1.0 - Familiarizing with the project
I ran:
mvn clean package -DskipTests

a .jar file was created in target/my-api-1.0.0.1591234567.jar

Run jar file:

java -jar target/my-api-1.0.0.1591234567.jar

output was:
16:58:46.872 [main] INFO u.c.c.hr.devops.DevopsTestApp - Starting awesome application uk.co.correvate.hr.devops:my-api:1.0.0.1591234567 ...

1.1 - CI Pipeline

Create a local jenkins using docker (in docker)
According to
https://medium.com/@kangasta/developing-jenkins-pipelines-on-top-of-docker-for-windows-1a4146d3734d

This binds windows docker with WSL docker for much fun.

Jenkins pipeline for changing the timestamp in the pom.xml file

```
environment {
        TIMESTAMP = sh(script: "echo `date +%s`", returnStdout:
true).trim()
    }
```

Pom.xml:

```
<groupId>uk.co.correvate.hr.devops</groupId>
    <artifactId>my-api</artifactId>
    <version>1.0.1.${env.TIMESTAMP}</version>
    <packaging>jar</packaging>
```

In the jenkins log:
[INFO]
[INFO] ------------------< uk.co.correvate.hr.devops:my-api >------------------
[INFO] Building my-api 1.0.1.1596554097
[INFO] --------------------------------[jar]---------------------------------

[INFO]
[INFO] --- spring-boot-maven-plugin:2.3.1.RELEASE:build-info (default) @ my-api ---
[INFO]

1.2 - Docker

Created a Dockerfile

```
FROM adoptopenjdk/openjdk11:latest
RUN mkdir /opt/app
COPY my-api.jar /opt/app/my-api.jar
CMD ["java", "-jar", "/opt/app/my-api.jar"]
```

Tweaked the Jenkinsfile to use the plugin
Since you use a private repo you need to create a crededital file under ~/.docker/config.json

"auths": {
  "https://index.docker.io/v1/": {
    "auth": "FOO",
    "email": "me@me.com"
  }
}

Mind you the auth parameter is a Base64-encoded version of username:password (i.e., echo -n 'username:password' | base64).

Removed comment from pom.xml to use the plugin

Now, Jenkinsfile looks like this:

```
pipeline {

        agent {
        docker {
            image 'adoptopenjdk/maven-openjdk11'
            args '-v /root/.m2:/root/.m2'
        }
    }
    environment {
        TIMESTAMP = sh(script: "echo `date +%s`", returnStdout:
true).trim()
    }

    stages {
        stage('Build container and push') {
            steps {
                dir("my-api") {
                    sh 'mvn clean package -DskipTests && cp
target/my-api-1.0.1.${TIMESTAMP}.jar src/main/docker/my-api.jar'
                    sh 'mvn docker:build docker:push'
                }
            }
        }
    }
    post {
        always {
            cleanWs()
        }
    }
}
```

The plugin will build, copy the .jar to the folder src/main/docker/ create and ship the image.

And the output is:

```
Started by user admin
Obtained my-api/Jenkinsfile from git https://github.com/smolianskyd/jenkins-docker-maven.git
Running in Durability level: MAX_SURVIVABILITY
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/jenkins_home/workspace/timestamp
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/smolianskyd/jenkins-docker-maven.git
 > git init /var/jenkins_home/workspace/timestamp # timeout=10
Fetching upstream changes from https://github.com/smolianskyd/jenkins-docker-maven.git
 > git --version # timeout=10
 > git --version # 'git version 2.20.4'
 > git fetch --tags --force --progress -- https://github.com/smolianskyd/jenkins-docker-maven.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/smolianskyd/jenkins-docker-maven.git # timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/smolianskyd/jenkins-docker-maven.git # timeout=10
Fetching upstream changes from https://github.com/smolianskyd/jenkins-docker-maven.git
 > git fetch --tags --force --progress -- https://github.com/smolianskyd/jenkins-docker-maven.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
 > git rev-parse refs/remotes/origin/origin/master^{commit} # timeout=10
Checking out Revision aa70d7f3c915af81716aea5cfc38d1009dd4fbc4 (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f aa70d7f3c915af81716aea5cfc38d1009dd4fbc4 # timeout=10
Commit message: "package"
 > git rev-list --no-walk 79abe0b3bcee23ad1eeba7e35e0d3248012431c1 # timeout=10
[Pipeline] }
[Pipeline] // stage
[Pipeline] withEnv
[Pipeline] {
[Pipeline] isUnix
[Pipeline] sh
+ docker inspect -f . adoptopenjdk/maven-openjdk11
.
[Pipeline] withDockerContainer
Jenkins seems to be running inside container
e1086118b316194d54a87e224863effd264e147045f1634eb4b52036ba3fe397
```

```
$ docker run -t -d -u 0:0 -v /root/.m2:/root/.m2 -w /var/jenkins_home/workspace/timestamp
--volumes-from e1086118b316194d54a87e224863effd264e147045f1634eb4b52036ba3fe397 -e
******** -e ******** -e ******** -e ******** -e ******** -e ******** -e ******** -e ******** -e ********
-e ******** -e ******** -e ******** -e ******** -e ******** -e ******** -e ******** -e ******** -e
******** -e ******** -e ******** -e ******** -e ******** adoptopenjdk/maven-openjdk11 cat
$ docker top 565875ce0a0722fc8f6944b82e03469554e57b4e40e438d4a0074e163b972049 -eo
pid,comm
[Pipeline] {
[Pipeline] sh
+ date +%s
+ echo 1596633441
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Build container and push)
[Pipeline] dir
Running in /var/jenkins_home/workspace/timestamp/my-api
[Pipeline] {
[Pipeline] sh
+ mvn clean package -DskipTests
[INFO] Scanning for projects...
[WARNING]
[WARNING] Some problems were encountered while building the effective model for
uk.co.correvate.hr.devops:my-api:jar:1.0.1.1596633441
[WARNING] 'version' contains an expression but should be a constant. @
uk.co.correvate.hr.devops:my-api:1.0.1.${env.TIMESTAMP},
/var/jenkins_home/workspace/timestamp/my-api/pom.xml, line 9, column 11
[WARNING]
[WARNING] It is highly recommended to fix these problems because they threaten the stability of
your build.
[WARNING]
[WARNING] For this reason, future Maven versions might no longer support building such
malformed projects.
[WARNING]
[INFO]
[INFO] ------------------< uk.co.correvate.hr.devops:my-api >------------------
[INFO] Building my-api 1.0.1.1596633441
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ my-api ---
[INFO]
[INFO] --- spring-boot-maven-plugin:2.3.1.RELEASE:build-info (default) @ my-api ---
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ my-api ---
```

```
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 1 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:compile (default-compile) @ my-api ---
[INFO] Changes detected - recompiling the module!
[WARNING] File encoding has not been set, using platform encoding UTF-8, i.e. build is platform dependent!
[INFO] Compiling 1 source file to /var/jenkins_home/workspace/timestamp/my-api/target/classes
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ my-api ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] skip non existing resourceDirectory /var/jenkins_home/workspace/timestamp/my-api/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:testCompile (default-testCompile) @ my-api ---
[INFO] Changes detected - recompiling the module!
[WARNING] File encoding has not been set, using platform encoding UTF-8, i.e. build is platform dependent!
[INFO] Compiling 1 source file to /var/jenkins_home/workspace/timestamp/my-api/target/test-classes
[INFO]
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ my-api ---
[INFO] Tests are skipped.
[INFO]
[INFO] --- maven-jar-plugin:2.4:jar (default-jar) @ my-api ---
[INFO] Building jar: /var/jenkins_home/workspace/timestamp/my-api/target/my-api-1.0.1.1596633441.jar
[INFO]
[INFO] --- spring-boot-maven-plugin:2.3.1.RELEASE:repackage (default) @ my-api ---
[INFO] Replacing main artifact with repackaged archive
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  4.050 s
[INFO] Finished at: 2020-08-05T13:17:27Z
[INFO] ------------------------------------------------------------------------
+ cp target/my-api-1.0.1.1596633441.jar src/main/docker/my-api.jar
[Pipeline] sh
+ mvn docker:build docker:push
[INFO] Scanning for projects...
[WARNING]
[WARNING] Some problems were encountered while building the effective model for uk.co.correvate.hr.devops:my-api:jar:1.0.1.1596633441
```

```
[WARNING] 'version' contains an expression but should be a constant. @
uk.co.correvate.hr.devops:my-api:1.0.1.${env.TIMESTAMP},
/var/jenkins_home/workspace/timestamp/my-api/pom.xml, line 9, column 11
[WARNING]
[WARNING] It is highly recommended to fix these problems because they threaten the stability of
your build.
[WARNING]
[WARNING] For this reason, future Maven versions might no longer support building such
malformed projects.
[WARNING]
[INFO]
[INFO] ------------------< uk.co.correvate.hr.devops:my-api >------------------
[INFO] Building my-api 1.0.1.1596633441
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- docker-maven-plugin:0.33.0:build (default-cli) @ my-api ---
[INFO] Copying files to
/var/jenkins_home/workspace/timestamp/my-api/target/docker/correvate-test/my-api/1.0.1.15966334
41/build/maven
[WARNING] DOCKER> Dockerfile
/var/jenkins_home/workspace/timestamp/my-api/src/main/docker/Dockerfile does not contain an
ADD or COPY directive to include assembly created at maven. Ignoring assembly.
[INFO] Building tar:
/var/jenkins_home/workspace/timestamp/my-api/target/docker/correvate-test/my-api/1.0.1.15966334
41/tmp/docker-build.tar
[INFO] DOCKER> [correvate-test/my-api:1.0.1.1596633441] "my-api": Created docker-build.tar in 1
second
[INFO] DOCKER> Step 1/4 : FROM adoptopenjdk/openjdk11:latest
[INFO] DOCKER>
[INFO] DOCKER> ---> 99afe502a027
[INFO] DOCKER> Step 2/4 : RUN mkdir /opt/app
[INFO] DOCKER>
[INFO] DOCKER> ---> Using cache
[INFO] DOCKER> ---> 3ac8330edb97
[INFO] DOCKER> Step 3/4 : COPY my-api.jar /opt/app/my-api.jar
[INFO] DOCKER>
[INFO] DOCKER> ---> 357e8568076f
[INFO] DOCKER> Step 4/4 : CMD ["java", "-jar", "/opt/app/my-api.jar"]
[INFO] DOCKER>
[INFO] DOCKER> ---> Running in f93b44080e3b
[INFO] DOCKER> Removing intermediate container f93b44080e3b
[INFO] DOCKER> ---> 2f90afe705cf
[INFO] DOCKER> [Warning] One or more build-args [VERSION] were not consumed
[INFO] DOCKER> Successfully built 2f90afe705cf
[INFO] DOCKER> Successfully tagged correvate-test/my-api:1.0.1.1596633441
```

[INFO] DOCKER> [correvate-test/my-api:1.0.1.1596633441] "my-api": Built image sha256:2f90a
[INFO]
[INFO] --- docker-maven-plugin:0.33.0:push (default-cli) @ my-api ---
[INFO] DOCKER> The push refers to repository [docker.io/correvate-test/my-api]
348897275498: Preparing
fc7b54b43816: Preparing
50b2f50bbd0e: Preparing
23e927a44b7a: Preparing
8682f9a74649: Preparing
d3a6da143c91: Preparing
83f4287e1f04: Preparing
7ef368776582: Preparing
d3a6da143c91: Waiting
83f4287e1f04: Waiting
7ef368776582: Waiting
[INFO] DOCKER> Temporary image tag skipped. Target image
'correvate-test/my-api:1.0.1.1596633441' already has registry set or no registry is available
[ERROR] DOCKER> Unable to push 'correvate-test/my-api:1.0.1.1596633441' : denied: requested access to the resource is denied [denied: requested access to the resource is denied]
[INFO] --
[INFO] BUILD FAILURE
[INFO] --
[INFO] Total time: 6.644 s
[INFO] Finished at: 2020-08-05T13:17:36Z
[INFO] --
[ERROR] Failed to execute goal io.fabric8:docker-maven-plugin:0.33.0:push (default-cli) on project my-api: Unable to push 'correvate-test/my-api:1.0.1.1596633441' : denied: requested access to the resource is denied -> [Help 1]
[ERROR]
[ERROR] To see the full stack trace of the errors, re-run Maven with the -e switch.
[ERROR] Re-run Maven using the -X switch to enable full debug logging.
[ERROR]
[ERROR] For more information about the errors and possible solutions, please read the following articles:
[ERROR] [Help 1] http://cwiki.apache.org/confluence/display/MAVEN/MojoExecutionException
[Pipeline] }
[Pipeline] // dir
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] cleanWs
[WS-CLEANUP] Deleting project workspace...
[WS-CLEANUP] Deferred wipeout is used...
[WS-CLEANUP] done

```
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
$ docker stop --time=1
565875ce0a0722fc8f6944b82e03469554e57b4e40e438d4a0074e163b972049
$ docker rm -f 565875ce0a0722fc8f6944b82e03469554e57b4e40e438d4a0074e163b972049
[Pipeline] // withDockerContainer
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
ERROR: script returned exit code 1
Finished: FAILURE
```

If the auth details were in place the image would have been pushed successfully.

1.3 - Docker compose & logging
Elasticsearch Setup

Let’s create the image locally with the previously created Dockerfile

```
docker build --tag my-api:latest .

Sending build context to Docker daemon  18.38MB
Step 1/4 : FROM adoptopenjdk/openjdk11:latest
 ---> 99afe502a027
Step 2/4 : RUN mkdir /opt/app
 ---> Using cache
 ---> 3ac8330edb97
Step 3/4 : COPY my-api.jar /opt/app/my-api.jar
 ---> f859c27fc5e5
Step 4/4 : CMD ["java", "-jar", "/opt/app/my-api.jar"]
```

---> Running in 989c3cfd47a5
Removing intermediate container 989c3cfd47a5
---> 032f6605d1cc
Successfully built 032f6605d1cc
Successfully tagged my-api:latest

Consider the following docker-compose.yml

```
version: '2.2'
volumes:
  esdata:
    driver: local

networks:
  correvate:

services:
  # Elasticsearch + Kibana
  elasticsearch:
    image: docker.elastic.co/elasticsearch/elasticsearch:7.8.0
    environment:
      - discovery.type=single-node
      - "ES_JAVA_OPTS=-Xms512m -Xmx512m"
    volumes:
      - esdata:/usr/share/elasticsearch/data
    ports:
      - 9200:9200
    networks:
      - correvate

  kibana:
    image: docker.elastic.co/kibana/kibana:7.8.0
    ports:
      - 5601:5601
    environment:
      ELASTICSEARCH_URL: http://elasticsearch:9200
      ELASTICSEARCH_HOSTS: http://elasticsearch:9200
```

```
    networks:
      - correvate

  # Log processor & forwarder
  fluentbit:
    build: fluentbit/.
    ports:
      - "24224:24224"
      - "24224:24224/udp"
    networks:
      - correvate

  my-api:
    image: my-api:latest
    depends_on:
      - fluentbit
    logging:
      driver: fluentd
      options:
        tag: my-api
    networks:
      - correvate
```

Elasticsearch setup:

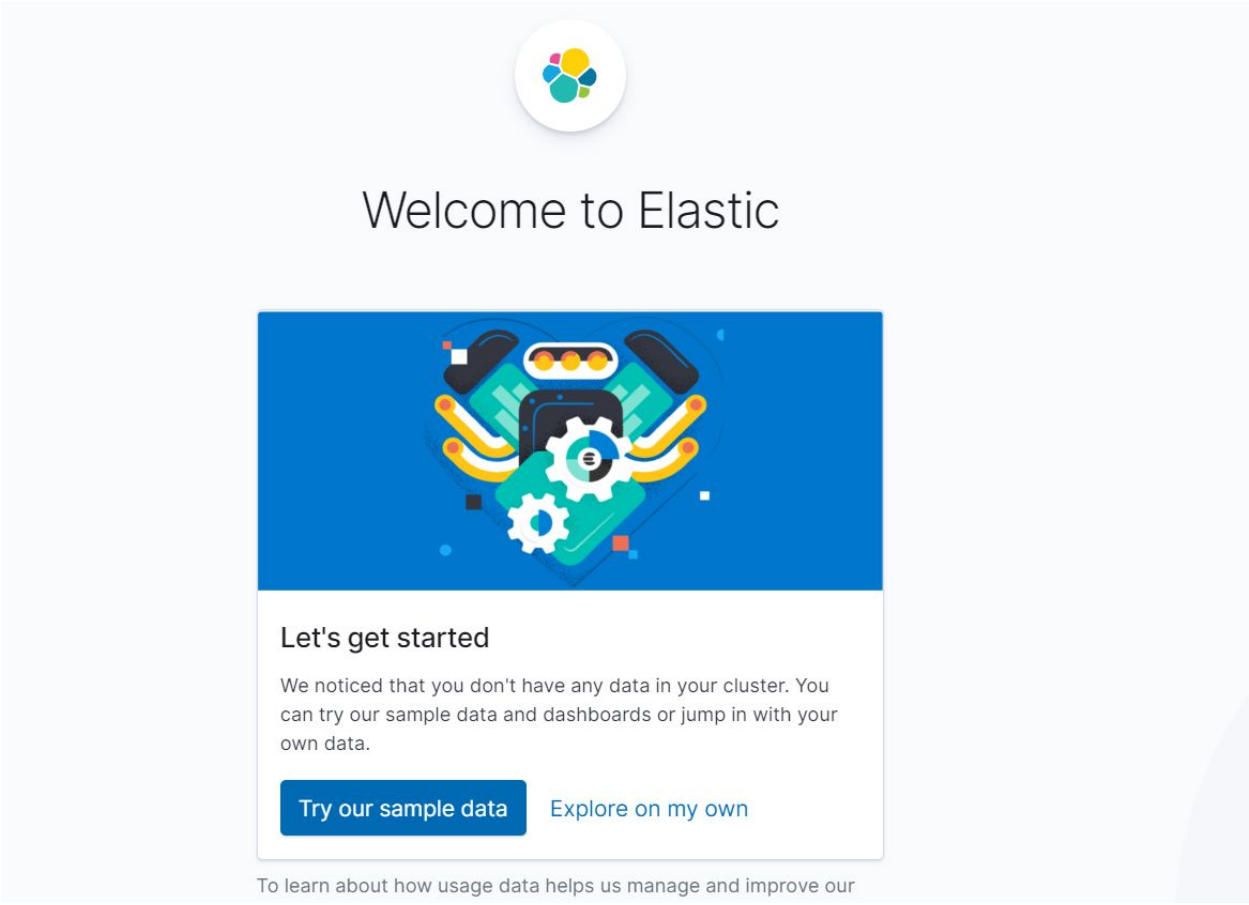
Open up http://localhost:9200 and see


```
{
  "name" : "7a98f541ad92",
  "cluster_name" : "docker-cluster",
  "cluster_uuid" : "nJm2exD-THi9YubU5W2tsA",
  "version" : {
    "number" : "7.8.0",
    "build_flavor" : "default",
    "build_type" : "docker",
    "build_hash" : "757314695644ea9a1dc2fecd26d1a43856725e65",
    "build_date" : "2020-06-14T19:35:50.234439Z",
    "build_snapshot" : false,
```

```
    "lucene_version" : "8.5.1",
    "minimum_wire_compatibility_version" : "6.8.0",
    "minimum_index_compatibility_version" : "6.0.0-beta1"
  },
  "tagline" : "You Know, for Search"
}
```


Kibana setup

Go to `http://localhost:5601`

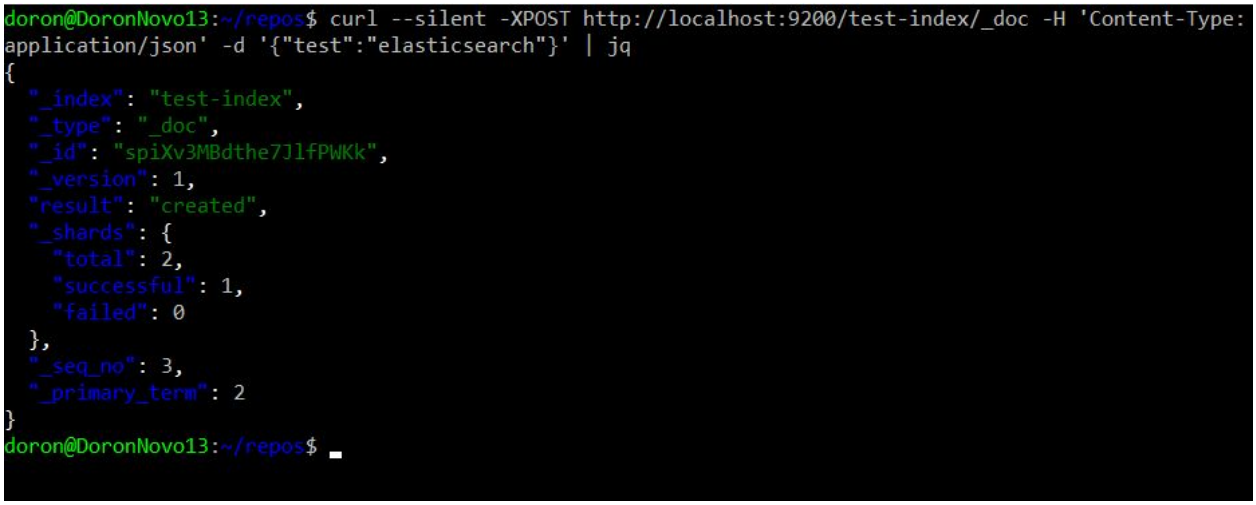


Testing the setup:

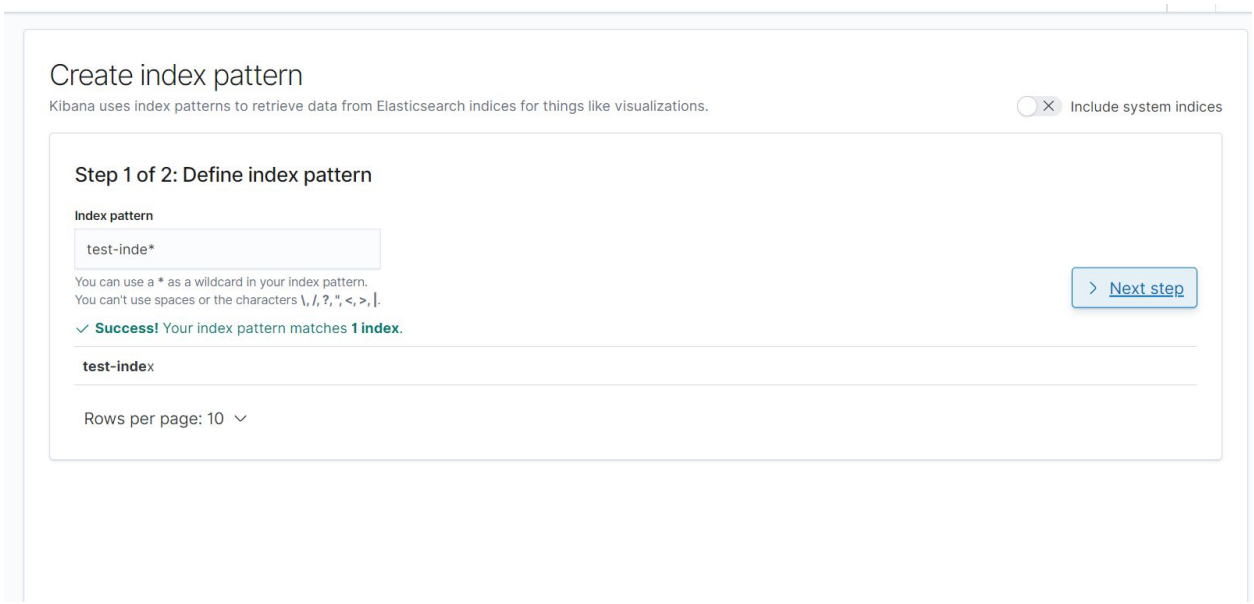
curl --silent -XPOST http://localhost:9200/test-index/_doc -H 'Content-Type: application/json' -d '{"test":"elasticsearch"}' | jq

The script above will index the data on test-index

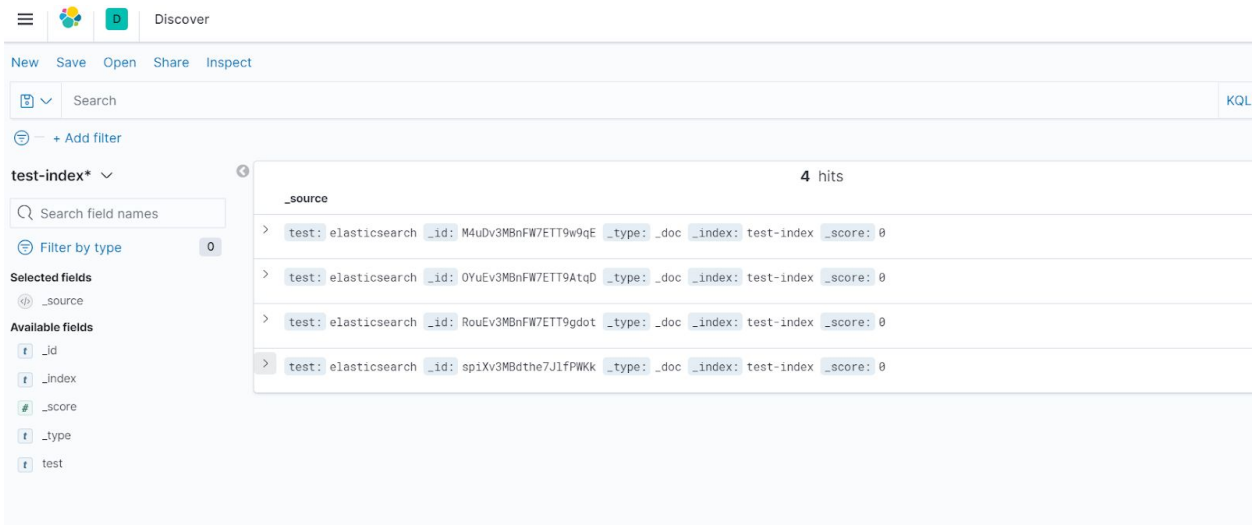
The response is:

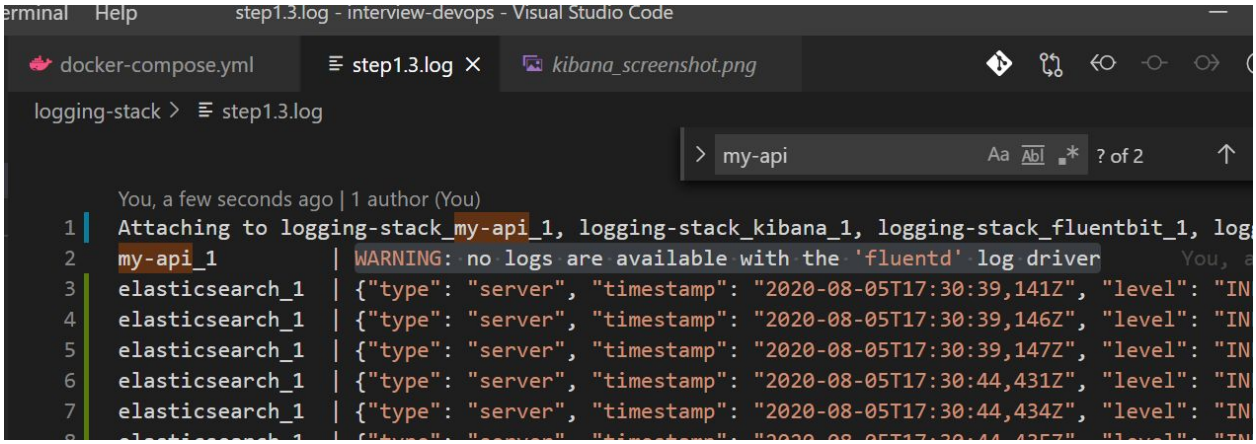


Then you can check the kibana on the discovery tab or this URLhttp://localhost:5601/app/kibana#/management/kibana/index_pattern you will see test-index is registered.



Go to the next step, and click create index pattern button. Go back to discover tab and you will see your data there.

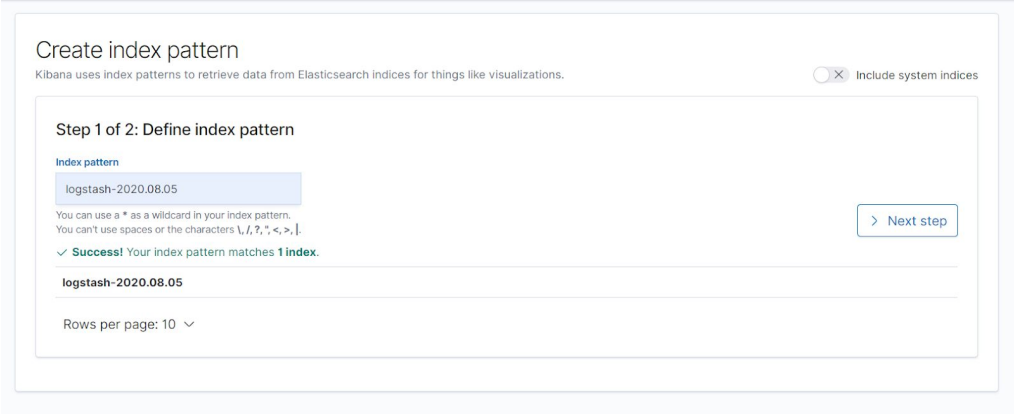




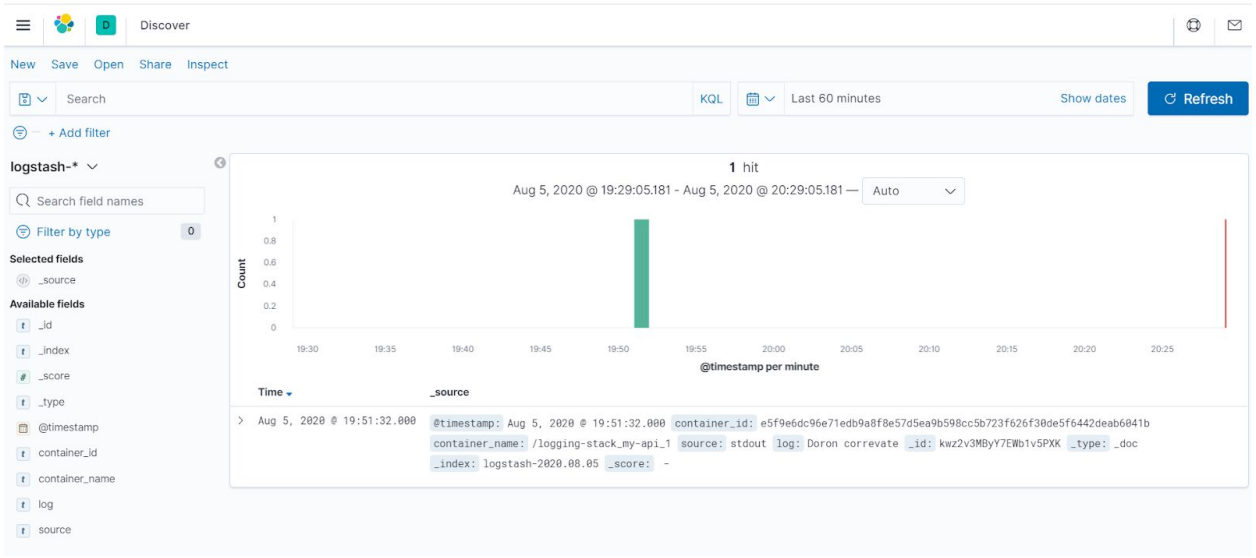
For some reason I couldn't make the app output logstash logs.
I added to the logging options: `fluentd-address: 127.0.0.1:24224`

```
my-api:
  image: my-api:latest
  depends_on:
    - fluentbit
  logging:
    driver: fluentd
    options:
      fluentd-address: 127.0.0.1:24224
      tag: my-api
  networks:
    - correvate
```

Logstash then appeared!



Also the log from the app:



2-Devops process

Overview

How would you get a Spring Boot web application from a development environment into a production environment from scratch?

Goals

1. Setup AWS CLI, eksctl and kubectl
2. Deploy a kubernetes cluster with an on-demand node group
3. Add spot node groups to your EKS cluster
4. Install AWS node termination handler
5. Deploy the kubernetes cluster-autoscaler
6. Deploy prometheus dashboard for cluster overview
7. Create workspace production
8. Create ECR repo for Angular SPA frontend
9. Create ECR repo for Spring boot backend
10. Attach policy for pushing and pulling image
11. Create a push-pull IAM policy

12. Create image in ECR for Angular SPA frontend & Spring boot backend
13. Deploy dockerized Angular SPA frontend with a loadbalancer
14. Deploy dockerized Spring boot backend image
15. Deploy dockerized mongodb slaves
16. Deploy dockerized mongodb master
17. Deploy secrets and credentials as a sidecar pod
18. Create a user in AWS IAM for jenkins and authenticate it
19. Create a user in Github for Jenkins and authenticate it.
20. Create a policy for the user with assume role capabilities
21. Setup HA Jenkins with slaves
22. Create a job
23. Assume role and deploy to production
24. Add loadbalancer to route 53

Specifications

For the above to be functional app should be dockerized

Infra-as-code

Use CloudFormation for IAC. CF creates all appropriate services and connections and is easy to maintain in a repo.

CI/CD

Deployment should come naturally after successful build and test.

Build and test are run on Jenkins HA instance after a checkout upon a new pull request trigger from Github. If build and tests are successful we can deploy to Dev and QA envs and to Production if there's sufficient test coverage, and static code analysis, stress test for the service(s) etc.

High-availability

As services are dockerized, kubernetes handles the startup and teardown of the pods. Also embrace the blue-green deployment of kubernetes with jenkins rollout methodology.

Milestones

I. Create test coverage to identify holes in the loop

In order to assess the quality of the code base and to prevent friction there needs to be a proper test coverage. Tests can be unit, integration, E2E, acceptance. This will be a gate for deployment.

II. Dockerize components for highly available highly scalable deployment

With every successful build update docker image this is done via a Jenkins job as well.

III. Build as changes occur

Jenkins should sniff a change in the repo and initiate a build, test and deploy. Jenkins jobs should be IAC as well and maintained in a repo. Alternatively grant the ability to deploy to environments by the product owner.

Jenkins itself should use a kubernetes cluster for both high availability and high scalability.