**POC – 3**

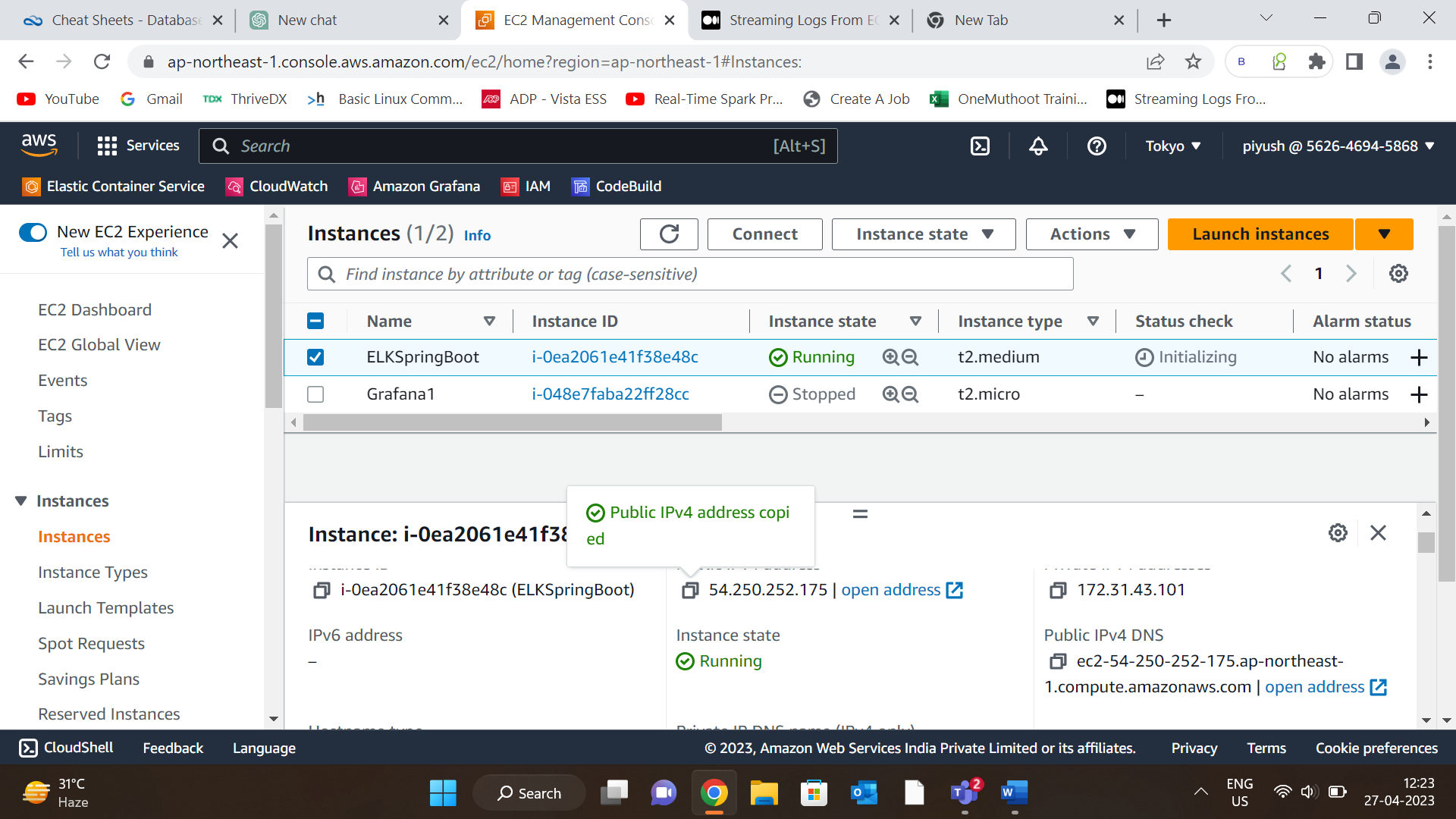
**My task is to containerized an Spring Boot application and streaming its logs to Elastic Search and view on Kibana.**

**CONCEPT EXPLAINED :**

With **Fluent bit** plugin for AWS container image, we can route logs to Amazon Cloud Watch and any other destinations like **Elasticsearch Service**.

I have used a new log driver for ECS task where we can deploy a Fluent Bit sidecar with the task and route logs to it. Using AWS FireLens, we can direct container logs to storage and analytics tools without modifying our deployment scripts. With a few configuration updates on AWS Fargate, select the destination and optionally define filters to instruct FireLens to send container logs to where they are needed.

First I have created an instance called ELKSpringBoot and in which I am using Ubuntu as AMI and we must have minimum 2 vCpu and 4GiB memory to run it smoothly as I am using t2.medium as an instance type. In this instance I have installed ELK. We have to configure Elastic Search and Kibana correctly before stating them.



Then, create an ECS Fargate Cluster 🡪 Create Task Definition

**If you want to see my TASK DEFINITION click on the link :**

[**https://github.com/Piyush-007jb/ELK-Json/blob/069f03511edcc6d0646c7b1fcc9c41d303c108a3/Spring-Firelens-revision5.json**](https://github.com/Piyush-007jb/ELK-Json/blob/069f03511edcc6d0646c7b1fcc9c41d303c108a3/Spring-Firelens-revision5.json)

**The task definition specifies two containers: log\_router and Spring-Boot.**

**log\_router** container uses the AWS for Fluent Bit image and awslogs log driver to send logs to Amazon CloudWatch Logs, with a log group /ecs/Spring-Firelens and a stream prefix firelens.

**The Spring-Boot** container uses an image from Amazon ECR and **awsfirelens log driver** to send logs to Elasticsearch. The container also maps two host ports (‘80’ and ‘8080’) to two container ports (’80’ and ‘8080’), respectively.

The task definition requires the FARGATE launch type, with 1024 CPU units and 3072 MB of memory. It also specifies an execution role with full access to Amazon Elasticsearch. The compatibility mode for the task is FARGATE, indicating that the task is intended to run on the AWS Fargate container platform.

In Spring Container log configuration I have define some parameters and we add more if we want to 🡪

**"Host":** The IP address of the endpoint where Elasticsearch is running.

**"Include\_Tag\_Key": "true"** (It is a boolean value indicating whether to include the tag key in the log record. If set to true, a tag with the key "tags" will be added to each log record.)

**"Index":** The name of the index in Elasticsearch where logs will be stored.

**"Name": "es"** (The name of the Elasticsearch domain where logs will be stored.)

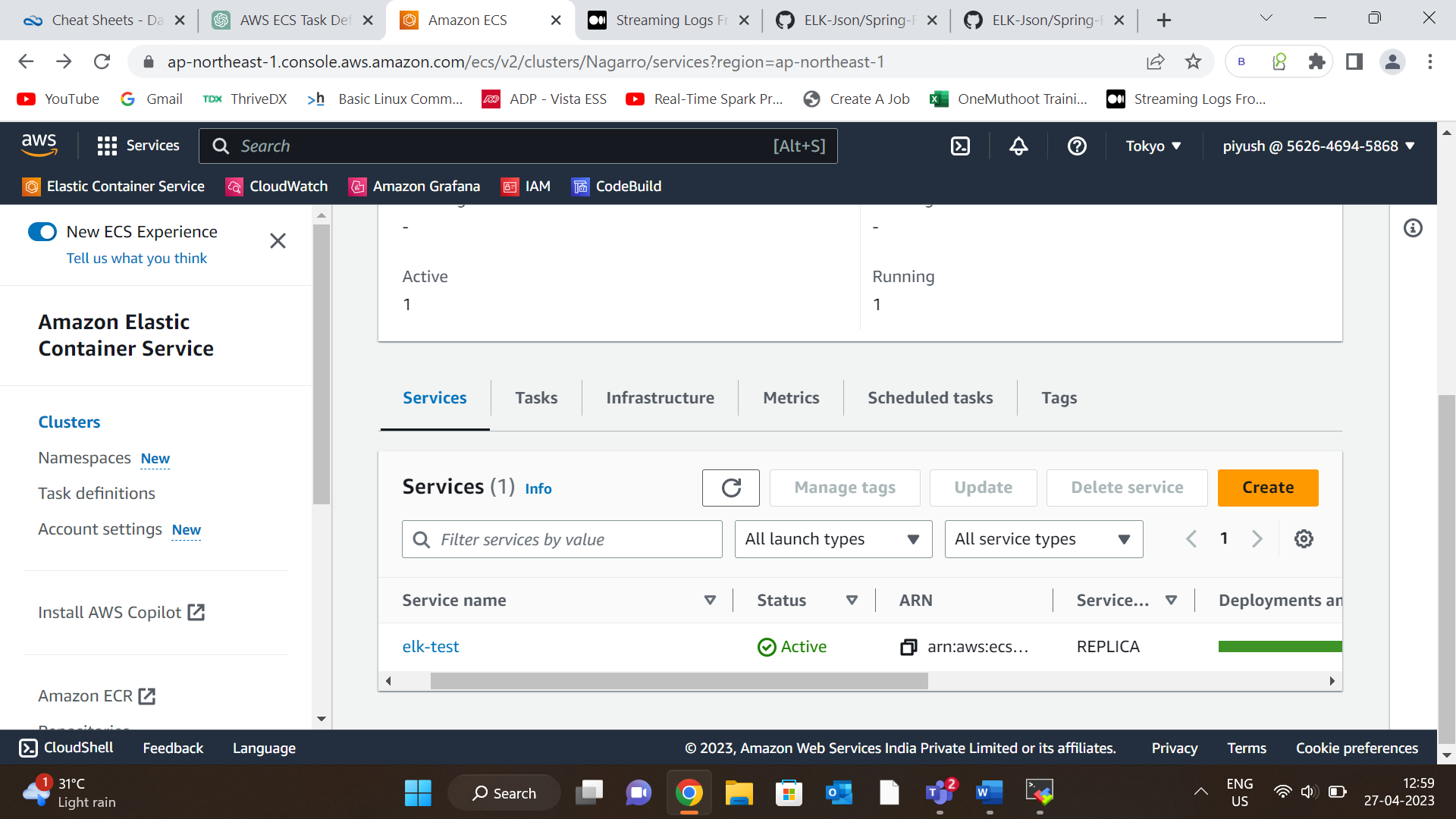
**"Port":** The port number of the Elasticsearch endpoint.

**"Suppress\_Type\_Name":** **"On" (**A boolean value indicating whether to suppress the type name in the log record. If set to "On", the type name will be removed from the log record**.)**

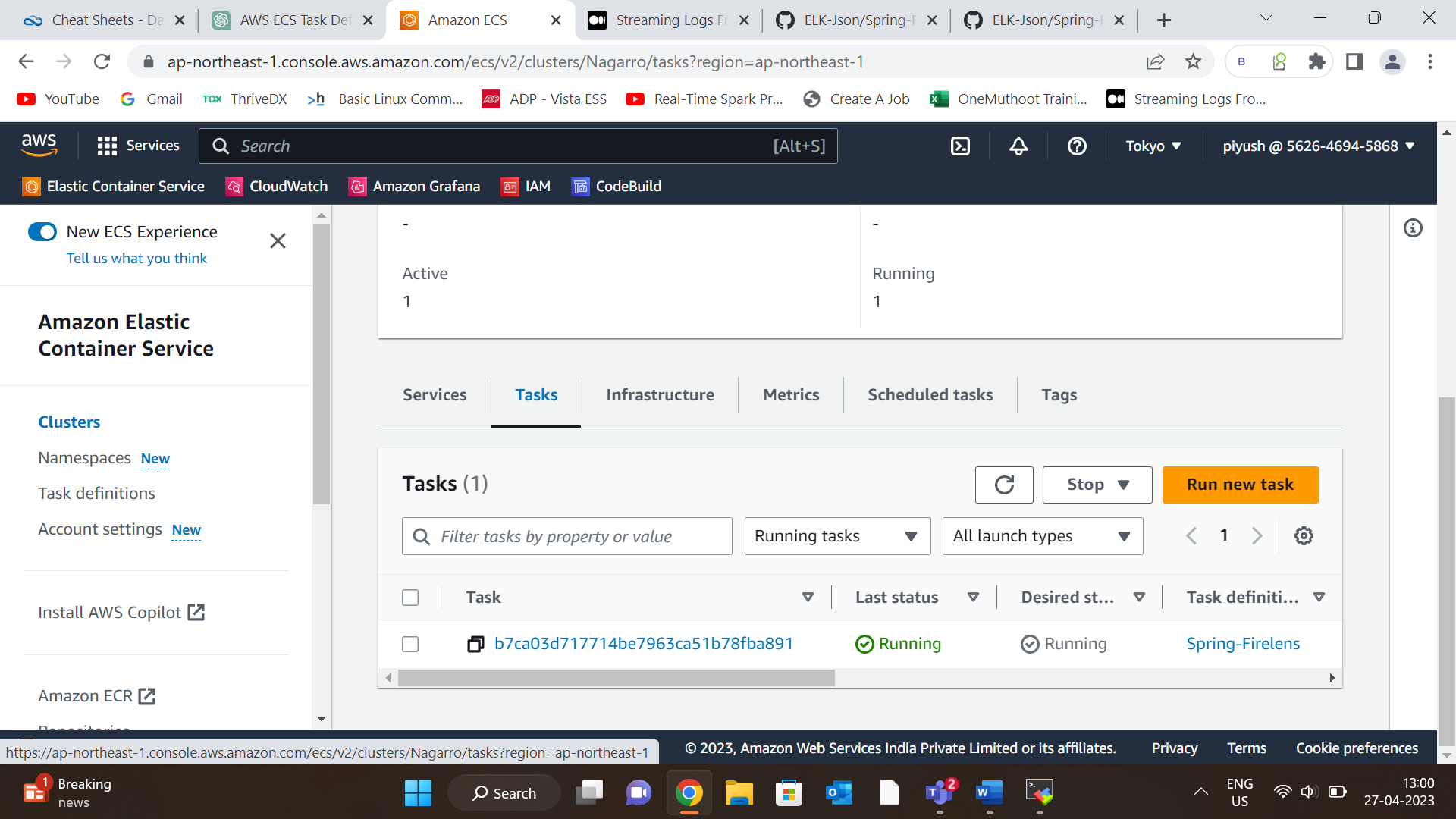
**"Tag\_Key":** **"tags" (**The name of the tag key to use for the logs. If not specified, it defaults to "tags".**)**

**After registering the task definition I have created a service using the task definition I have created with a task count 1 in the Cluster .**

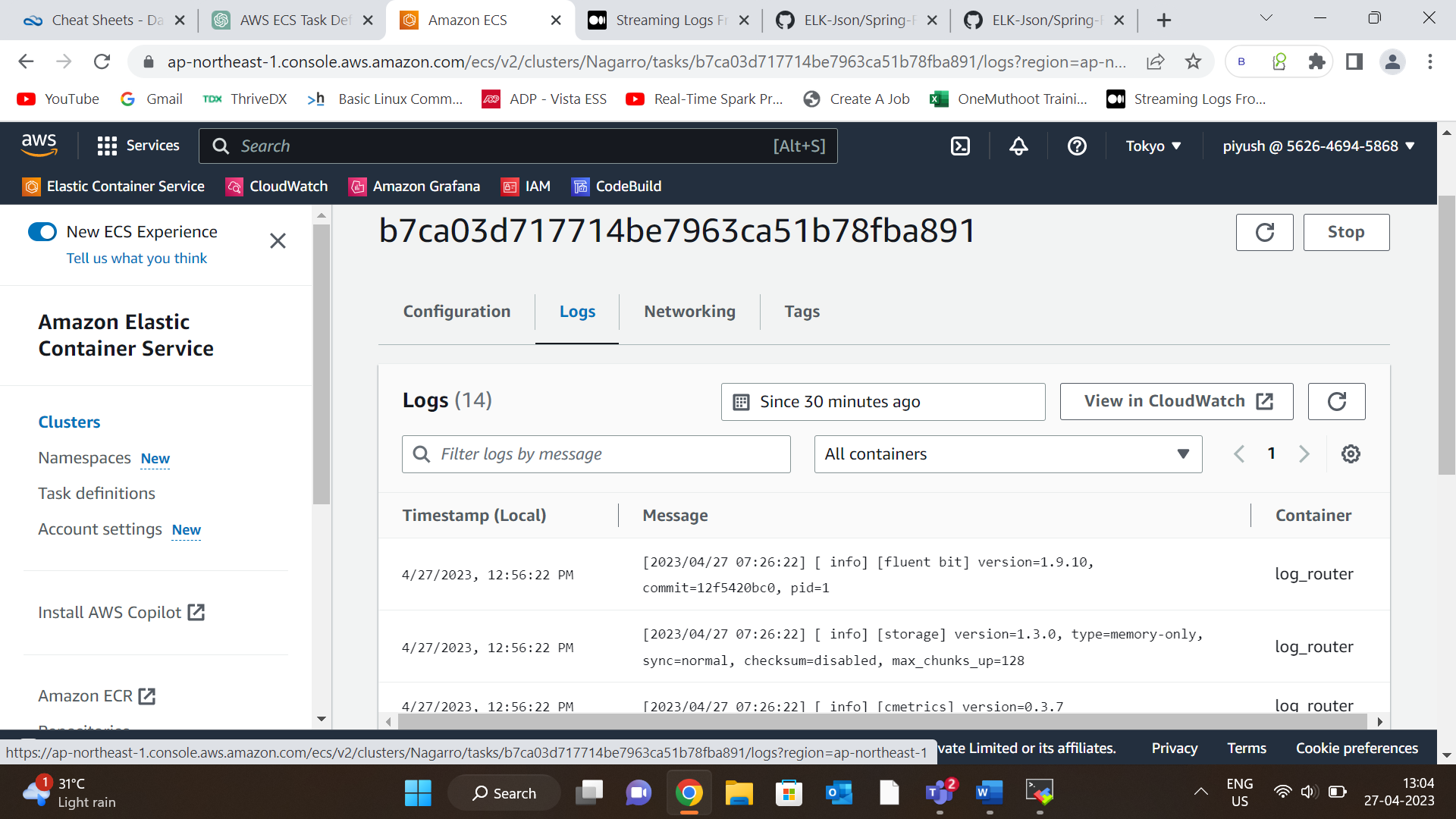
Before running the service you must confirm that the EC2 instance and the container you are going to run is in the same VPC and subnet.



**Here you can see service status is active and deployment is successful and One task is running** .



**After clicking on Task We can also check the logs and see if there is any error or not.**



**After that we conform from the browser if the test App is accessible** by using the public Ip assigned to the task

Please note that you must have allowed permission to the port on Security group attached to the service.

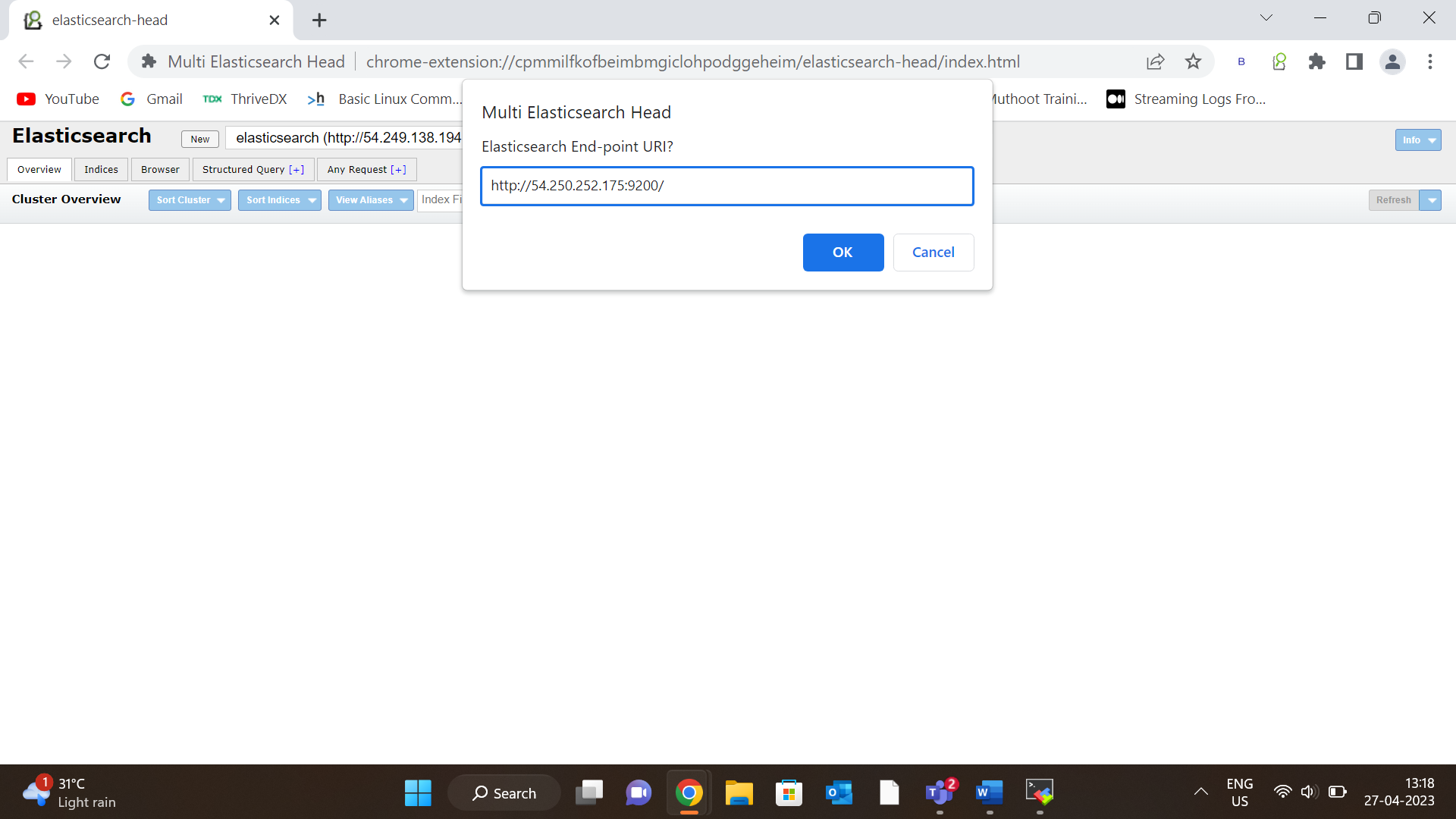
Graphical user interface, text, application, email

Description automatically generated

**Confirm the application logs in Elastic Search**

Once all the steps above are completed , we will start getting our Spring Boot application logs in elastic search.

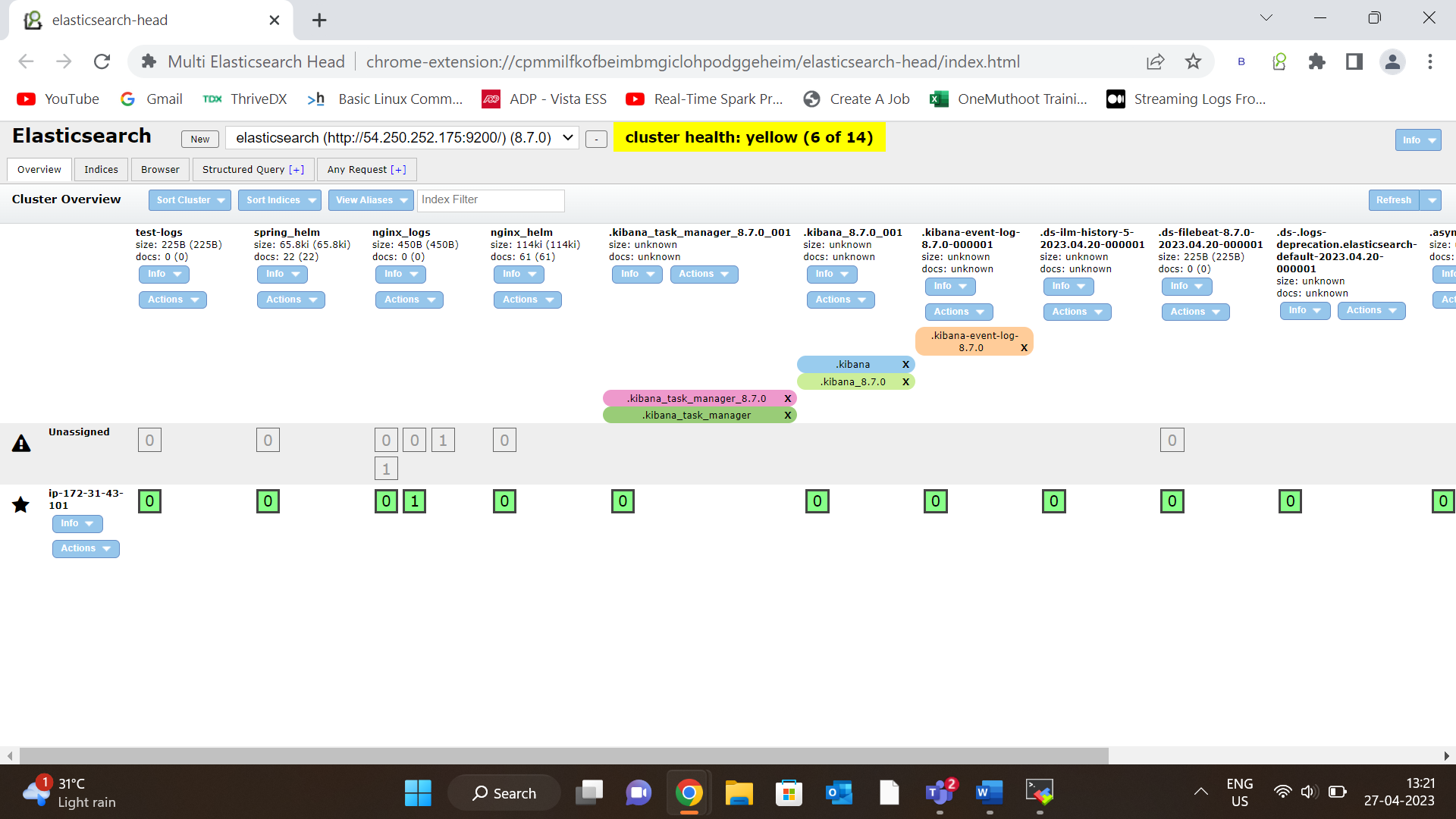
To confirm if my application data gets to Elastic search I have used a chrome extension called [ElasticSearch Head](https://chrome.google.com/webstore/detail/elasticsearch-head/ffmkiejjmecolpfloofpjologoblkegm" \t "_blank). Add the extension to chrome and access the endpoint of your elastic search .



**Click ok and you will se the indices that you have given in Task definition.**

**I have given spring\_helm as you see in the Elastic Search.**

**NOTE: If your logs are not ingested to Elastic Search you will not be able to see index , if it is visible to you then it is working properly.**

 **View logs in Kibana 🡪**

**First we go to kibana URL . To do that we enter our host IP and port no.(5601) in the browser.**

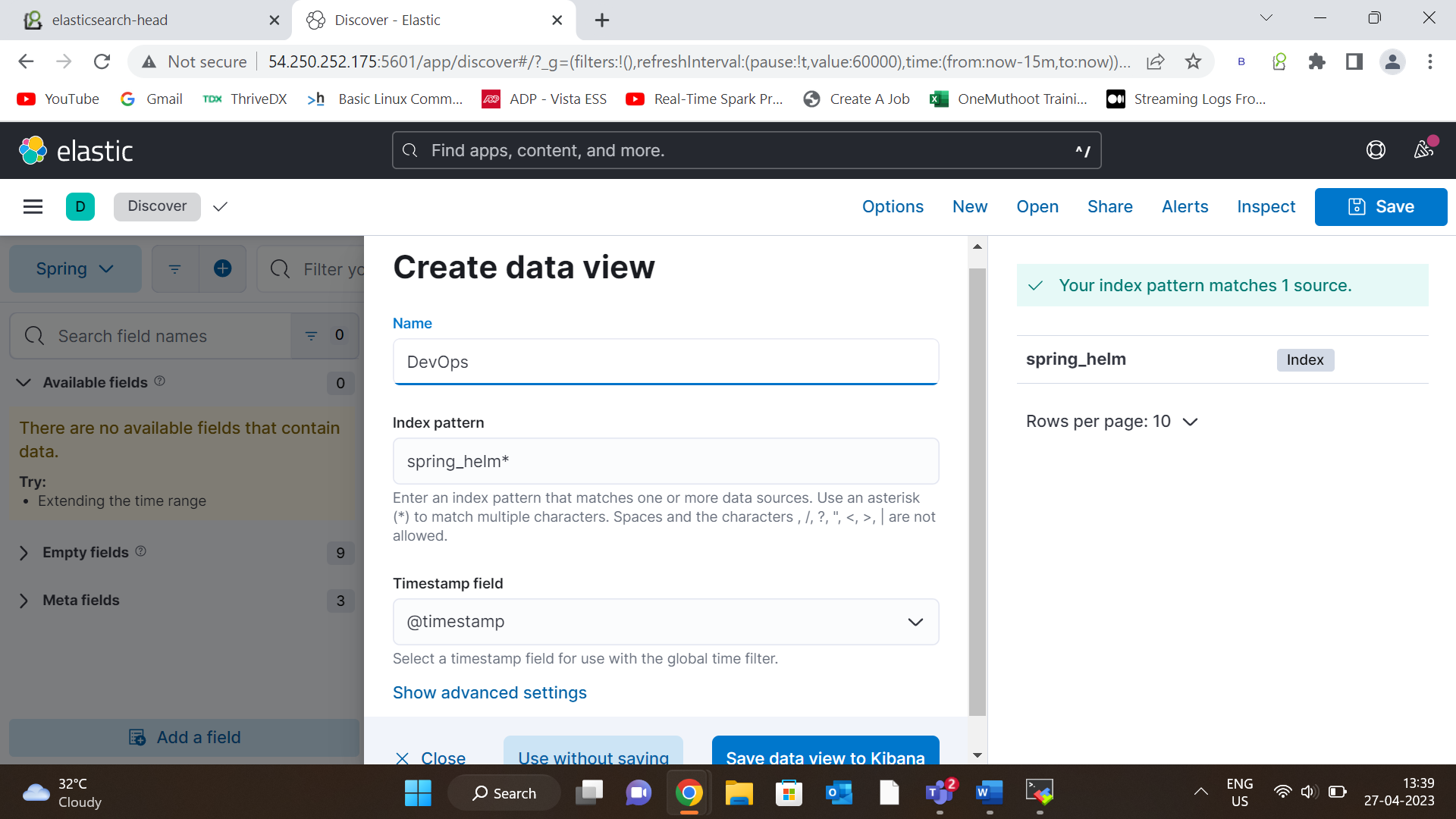
**Then, go to Analytic section 🡪 Discover .**

**After Discover, you have to create a data view (previously named index pattern in previous versions).**

Then we give name and index pattern that we confirmed on elastic search in the previous slides.

Graphical user interface, application, website

Description automatically generated



**After clicking on save data view to Kibana I am able to see the logs of my Spring-Boot application in Kibana .**

Graphical user interface, text, email

Description automatically generated

**We can also select any field that we want to see as I selected container\_name.keyword and it is showing my container name that is Sprng-Boot also we can add fields more fields in it.**

Graphical user interface, text, email

Description automatically generated