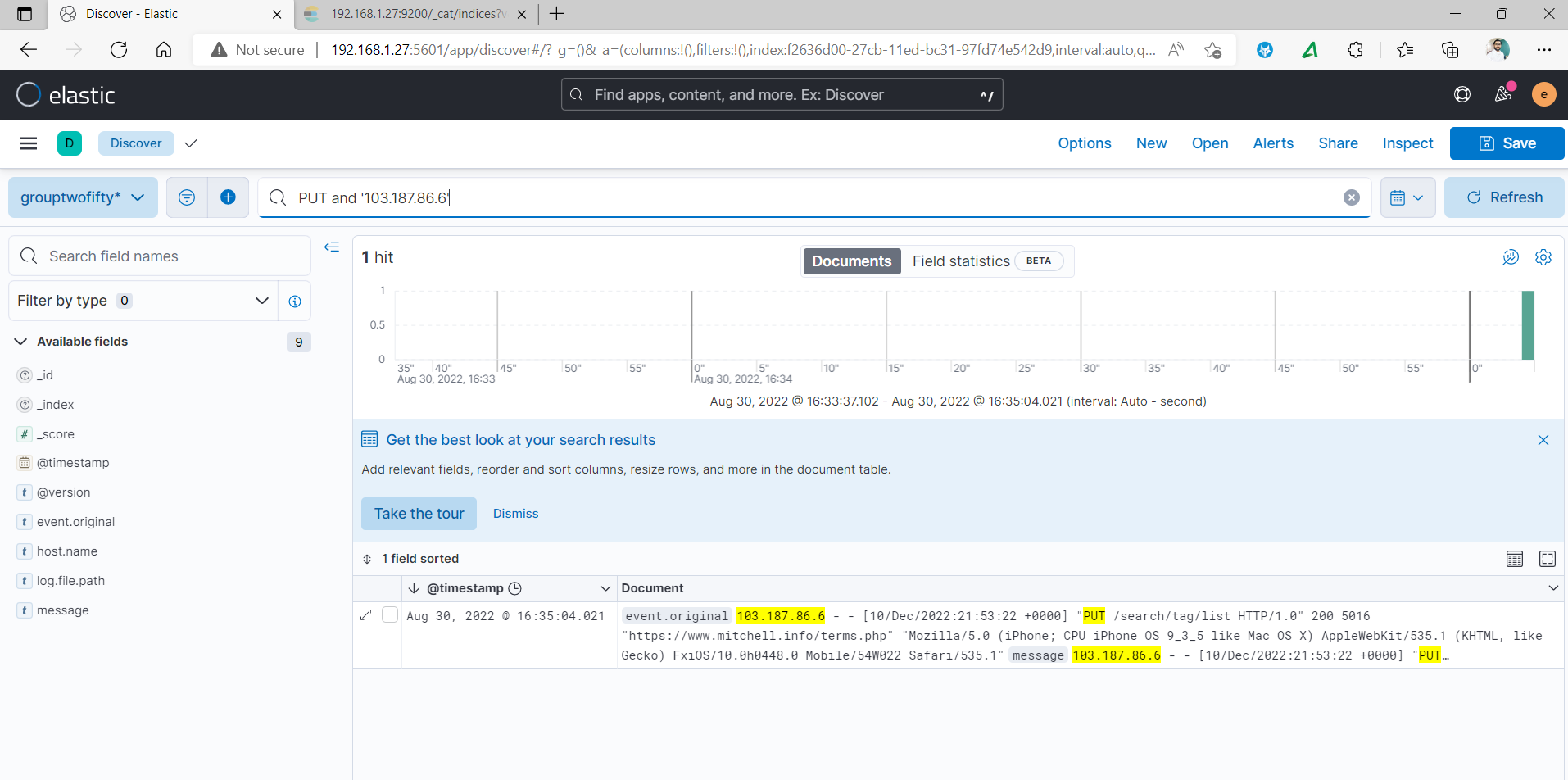
After 1 Min you can see the logs document are more below which is 53629



We can query strings from kibana to search log for us. Let’s search for Gecko.



Search for little complex query “PUT and '103.187.86.6'”



We can further process the logs before logging or use the keywords/rules base don our requirement to build alerts in kibana.