Elasticstack

Grundlagen

Joachim Walther

Elasticstack

Elastic Agent läßt sich auf verschiedene Arten betreiben:

1. Docker Container
2. Installiertes Programm
3. Mit lokaler Elasticsearch Installation
4. Mit Elastic Cloud

Wir wählen die Konfiguration Docker Container <-> Elastic Cloud.

Elasticstack

Image laden:

docker pull docker.elastic.co/beats/elastic-agent:8.2.2-arm64

Elasticstack

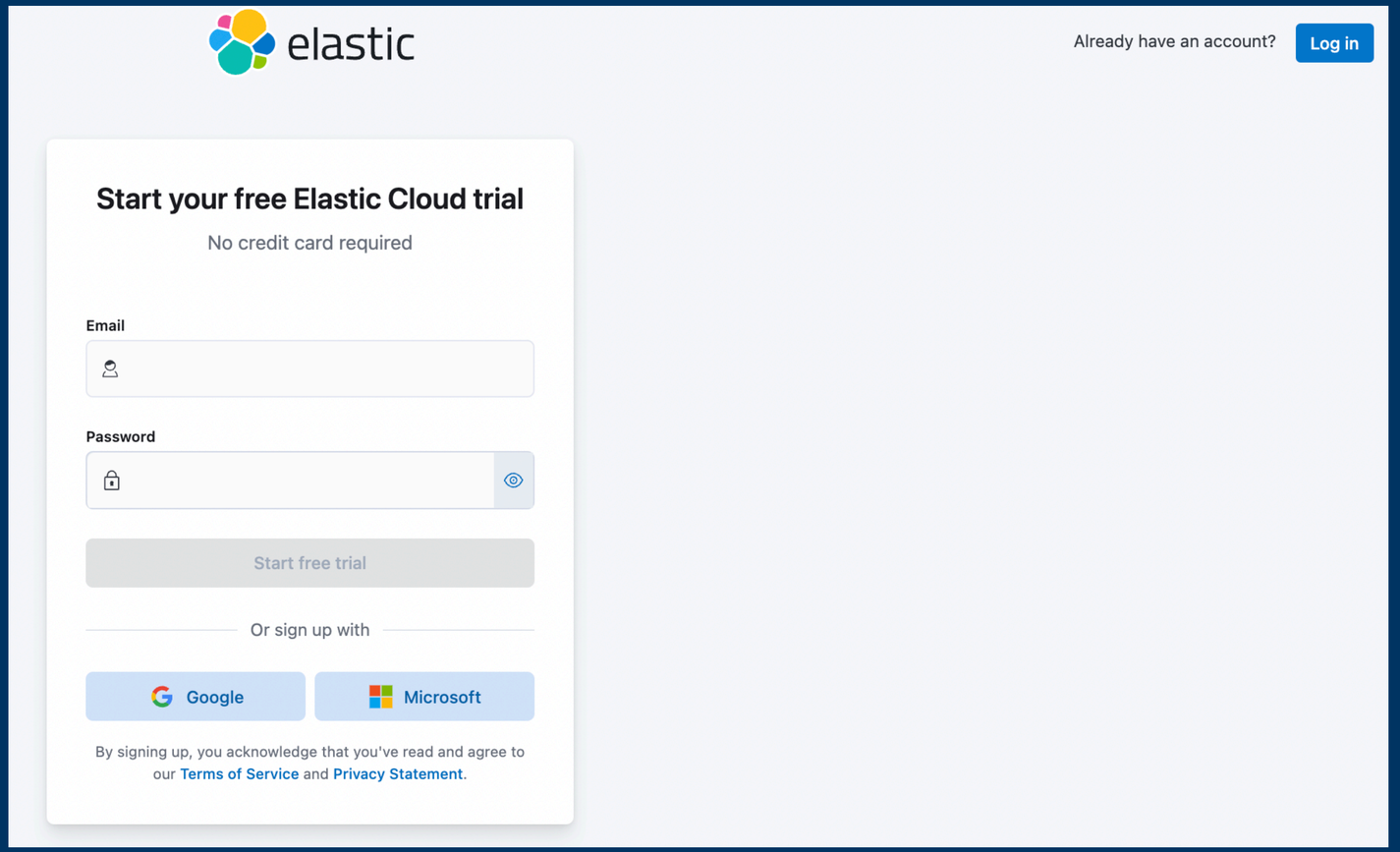
Container starten:

```
docker run \
  --env FLEET_ENROLL=1 \
  --env FLEET_URL={fleet-server-host-url} \
  --env FLEET_ENROLLMENT_TOKEN={enrollment-token} \
  --rm docker.elastic.co/beats/elastic-agent:8.2.2-arm64
```

Bevor wir dieses Kommando geben können, müssen wir auf Elastic Cloud ein Konto anlegen.

Elasticstack

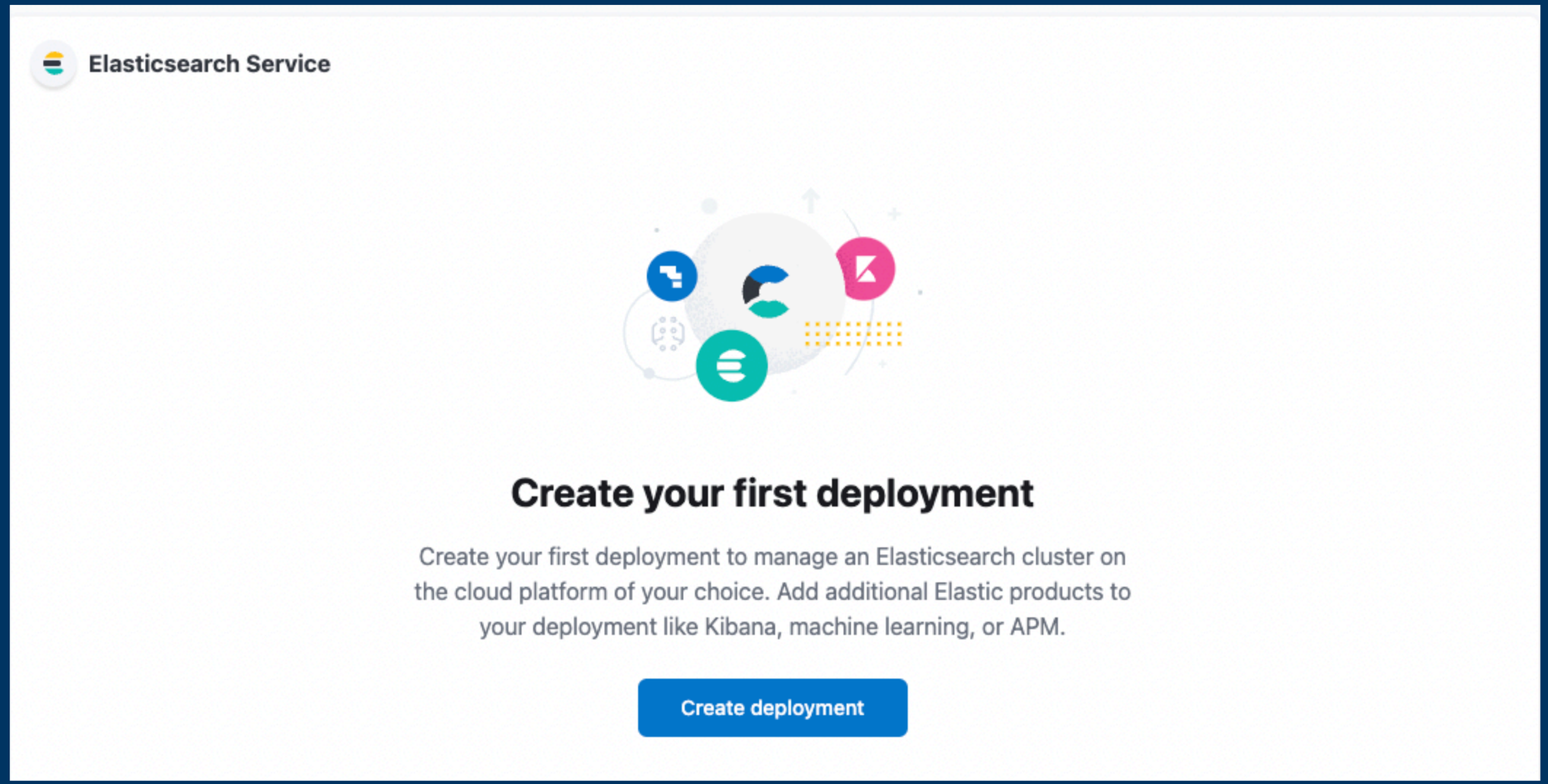
https://cloud.elastic.co/registration



Oder wenn man schon registriert ist, kann man rechts oben den Dialog wechseln und sich anmelden.
Wir nutzen den 30 Tage Erprobungszeitraum.

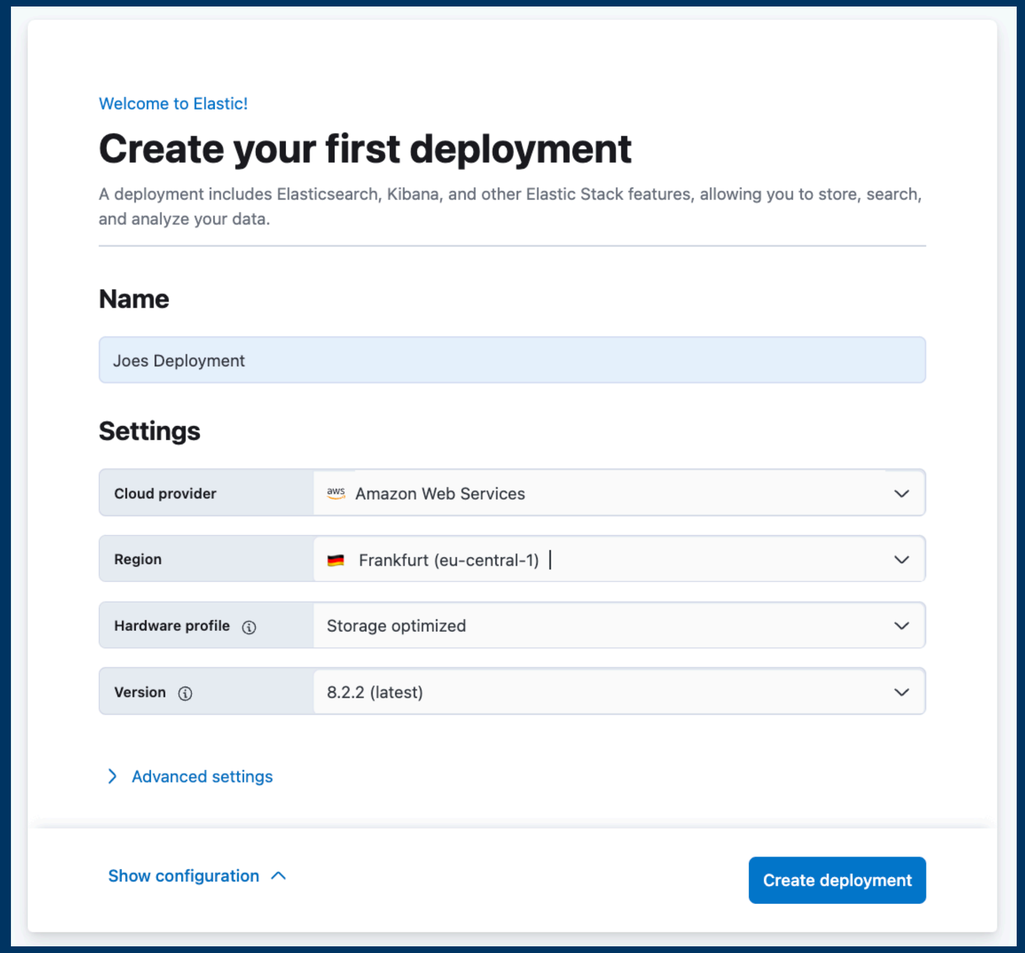
Elasticstack

Mit „create deployment“ fortfahren:



Elasticstack

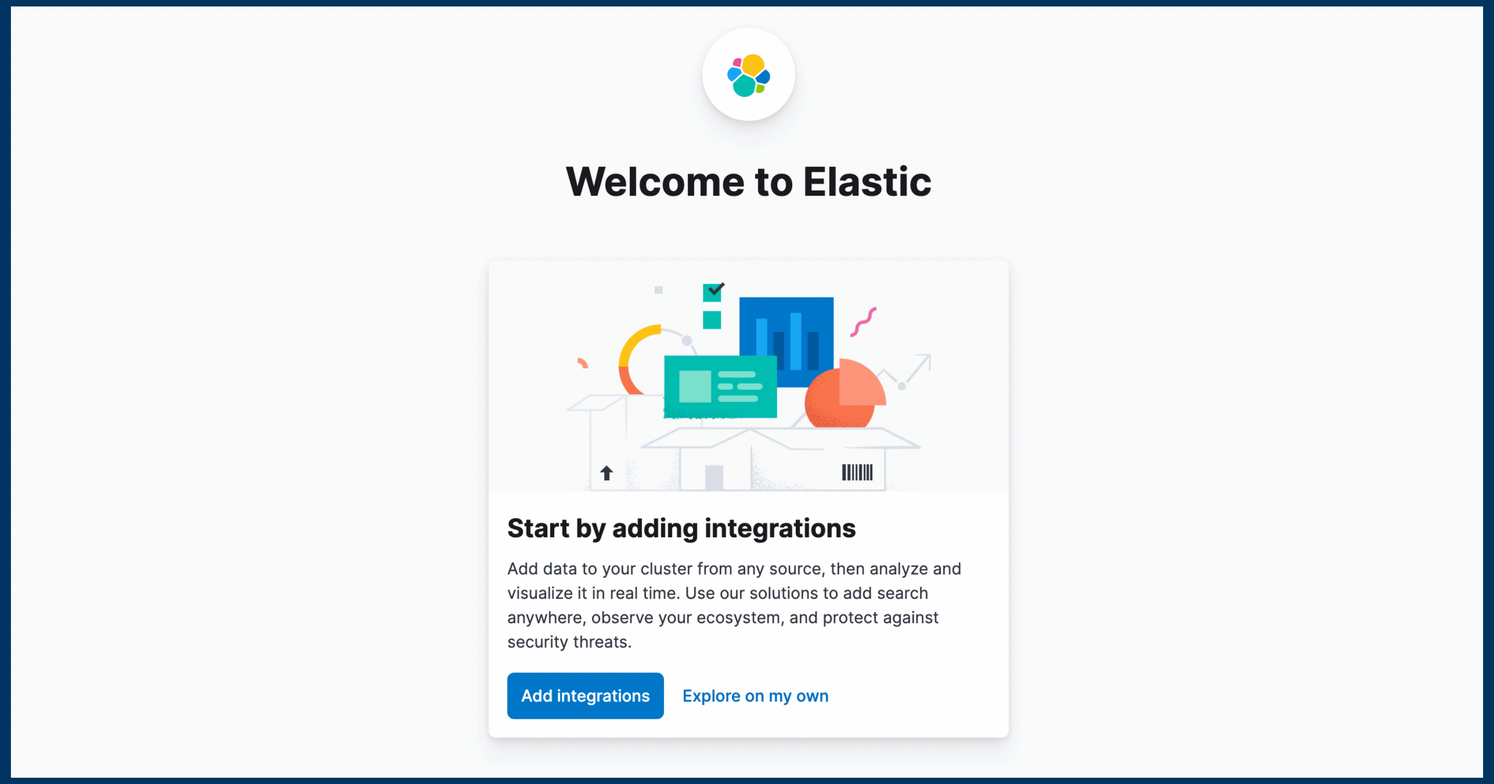
Mit Amazon bleiben wir in Deutschland ;-)



Credentials abspeichern, man weiß ja nie …

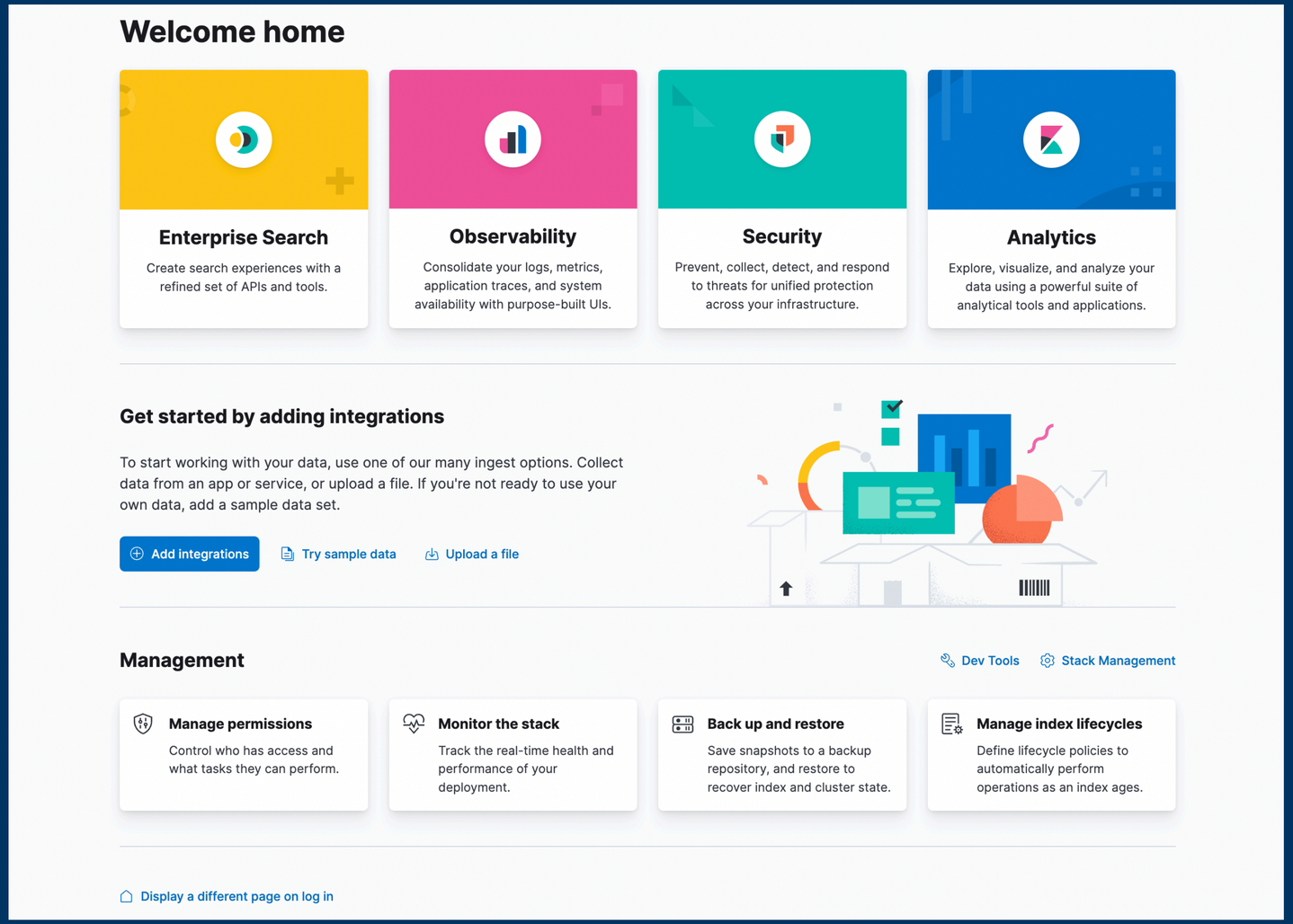
Elasticstack

Mit „Explore on my own“ geht’s weiter



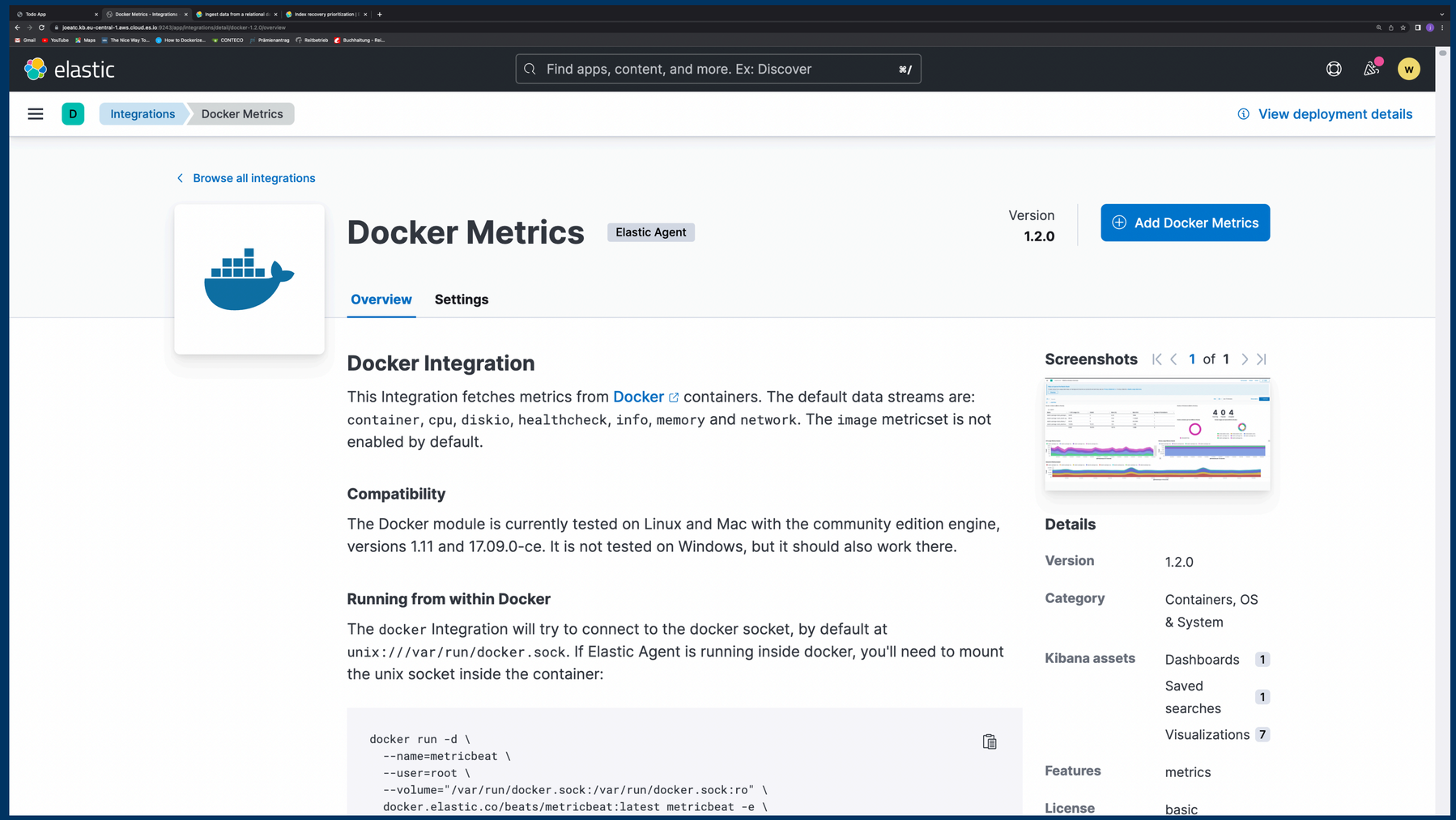
Elasticstack

Damit sind wir auf einer bekannten Seite. Wir nehmen die Voreinstellung, fahren aber dieses mal mit „Add integrations“ fort.



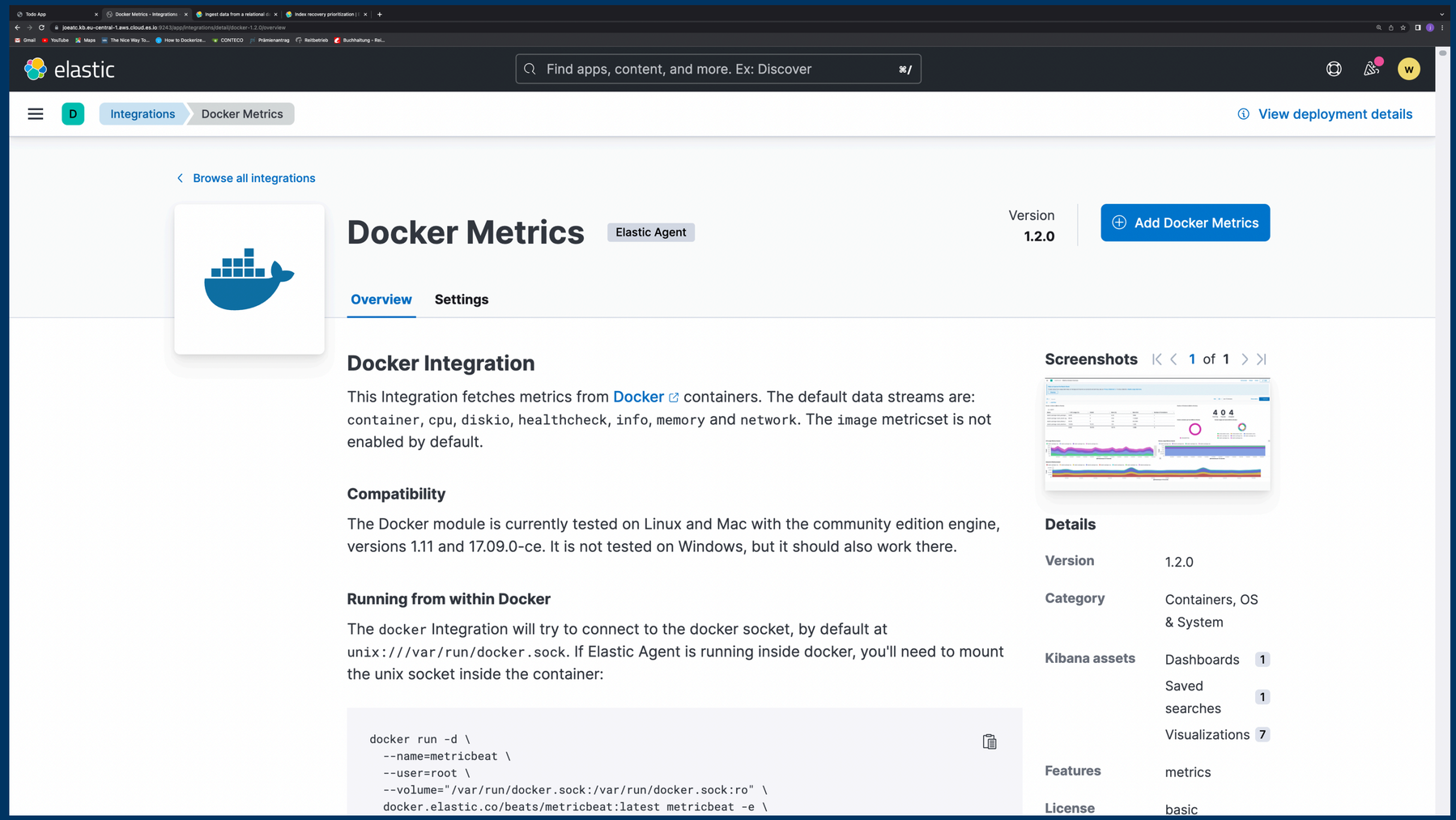
Elasticstack

Über die Auswahl „Containers“ und „Docker Metrics“ sind wir hier



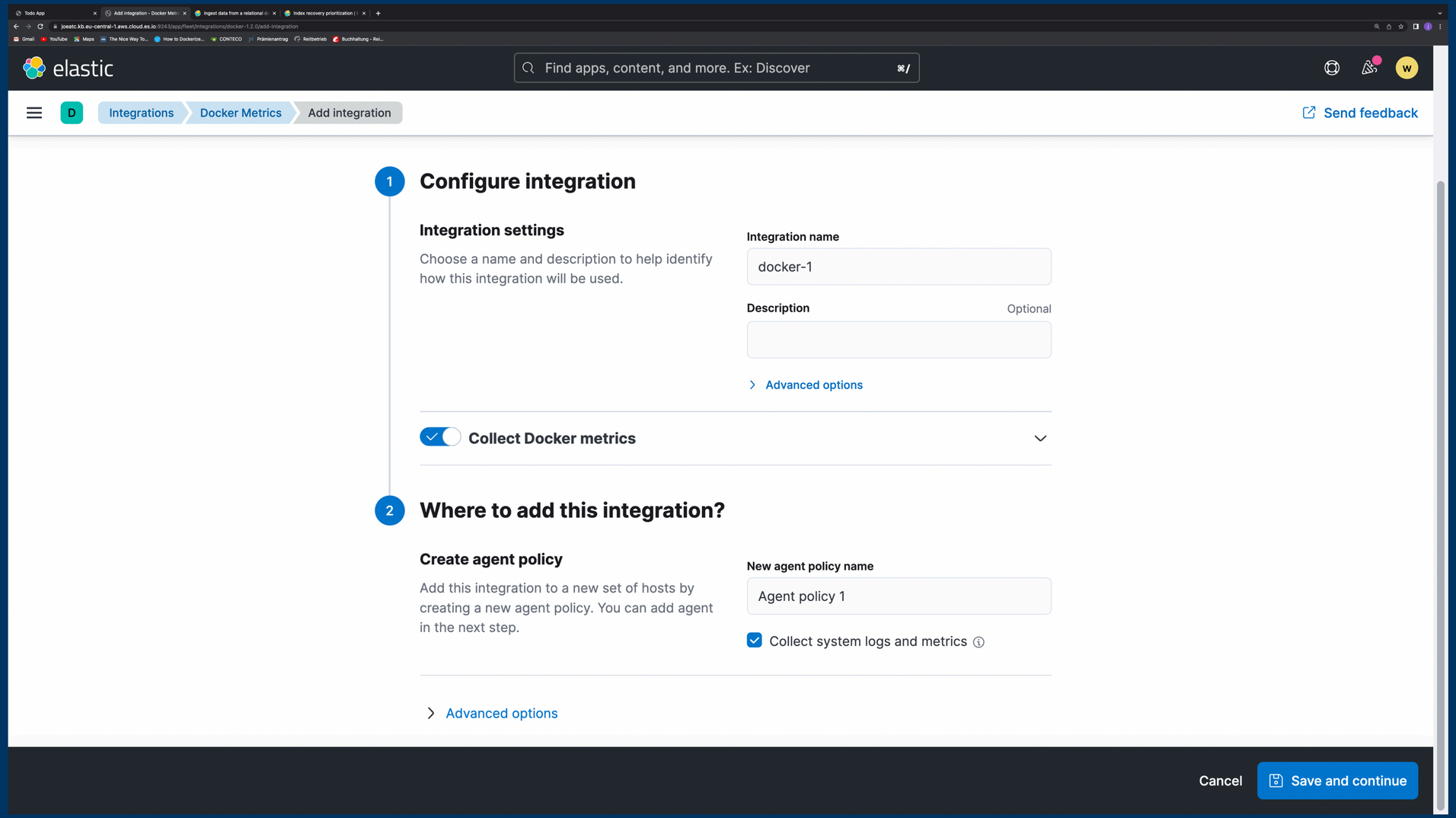
Elasticstack

Über die Auswahl „Containers“ und „Docker Metrics“ sind wir hier



Elasticstack

und über Add integration sind wir hier, die Voreinstellungen mit „save and continue“ übernehmen.



Elasticstack

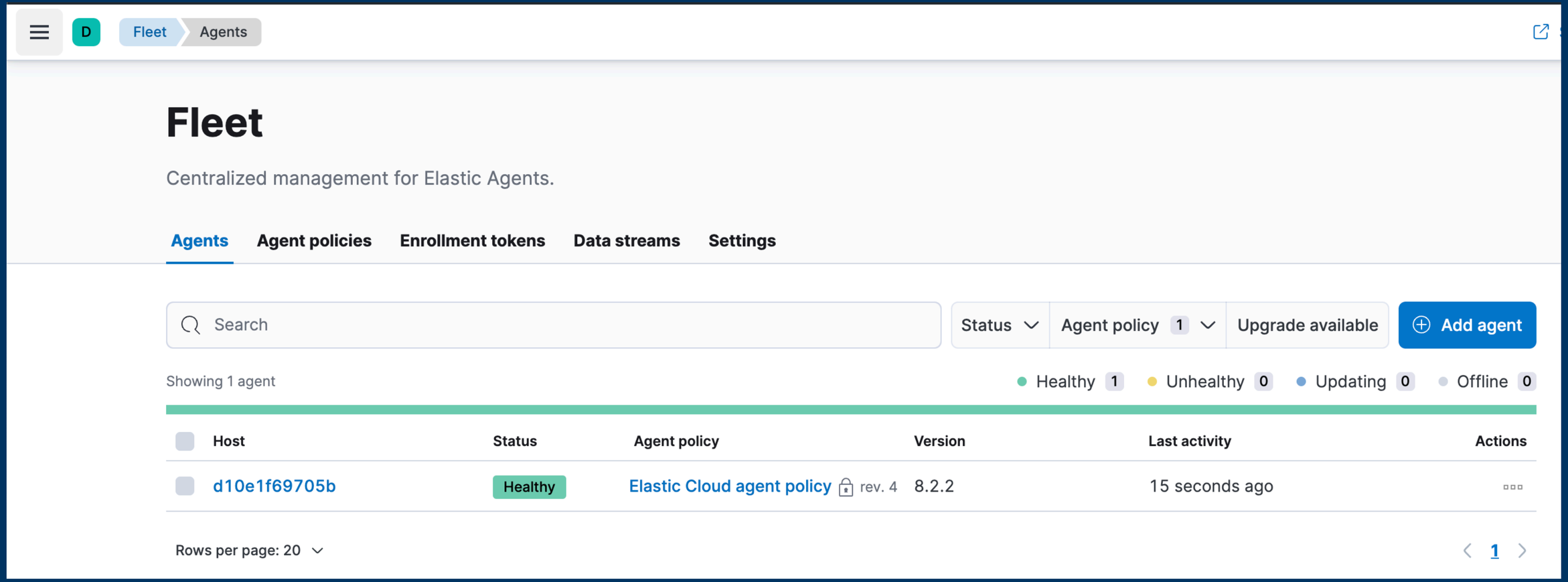
An dieser Stelle können wir nicht mit den Voreinstellungen weiter machen, da wir aufgrund unserer Technik mit Docker Containern arbeiten müssen.

Daher verlassen wir diesen Dialog mit Close.

Über das Hamburger Symbol links oben gehen wir im Abschnitt Management zu Fleet.

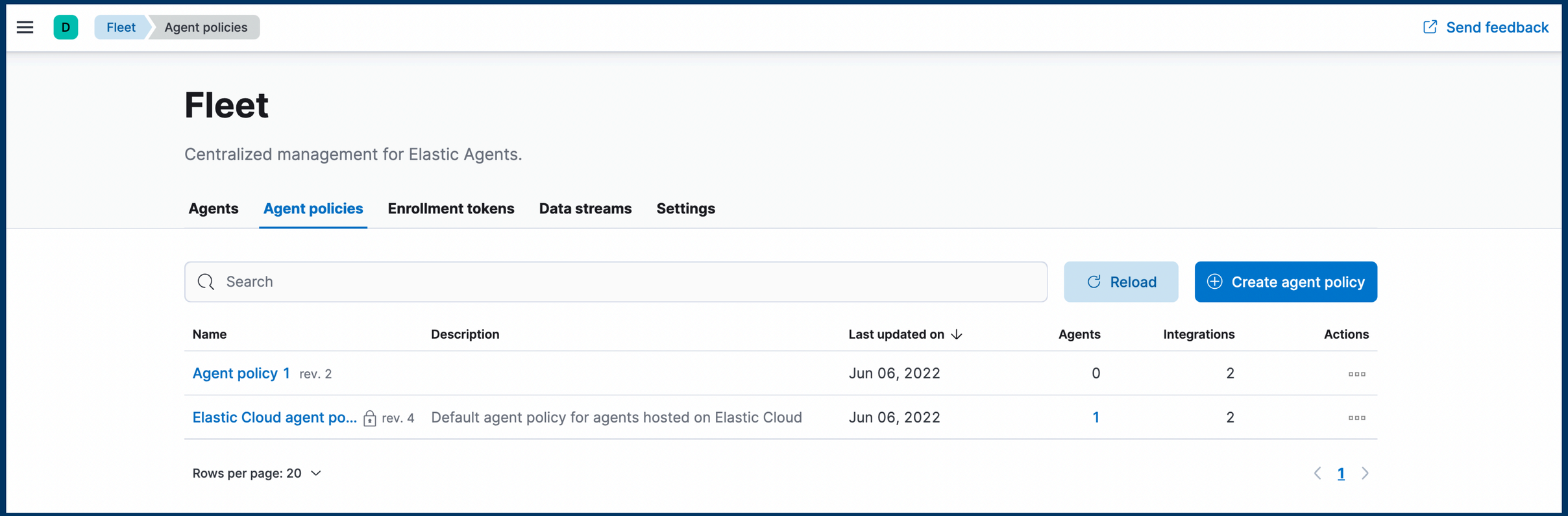
Elasticstack

Über die Dev Tools - Management gelangt man zu Fleet …



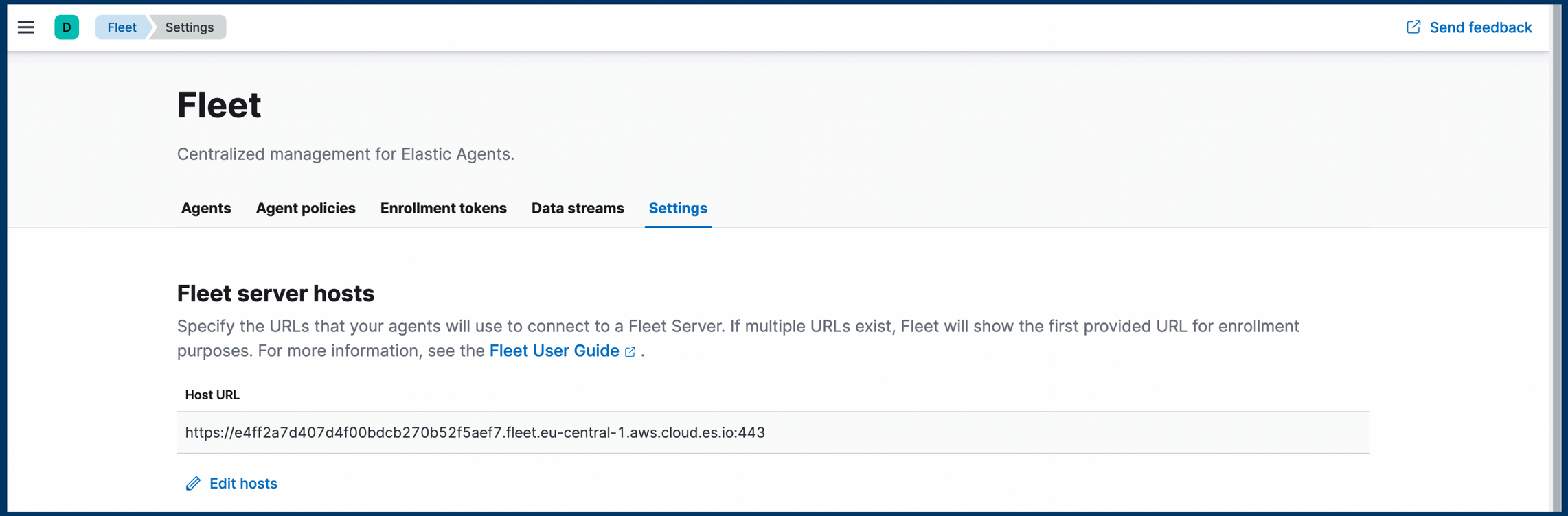
Elasticstack

Bevor wir einen neuen Agenten einrichten können benötigt man eine „policy“. Diese wurde schon angelegt, Agent policy 1.



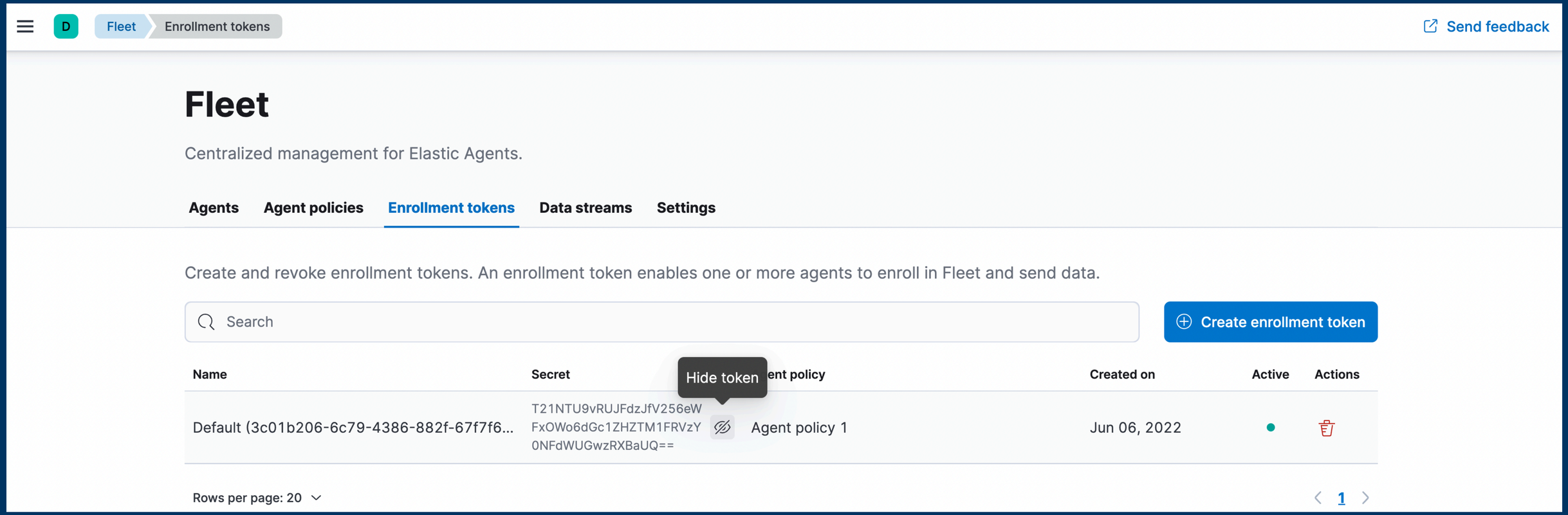
Elasticstack

Auf Fleet - Settings finden man die server host Adresse



Elasticstack

Auf Fleet - Enrollment Token bekommt man das Token:



Elasticstack

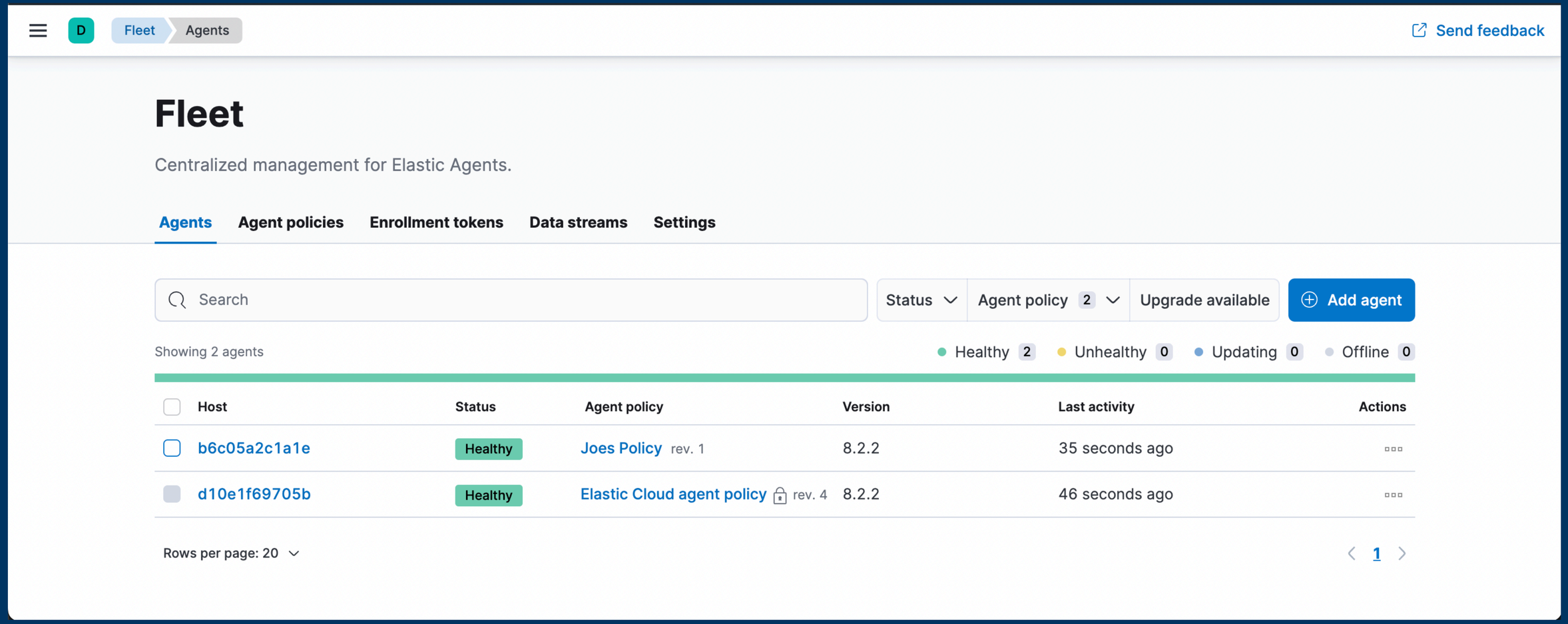
Damit haben wir alle Informationen zusammen, um den Container zu starten:

```
docker run \
  --env FLEET_ENROLL=1 \
  --env FLEET_URL=https://e4ff2a7d407d4f00bdcb270b52f5aef7.fleet.eu-central-1.aws.cloud.es.io:443 \
  --env
FLEET_ENROLLMENT_TOKEN=T21NTU9vRUJFdzJfV256eWFxOWo6dGc1ZHZTM1FRVzY0NFdWUGwzRXBaUQ== \
  --rm docker.elastic.co/beats/elastic-agent:8.2.2-arm64
```

Im Terminal werden viele JSON Dokumente erstellt. Dieses Terminal nicht schließen!

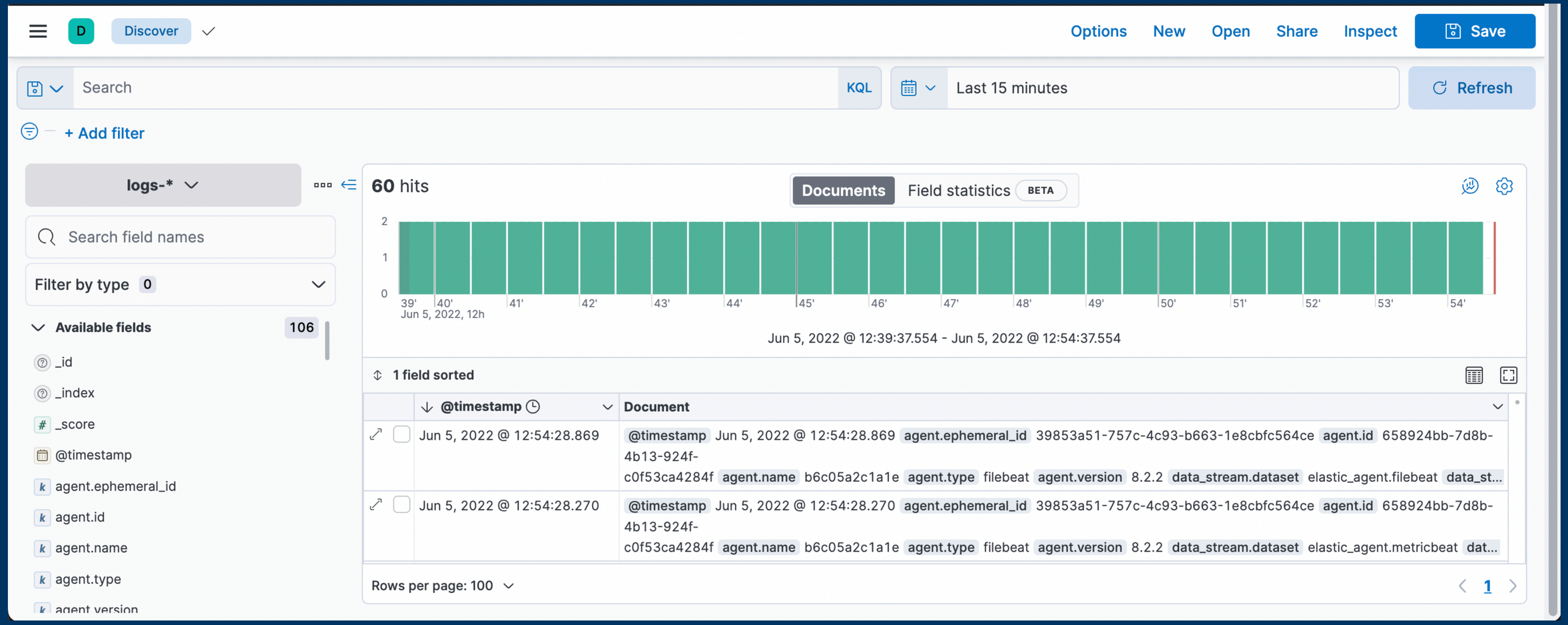
Elasticstack

Fleet Agents sollte dann in etwa so aussehen



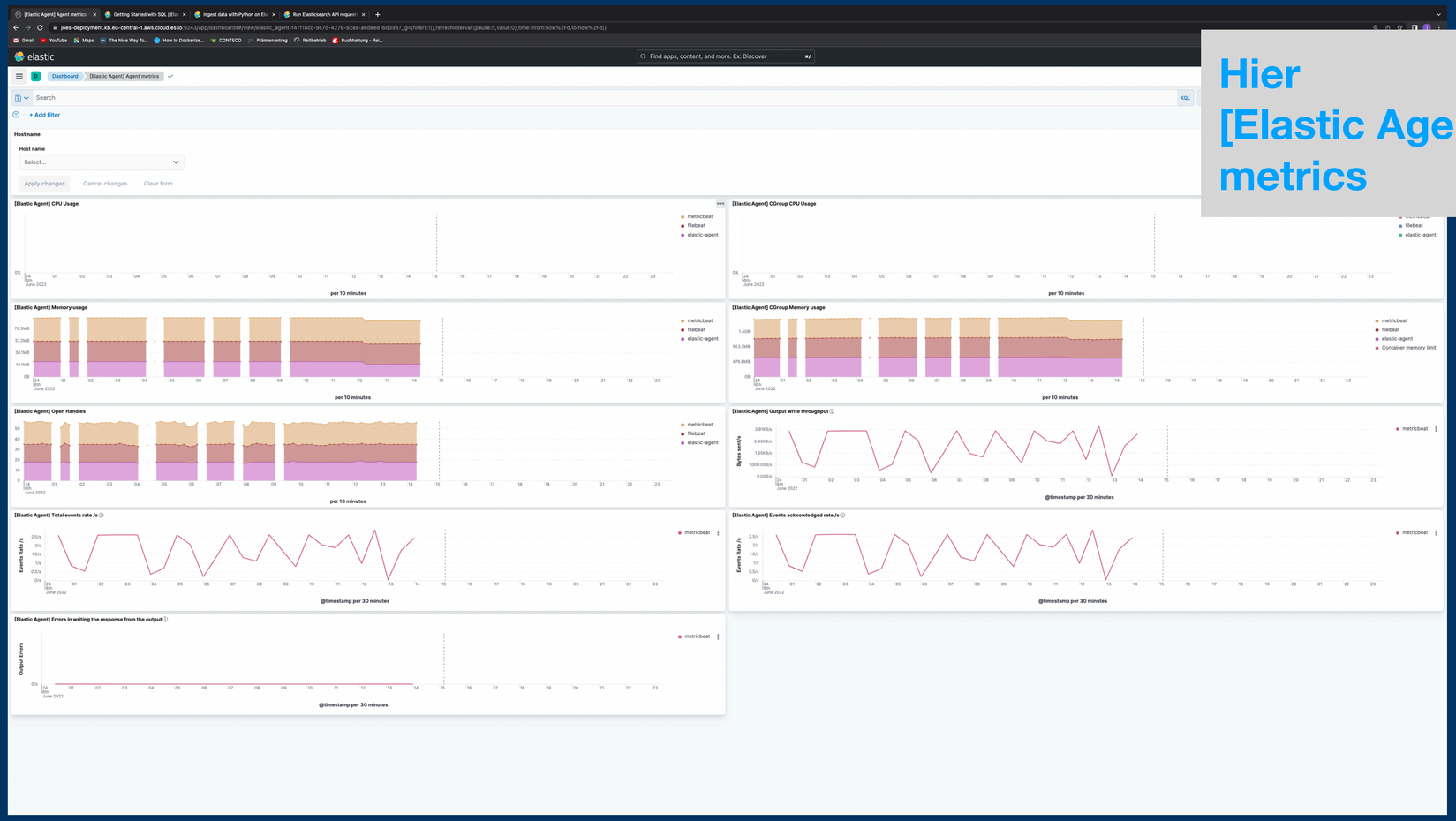
Elasticstack

Discover zeigt dann die eintreffenden Datenpakete …



Elasticstack

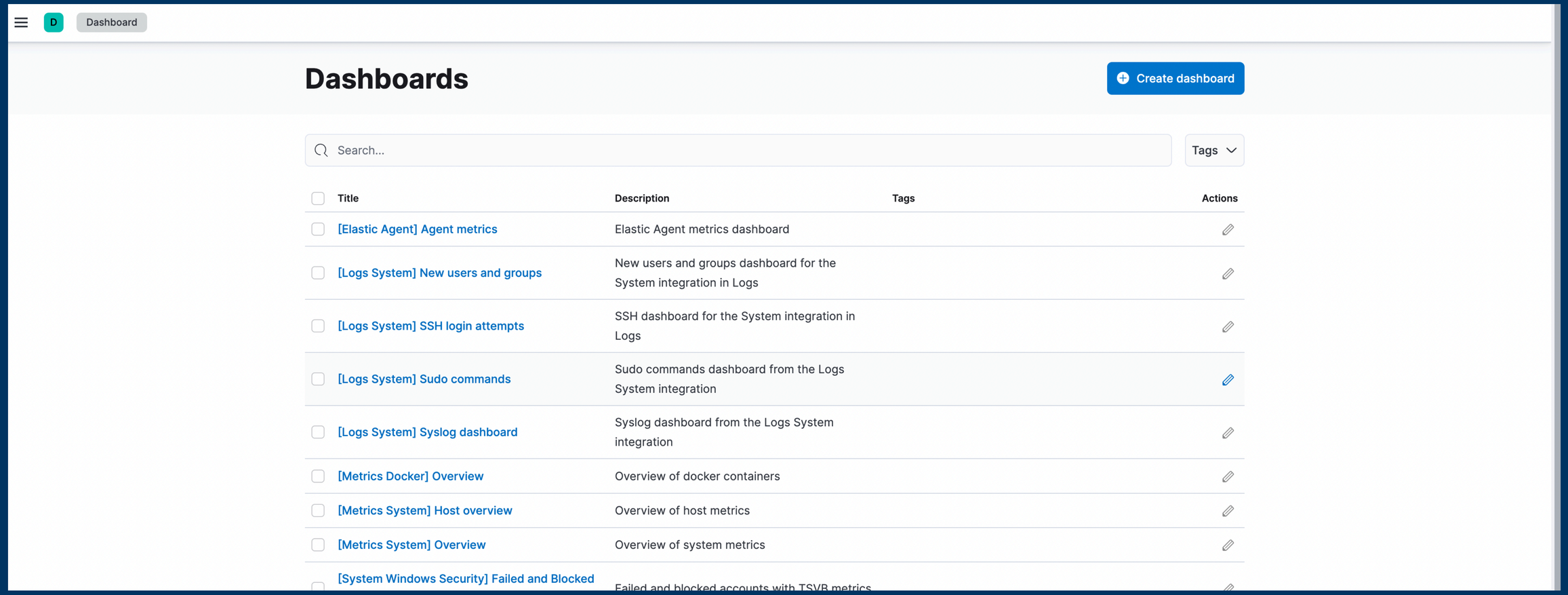
Daten können mit Dashboards visualisiert werden.



Hier
[Elastic Agent] Agent metrics

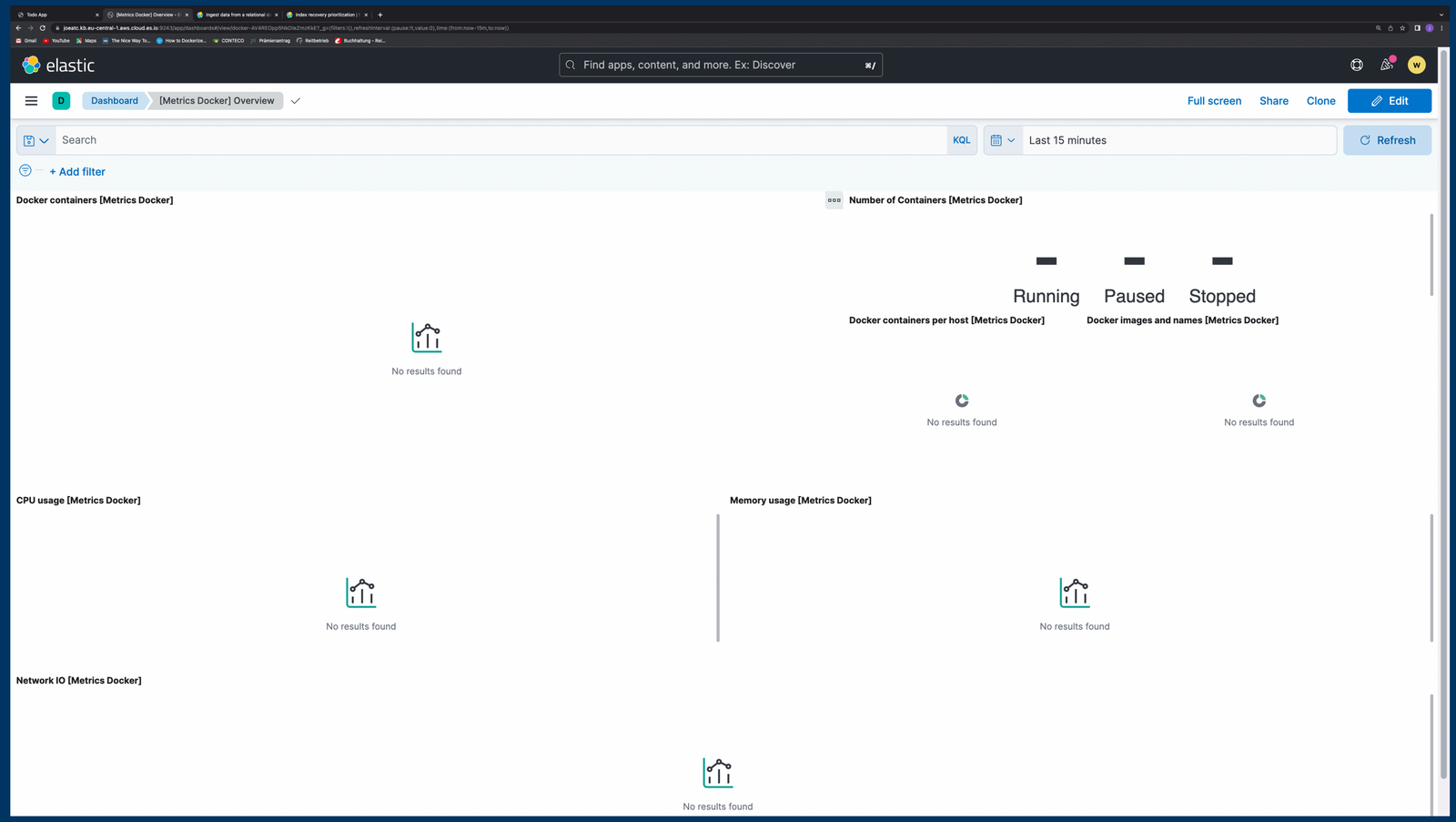
Elasticstack

Daten können mit Dashboards visualisiert werden. Etliche Dashboards sind bereits vorgefertigt. Wie zu erwarten, gibt es auch für Docker ein Dashboard: [Metrics Docker] Overview.



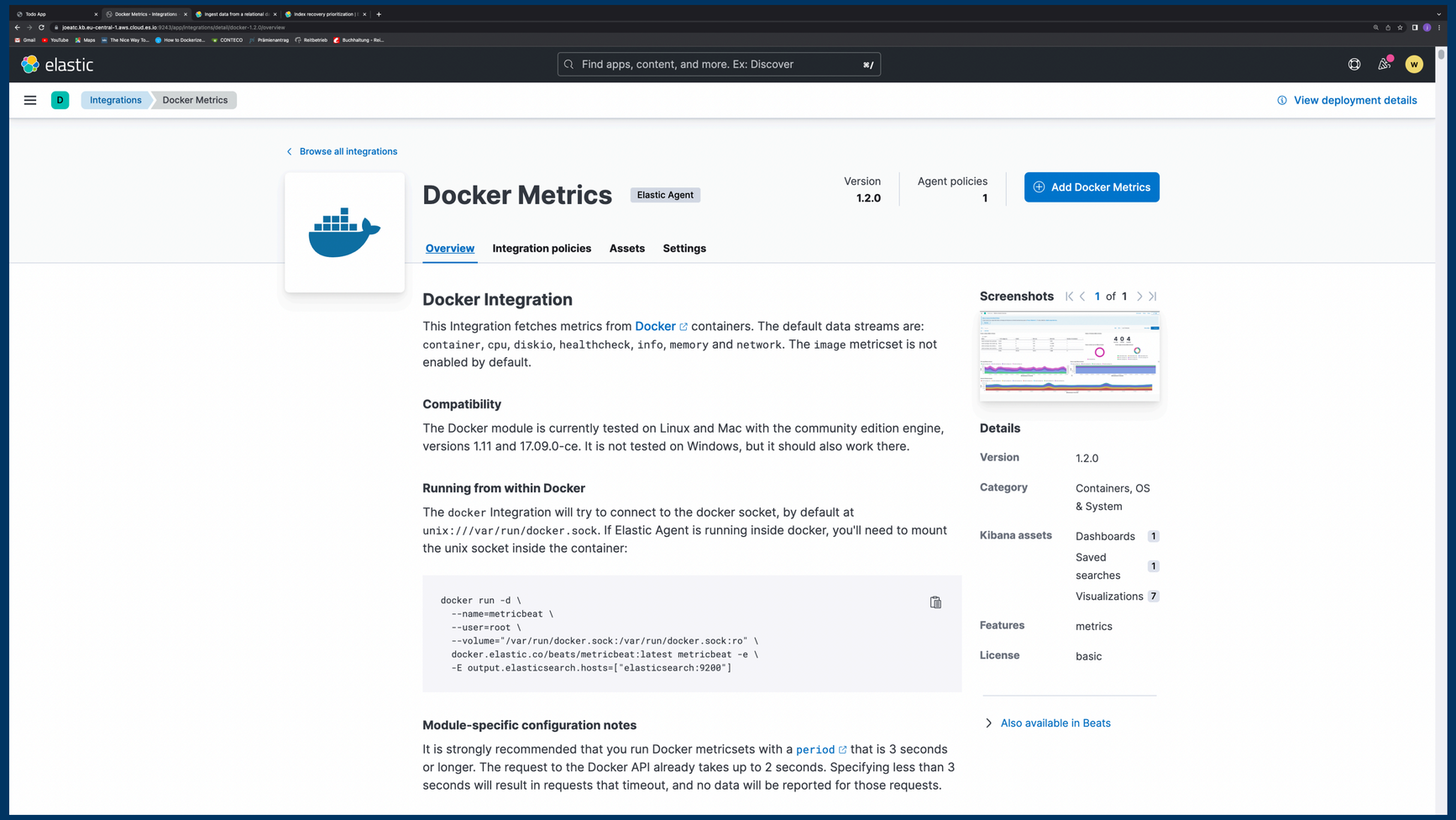
Elasticstack

Dieser zeigt aber noch keine Daten. Der gestartete Container Elastic Agent ist dafür nicht zuständig. Wir benötigen einen weiteren Container.



Elasticstack

Auf der Seite des Integrationsmoduls findet sich ein Hinweis auf die Installation dieses Containers. Leider nicht genug für unsere Konfiguration.



Elasticstack

Drei Informationen müssen gefunden werden:

1. output host —> Manage this Deployment —> Elastic Search copy endpoint
2. cloud id —> ebenda
3. cloud authentication —> die herunter geladenen credentials user und password

joe@MacBook % docker pull docker.elastic.co/beats/metricbeat:8.2.2-arm64

```
joe@MacBook % docker run -d \
  --name=metricbeat \
  --user=root \
  --volume="/var/run/docker.sock:/var/run/docker.sock:ro" \
  docker.elastic.co/beats/metricbeat:8.2.2-arm64 metricbeat -e \
  -E output.elasticsearch.hosts=https://joeatc.es.eu-central-1.aws.cloud.es.io:9243 \
  -E
cloud.id=joeatc:ZXUtY2VudHJhbC0xLmF3cy5jbG91ZC5lcy5pbyQxNTM3ODIzZGU1ZTU0OWI2OTZhM2NlNWRhO
TczMGZlNCQ4NTg3N2QzYzZjNWQ0ZjE2OTYwYzYyNWEwZDY5OTI5NQ== \
  -E cloud.auth=elastic:DVoxjSjZSEPgB4KV2X7OKOYH
```

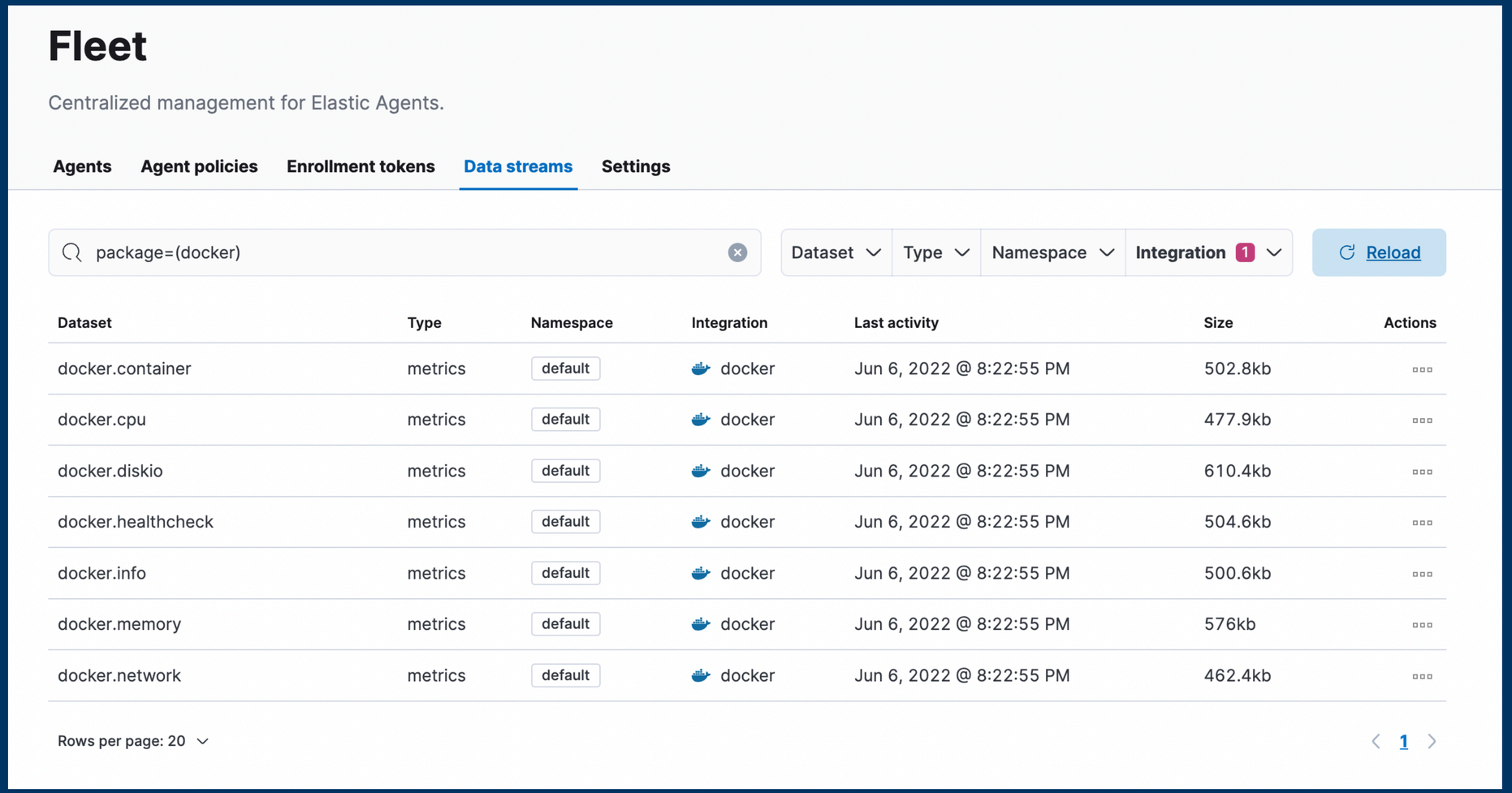
Elasticstack

Die Konsole meldet:

{"log.level":"info","@timestamp":"2022-06-06T18:16:36.846Z","log.logger":"publisher_pipeline_output","log.origin":{"file.name":"pipeline/client_worker.go","file.line":147},"message":"Connection to backoff(elasticsearch(https://1537823de5e549b696a3ce5da9730fe4.eu-central-1.aws.cloud.es.io:443)) established","service.name":"metricbeat","ecs.version":"1.6.0"}
{"log.level":"info","@timestamp":"2022-06-06T18:17:00.817Z","log.logger":"monitoring","log.origin":{"file.name":"log/log.go","file.line":184},"message":"Non-zero metrics in the last 30s","service.name":"metricbeat","monitoring":{"metrics":{"beat":{"cgroup":{"cpu":{"id":"/"},"memory":{"id":"/","mem":{"usage":{"bytes":68763648}}}},"cpu":{"system":{"ticks":270,"time":{"ms":270}},"total":{"ticks":860,"time":{"ms":860},"value":0},"user":{"ticks":590,"time":{"ms":590}}},"handles":{"limit":{"hard":1048576,"soft":1048576},"open":12},"info":{"ephemeral_id":"8d8bf070-4ce7-4af4-b88a-c4c806d7f70b","uptime":{"ms":33150},"version":"8.2.2"},"memstats":{"gc_next":21345648,"memory_alloc":12867840,"memory_sys":56050696,"memory_total":127438176,"rss":149110784},"runtime":{"goroutines":71}},"libbeat":{"config":{"module":{"running":3,"starts":3},"reloads":1,"scans":1},"output":{"events":{"acked":37,"active":0,"batches":5,"total":37},"read":{"bytes":8467},"type":"elasticsearch","write":{"bytes":380905}},"pipeline":{"clients":10,"events":{"active":4,"filtered":1,"published":41,"retry":13,"total":42},"queue":{"acked":37,"max_events":4096}}},"metricbeat":{"system":{"cpu":{"events":4,"success":4},"filesystem":{"events":1,"success":1},"fsstat":{"events":1,"success":1},"load":{"events":4,"success":4},"memory":{"events":4,"success":4},"network":{"events":14,"success":14},"process":{"events":6,"success":6},"process_summary":{"events":4,"success":4},"socket_summary":{"events":3,"success":3},"uptime":{"events":1,"success":1}}},"system":{"cpu":{"cores":4},"load":{"1":0.09,"15":0,"5":0.04,"norm":{"1":0.0225,"15":0,"5":0.01}}}},"ecs.version":"1.6.0"}}

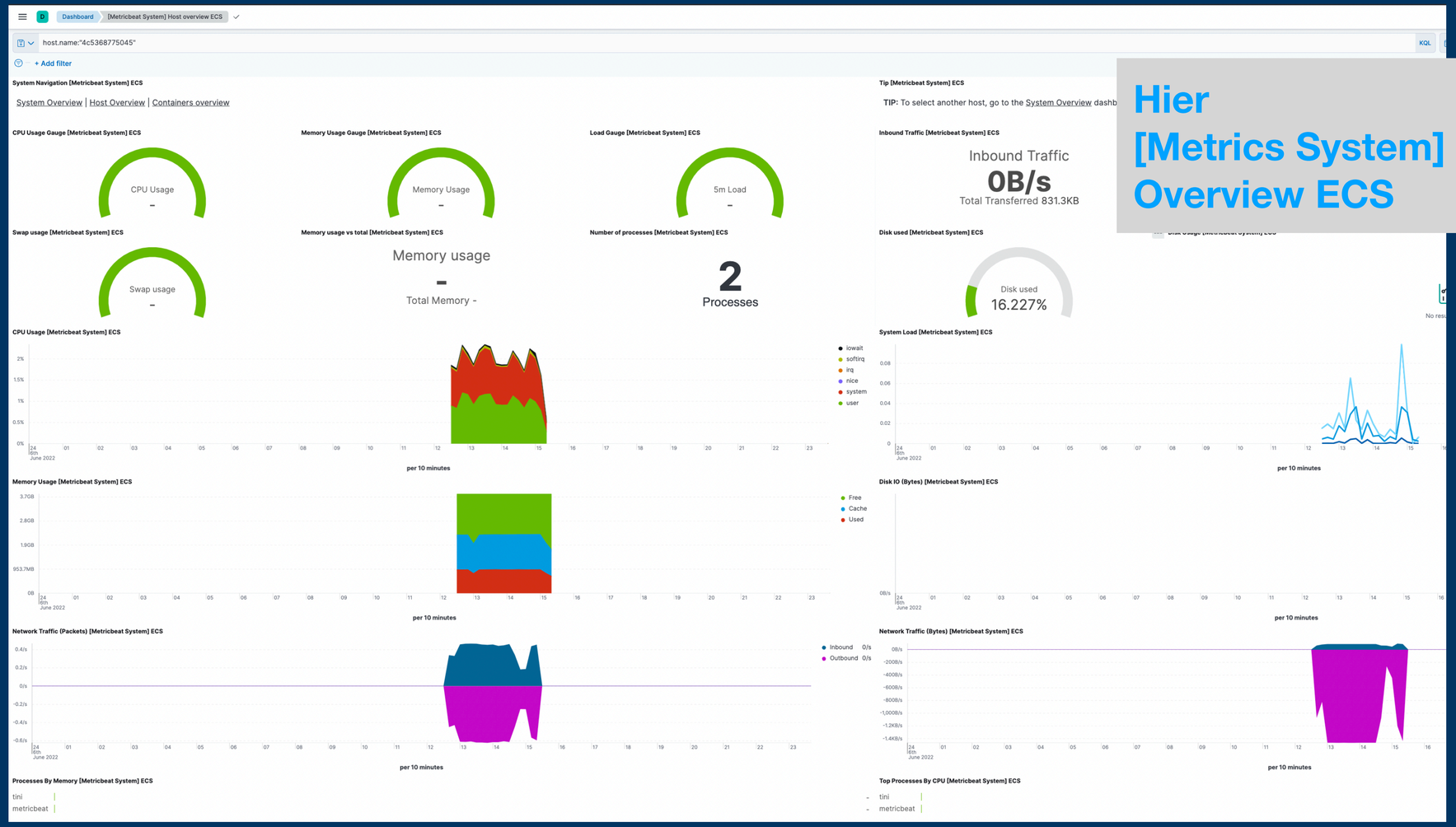
Elasticstack

Ein Blick in Fleet mit einem gelegentliche Reload zeigt, dass die Daten fließen.
Wenn man im Feld Integration „Docker“ wählt, erhält man diese Ansicht:



Elasticstack

Daten können mit Dashboards visualisiert werden.

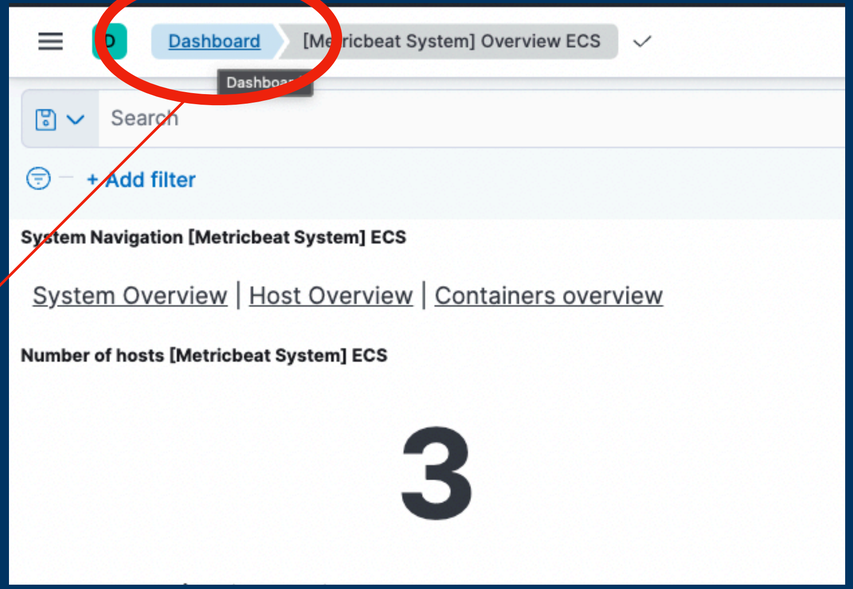


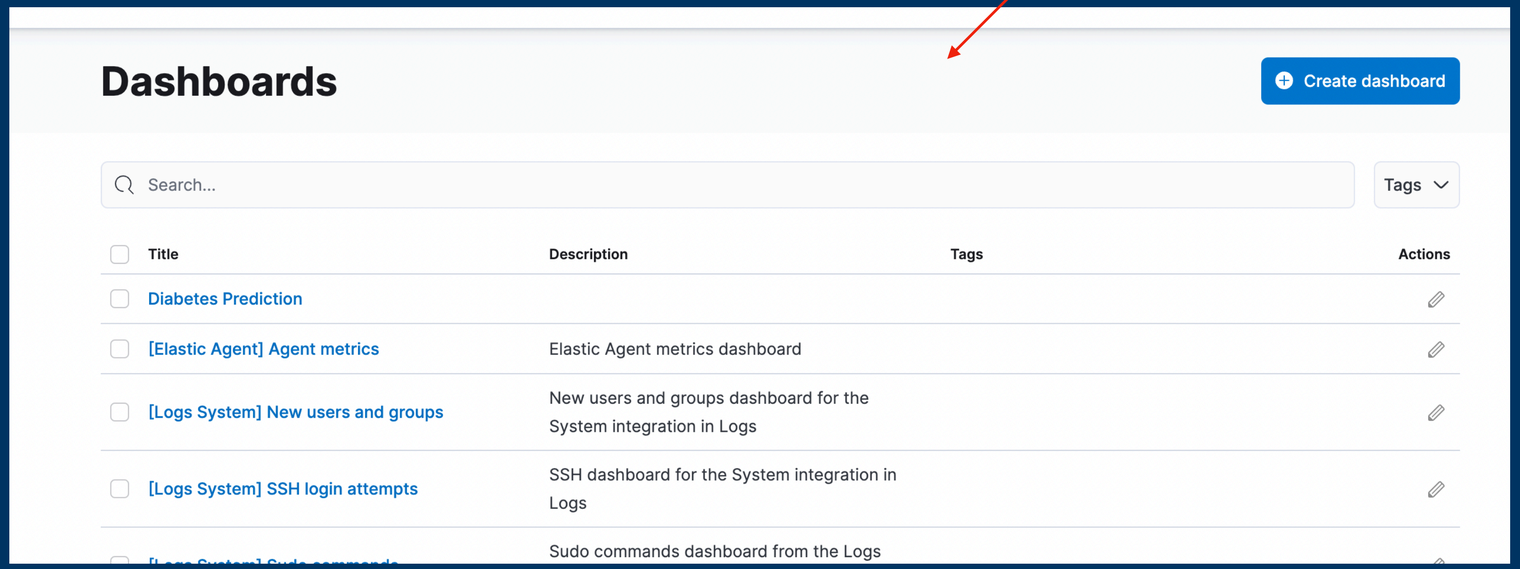
Hier
[Metrics System] Host
Overview ECS

Elasticstack

Verwendung von Kibana

Klickt man auf Dashboard, so wird die Liste der bestehenden Dashboards angezeigt.





Nicht alle Dashboards zeigen etwas an, oft fehlen die entsprechenden Daten.

Elasticstack

Stellt sich die Frage, ob unsere Lineare Regression auf das Deployment anwendbar ist:

In der Dokumentation über Python Integration finden sich die Anweisungen die für die Anbindung an die Cloud notwendig sind:

- ini-Datei erzeugen
- credentials eintragen
- package laden
- Elasticsearch Instanz anpassen

Elasticstack

example.ini

[ELASTIC]
cloud_id =
Joes_Deployment:ZXUtY2VudHJhbC0xLmF3cy5jbG91ZC5lcy5pbyRmM2ZmYWIwODcyNTI0OGRlOTU4MDVhYTM0MDJiODc4MyQwMDNlMzE2YmIxYWQ0ZTUzOTZmZjJjNzM3NWEzOTdiYw==
user = elastic
password = ML7v7y9hT5UGBwv4KEHYwuPP

Elasticstack

Pakete und Einstellungen in der Python Datei:

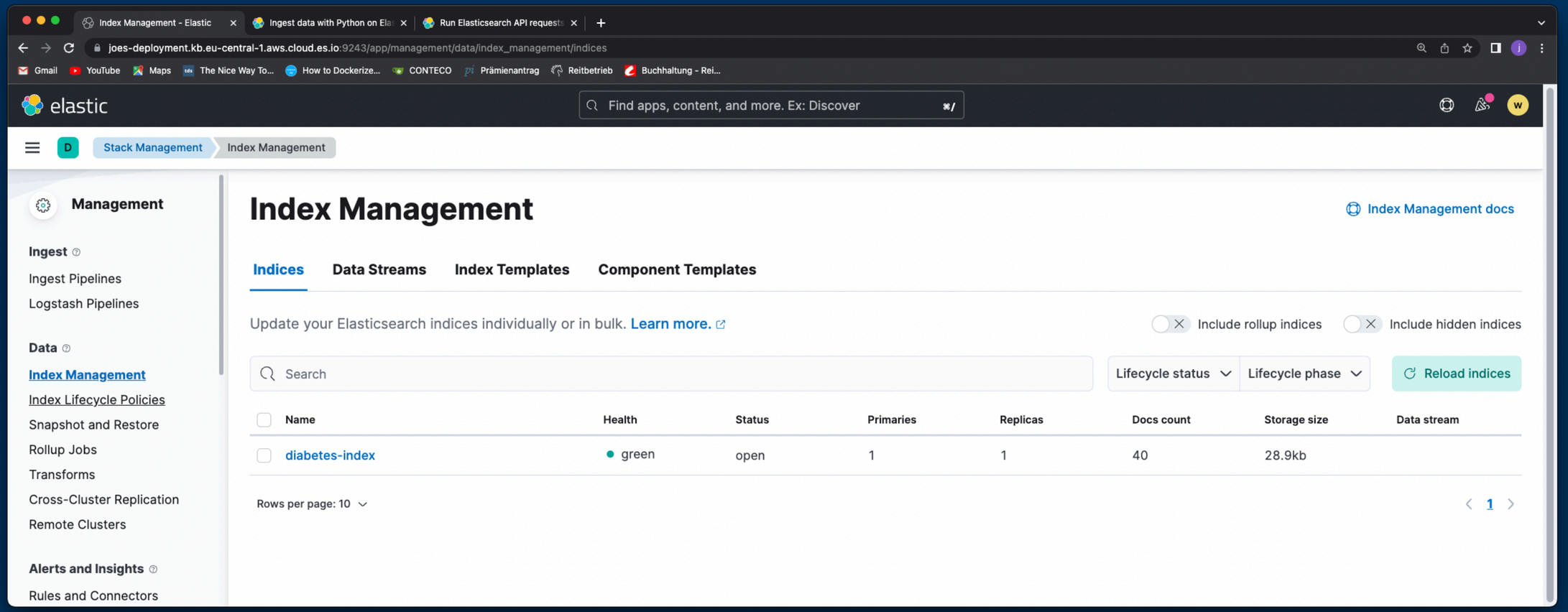
```
import configparser

config = configparser.ConfigParser()
config.read('example.ini')


# Create the client instance
es = Elasticsearch(
    cloud_id=config['ELASTIC']['cloud_id'],
    http_auth=(config['ELASTIC']['user'], config['ELASTIC']['password'])
)
```

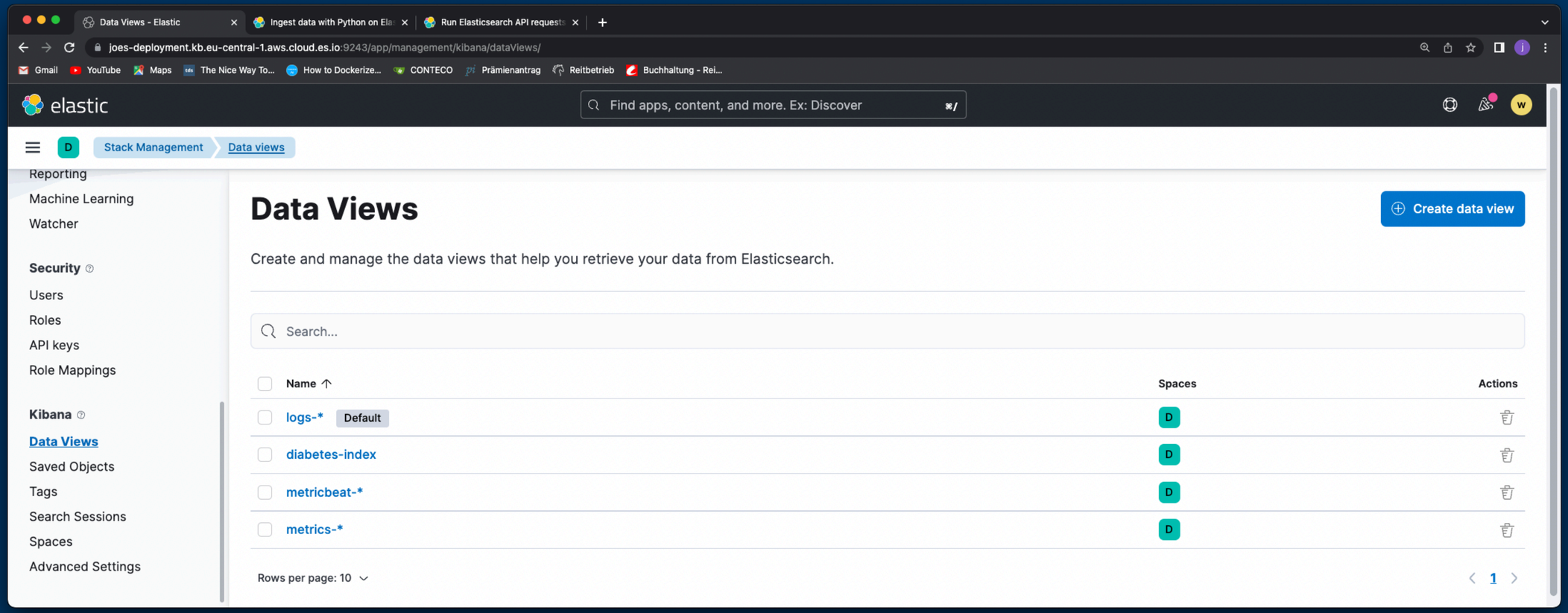
Elasticstack

Nach einem run Kommando sollte in Kibana - DevTools - Stack Management - Index Management der neue Index vorliegen



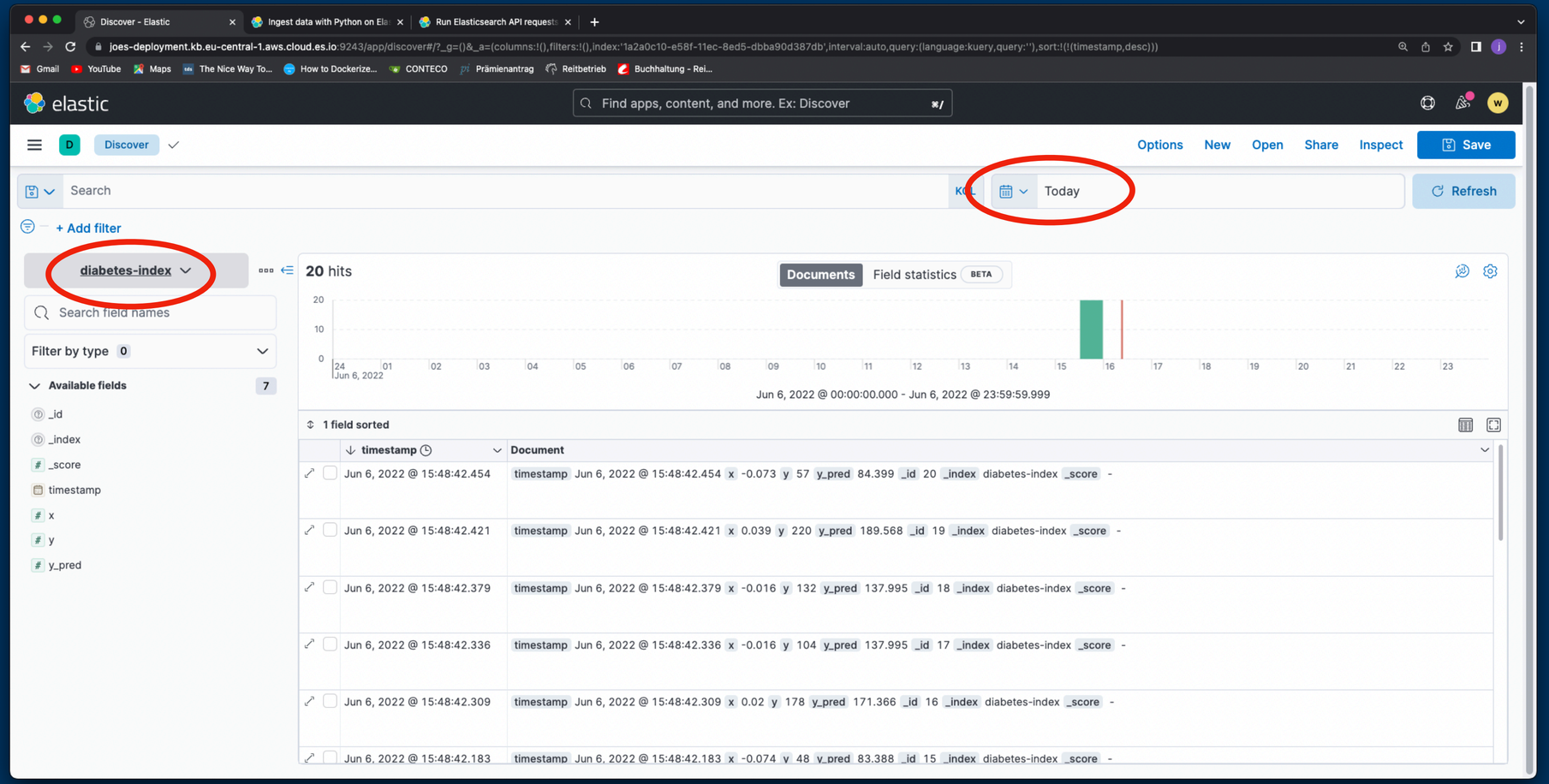
Elasticstack

Mittels Create data view wird eine neue Sicht auf die Daten erstellt



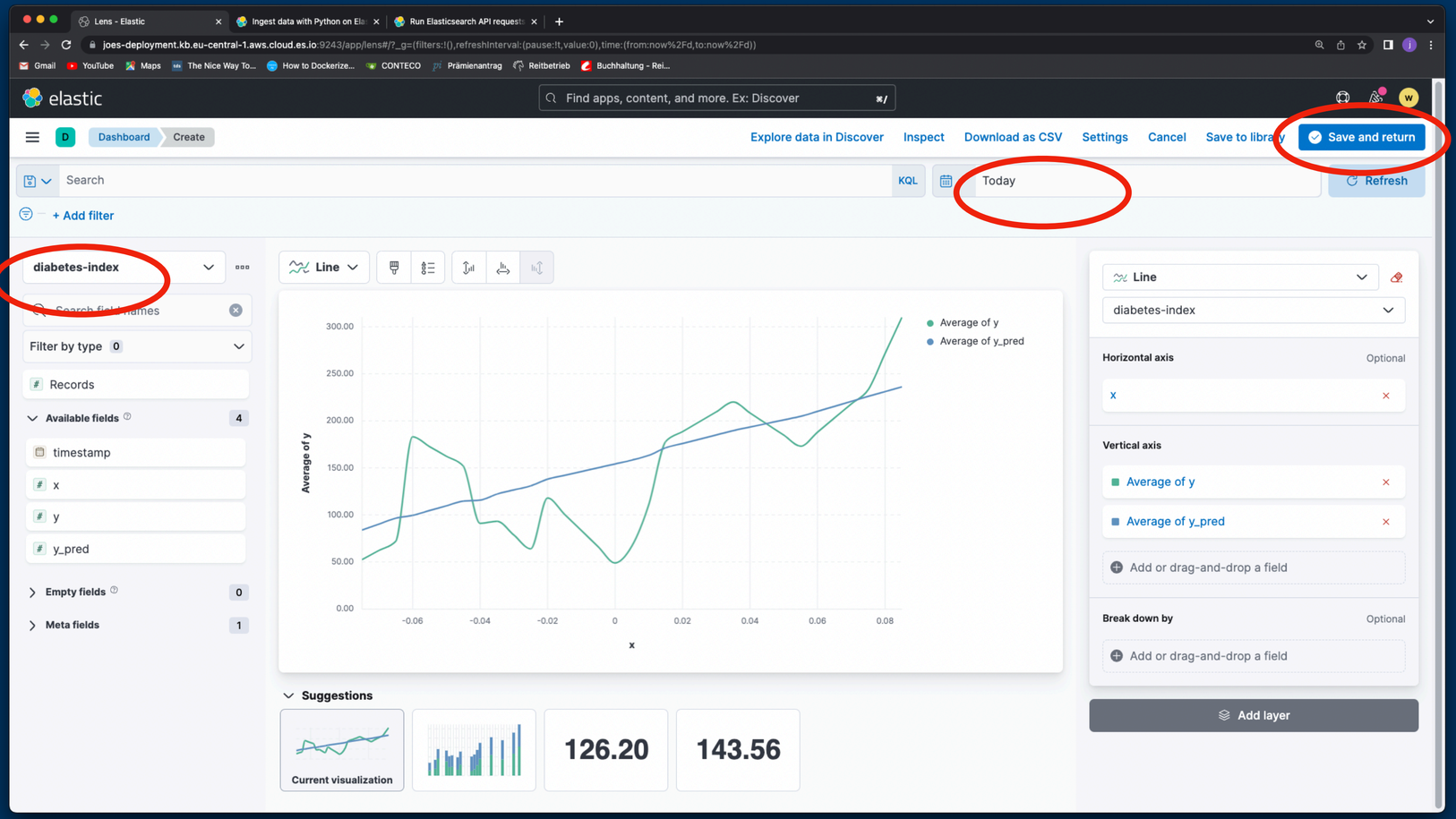
Elasticstack

Wählt man in discover den richtigen Index und passt den Zeitraum an („Today“), so sollten die Daten sichtbar sein.



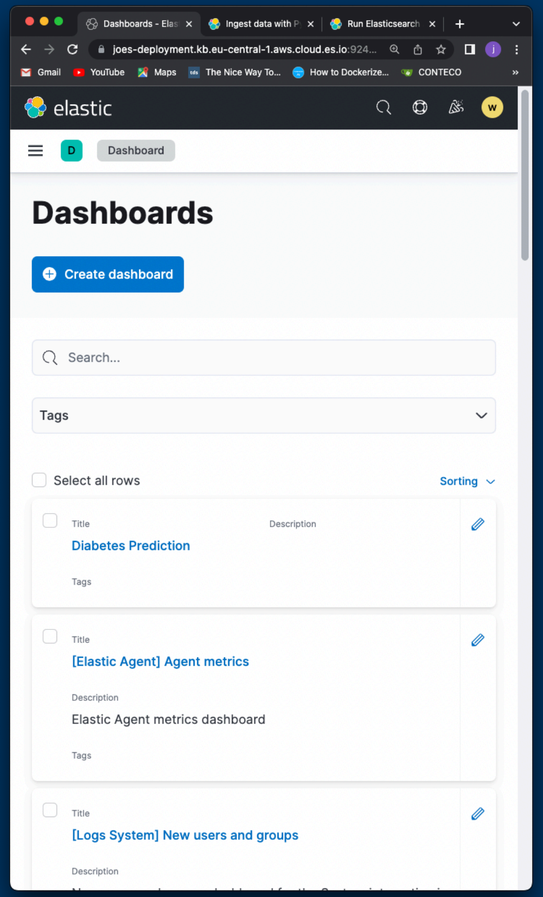
Elasticstack

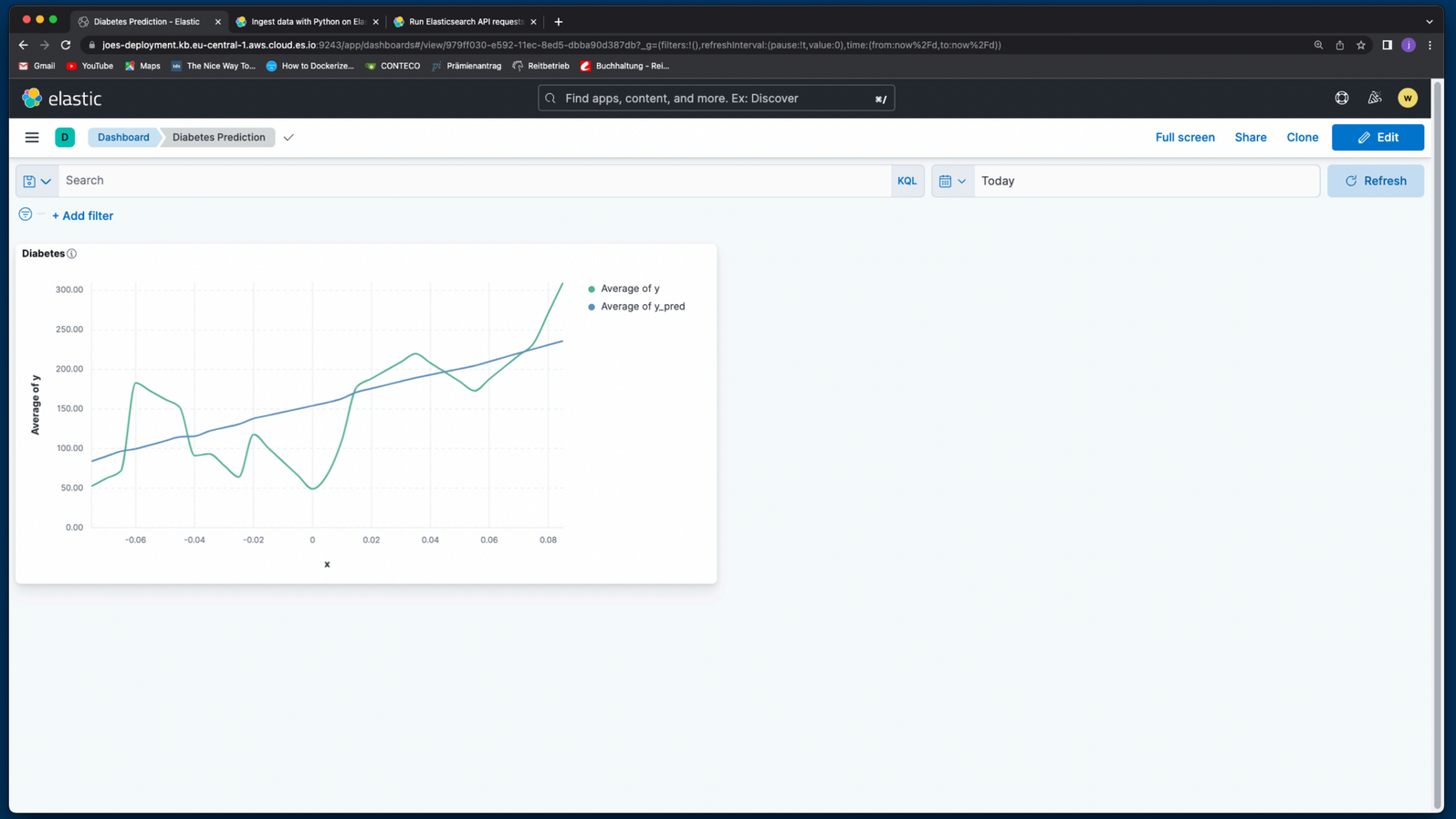
Über dashboard create gelangt man in die visualisation. Hier Kann der Grafiktyp und das Aussehen der Grafik angepasst werden. Speichern und zurück sichert die Arbeit.



Elasticstack

Über dashboard create gelangt man in die visualisation. Hier Kann der Grafiktyp und das Aussehen der Grafik angepasst werden. Speichern und zurück sichert die Arbeit.





Elasticstack

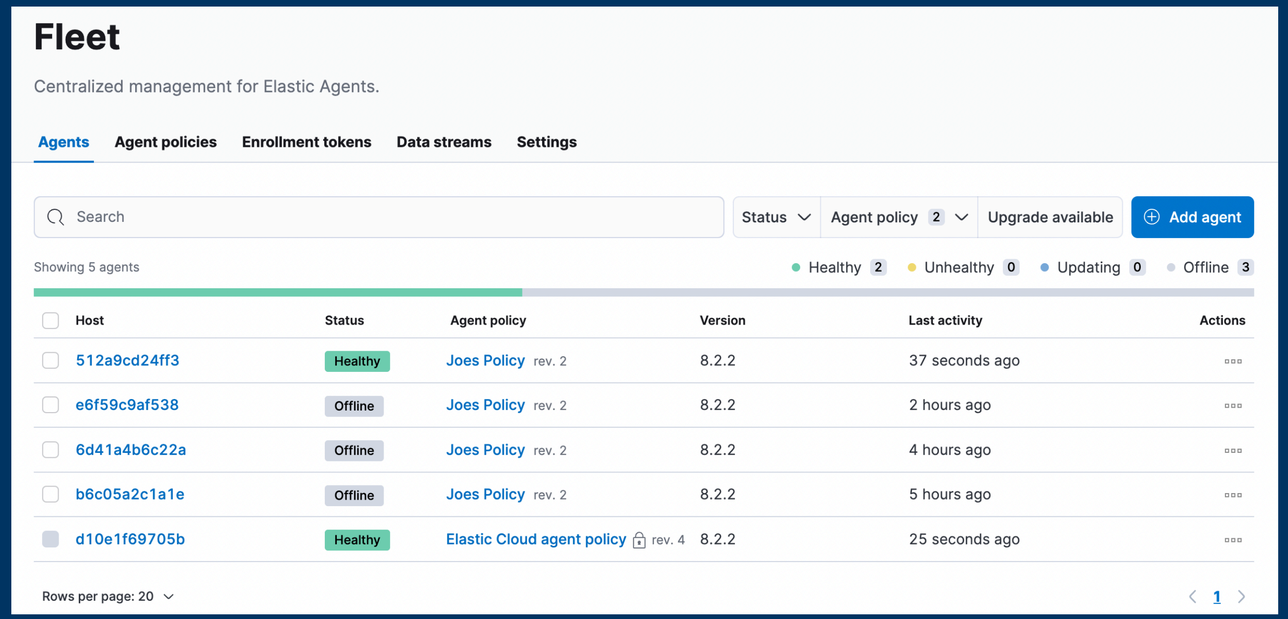
Konklusion

- Es wurden zwei Container erstellt, die kontinuierlich Daten an die Elastic Cloud senden.
- Dabei ist der Elastic Agent notwendig, um die Integration des Docker Metricbeats zu gewährleisten.
- Trotzdem laufend Daten ankommen, ist es möglich, Daten ganz anderer Art in den Cluster zu laden und auszuwerten.

Elasticstack

Konklusion

- Beim Arbeiten mit Containern ist darauf Rücksicht zu nehmen, dass das Schließen der Konsole zur Beendigung des Containers führt.
- Dadurch kommt es zu diesem Bild:



Offline Instanzen können gelöscht werden, da es sich um die alten, geschlossenen Container handelt. Es sei denn, man möchte die alten Daten noch auswerten.

Elasticstack

Fleet Agent Instanz löschen über Unenroll agent