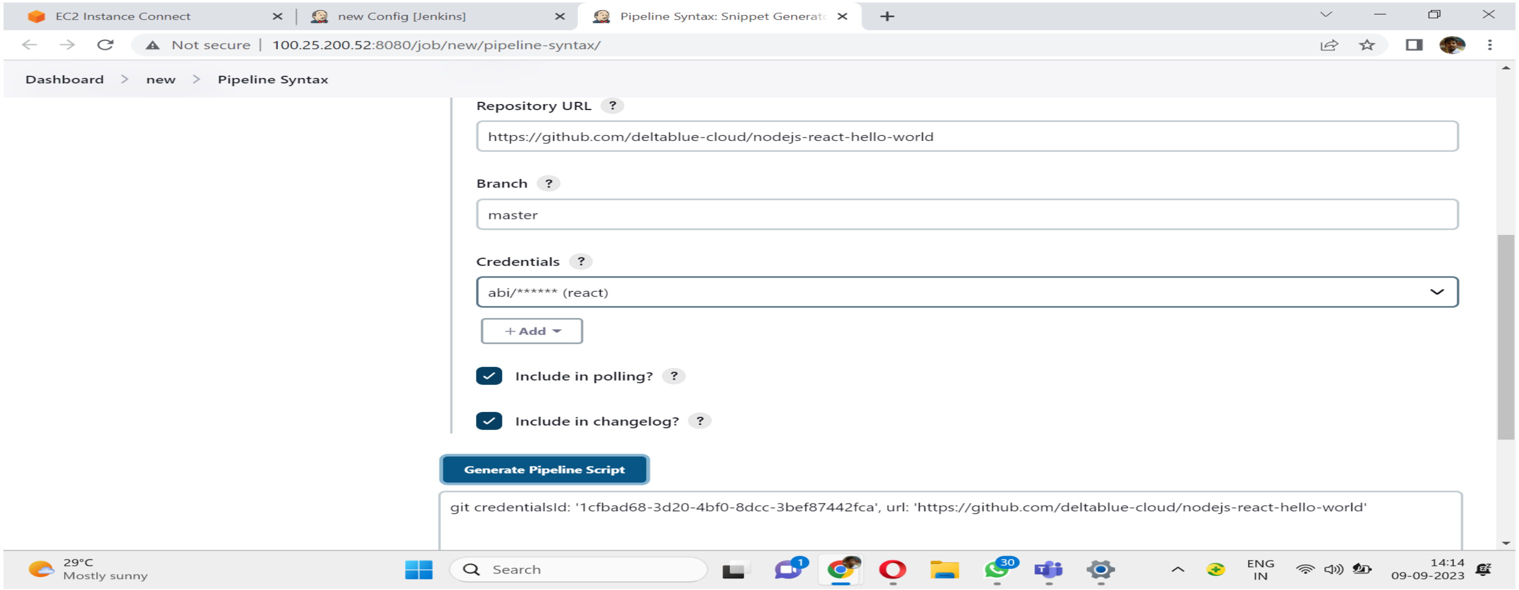
1.Jenkins pipeline for deploying react app

Step 1 clone the github repo
Step 2 Start & build the react app by using below jenkins declarative pipeline

```
     pipeline {
    agent any
    stages {
        stage('Checkout') {
            steps {
                git credentialsId: '1cfbad68-3d20-4bf0-8dcc-3bef87442fca', url:
  'https://github.com/deltablue-cloud/nodejs-react-hello-world.git'
            }
        }
        stage('Build and Run') {
            steps {
                sh 'npm install'

                sh 'nohup npm run start > start.log 2>&1 &'

                sh 'nohup npm run build > build.log 2>&1 &'
        }
    }
  }
}
```


EC2 Instance Connect
new Config [Jenkins]
Pipeline Syntax: Snippet Generat
Not secure | 100.25.200.52:8080/job/new/pipeline-syntax/
Dashboard > new > Pipeline Syntax
+ Add
Generate Pipeline Script
29°C
Mostly sunny
Search
ENG
IN
14:14
09-09-2023

Repository URL ?
https://github.com/deltablue-cloud/nodejs-react-hello-world

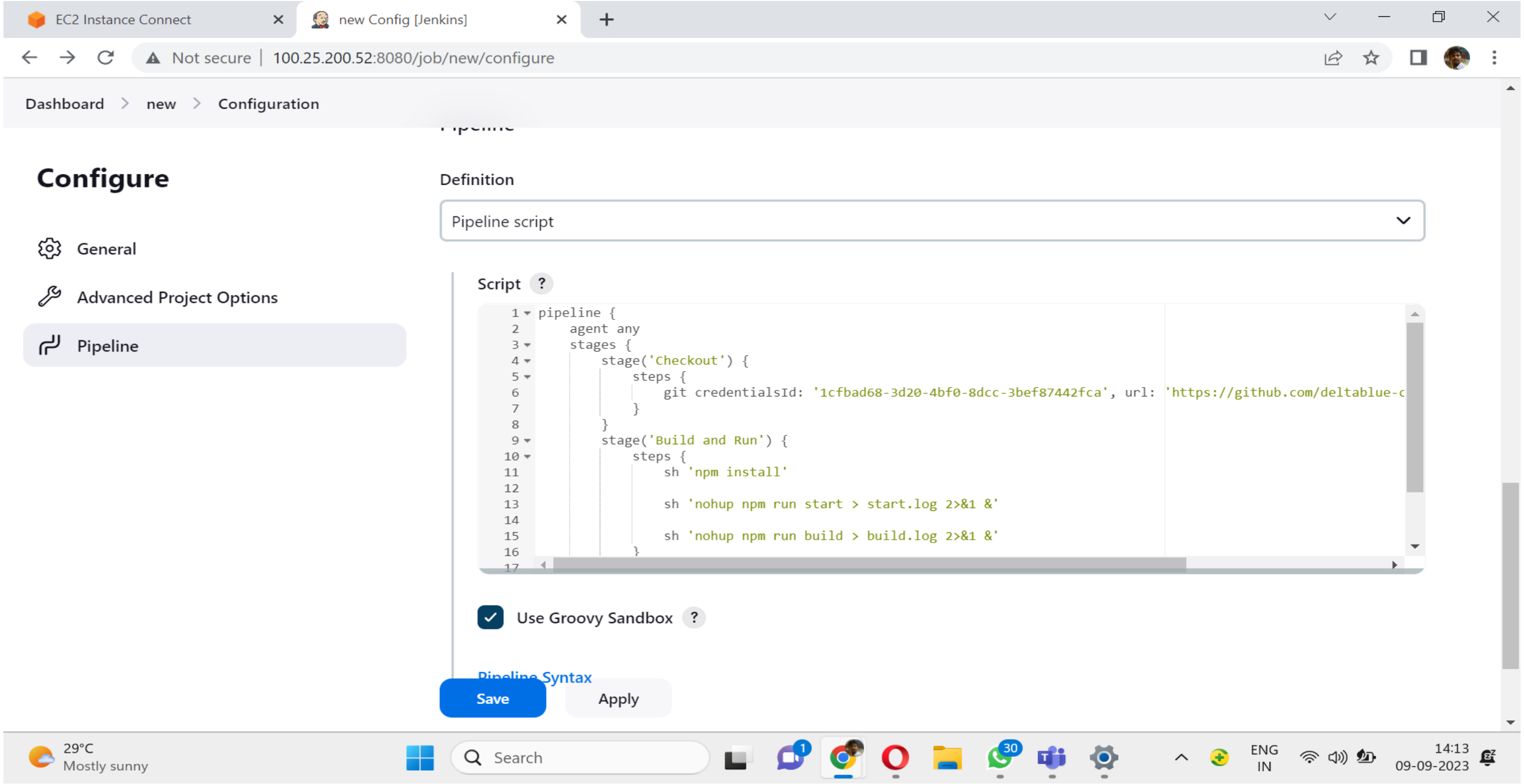
Branch ?
master

Credentials ?
abi/****** (react)

☑ Include in polling? ?

☑ Include in changelog? ?

git credentialsId: '1cfbad68-3d20-4bf0-8dcc-3bef87442fca', url: 'https://github.com/deltablue-cloud/nodejs-react-hello-world'



Dashboard > new > Configuration

Configure

Definition

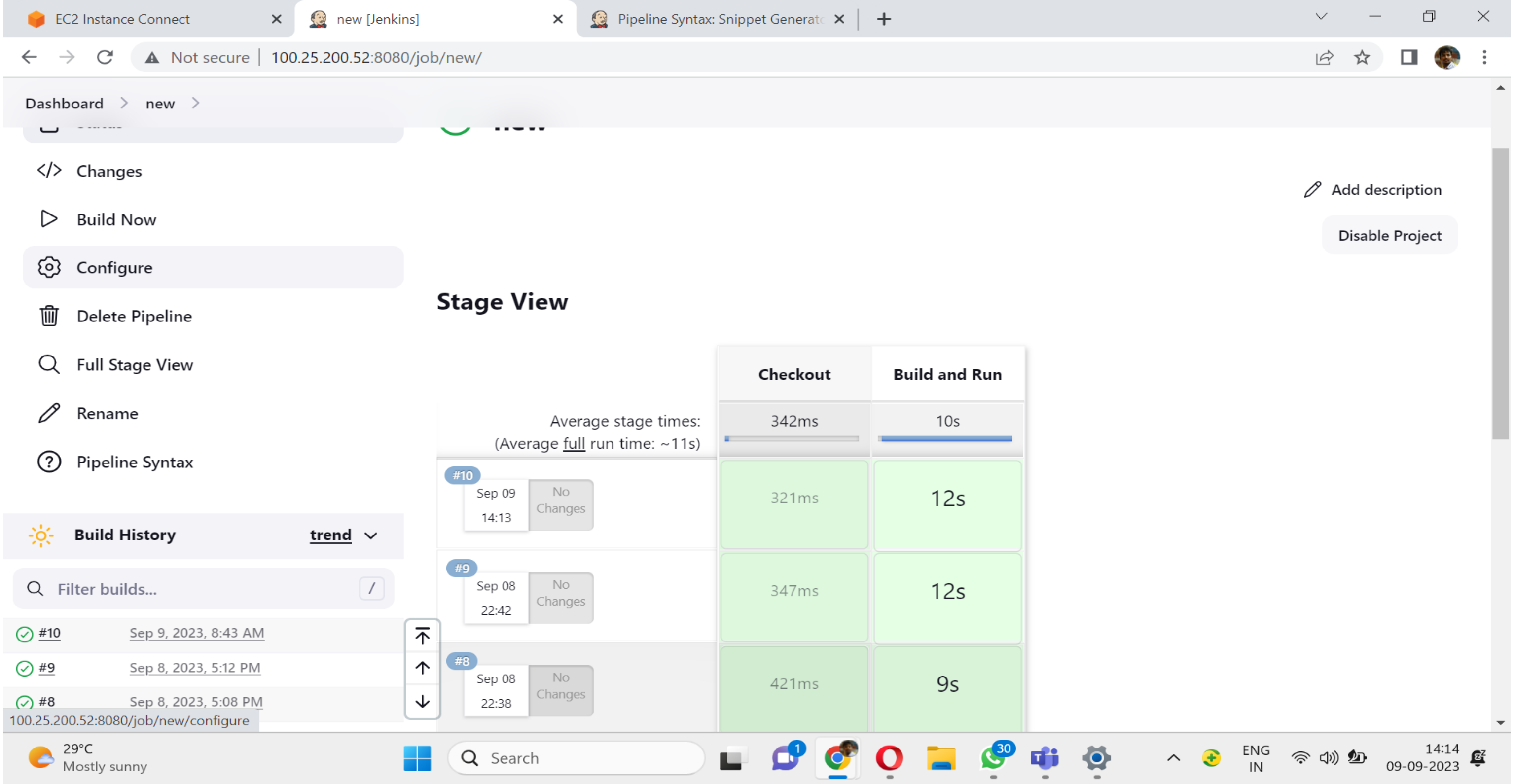
Pipeline script

Script ?

```
pipeline {
    agent any
    stages {
        stage('Checkout') {
            steps {
                git credentialsId: '1cfbad68-3d20-4bf0-8dcc-3bef87442fca', url: 'https://github.com/deltablue-c
            }
        }
        stage('Build and Run') {
            steps {
                sh 'npm install'

                sh 'nohup npm run start > start.log 2>&1 &'

                sh 'nohup npm run build > build.log 2>&1 &'
            }
```

☑ Use Groovy Sandbox ?

Pipeline Syntax


EC2 Instance Connect
new [Jenkins]
Pipeline Syntax: Snippet Generat
Not secure | 100.25.200.52:8080/job/new/

Dashboard > new >

Stage View

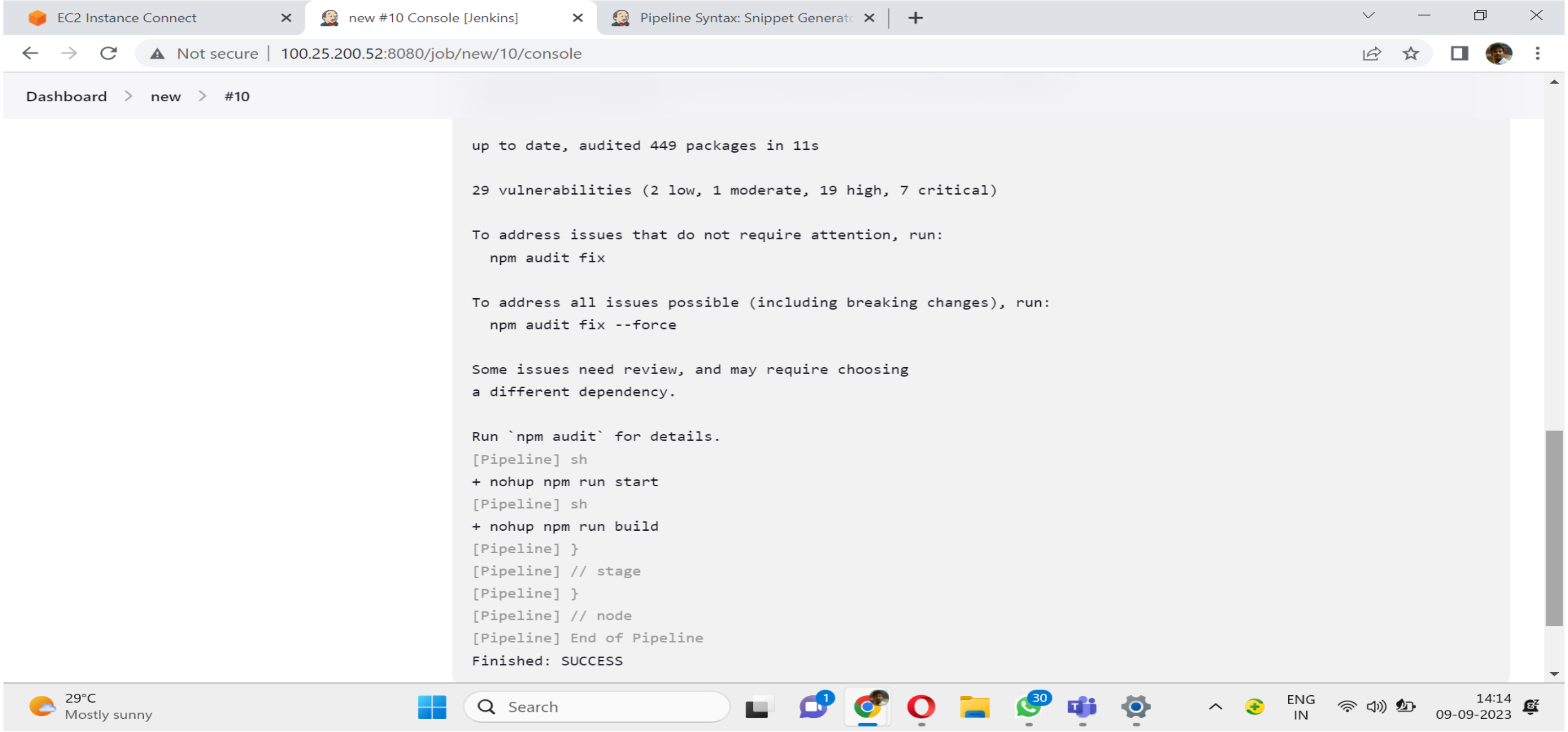
	Checkout	Build and Run
Average stage times: (Average full run time: ~11s)	342ms	10s
#10 Sep 09 14:13 No Changes	321ms	12s
#9 Sep 08 22:42 No Changes	347ms	12s
#8 Sep 08 22:38 No Changes	421ms	9s

Build History trend

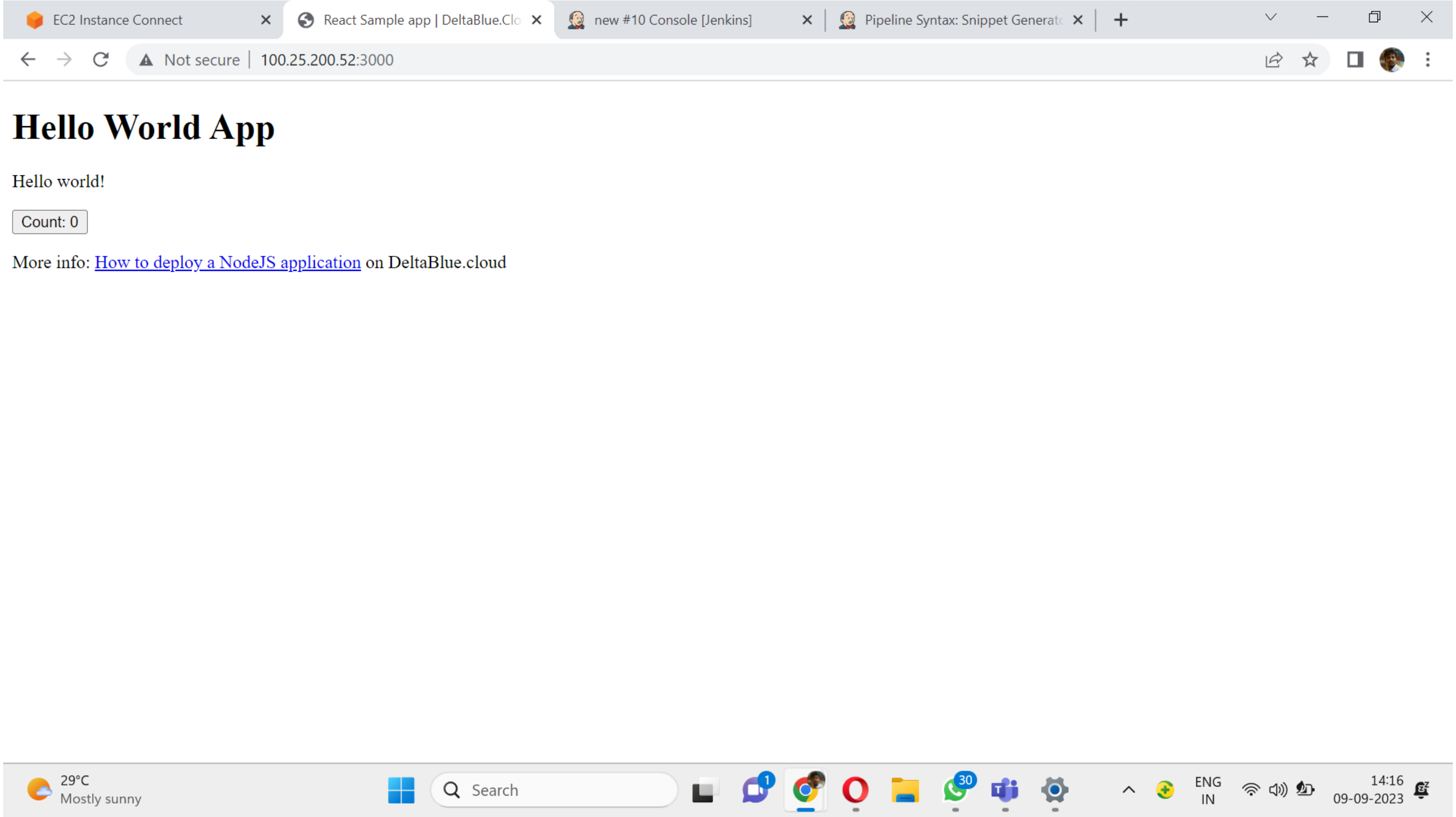
Filter builds... /

#10	Sep 9, 2023, 8:43 AM
#9	Sep 8, 2023, 5:12 PM
#8	Sep 8, 2023, 5:08 PM

100.25.200.52:8080/job/new/configure


EC2 Instance Connect
new #10 Console [Jenkins]
Pipeline Syntax: Snippet Generat
Not secure
100.25.200.52:8080/job/new/10/console
Dashboard > new > #10
29°C
Mostly sunny
Search
ENG
IN
14:14
09-09-2023

```
up to date, audited 449 packages in 11s

29 vulnerabilities (2 low, 1 moderate, 19 high, 7 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues possible (including breaking changes), run:
  npm audit fix --force

Some issues need review, and may require choosing
a different dependency.

Run `npm audit` for details.
[Pipeline] sh
+ nohup npm run start
[Pipeline] sh
+ nohup npm run build
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```



Hello World App

Hello world!

Count: 0

More info: How to deploy a NodeJS application on DeltaBlue.cloud

Create a docker-compose stack for kafka and its components and create a kafka topic.

Step 1 to create docker-compose file for Kafka & zookeeper

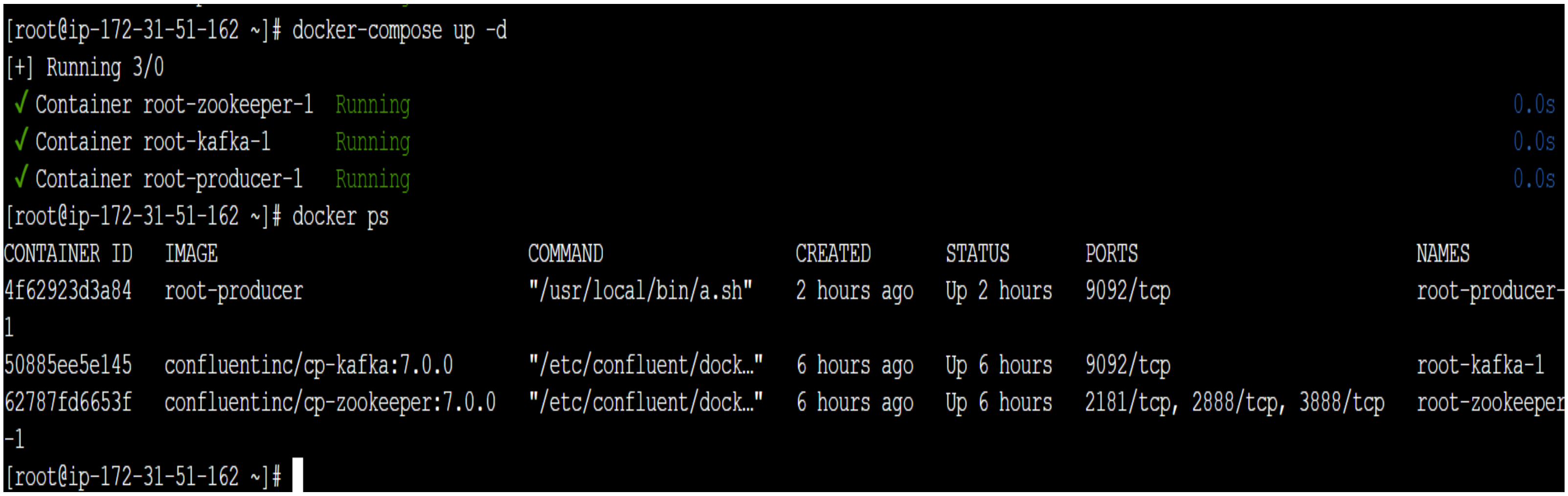
```
version: '3'
services:
  zookeeper:
    image: confluentinc/cp-zookeeper:7.0.0
    user: "root"
    environment:
      ZOOKEEPER_CLIENT_PORT: 2181
      ZOOKEEPER_TICK_TIME: 2000
  kafka:
    image: confluentinc/cp-kafka:7.0.0
    user: "root"
    depends_on:
      - zookeeper
    environment:
      KAFKA_BROKER_ID: 1
      KAFKA_LISTENERS: PLAINTEXT://:9092
      KAFKA_ADVERTISED_LISTENERS: PLAINTEXT://kafka:9092  # Use KAFKA_ADVERTISED_LISTENERS
      KAFKA_AUTO_CREATE_TOPICS_ENABLE: "true"
      KAFKA_OFFSETS_TOPIC_REPLICATION_FACTOR: 1
      KAFKA_ZOOKEEPER_CONNECT: zookeeper:2181

  producer:
    build:
      context: ./
    depends_on:
      - kafka
```

Step 2 start the services mentioned in compose.yml file by using command

```
docker-compose up -d
```

Step 3 check the container is up & running by using

```
docker ps
```

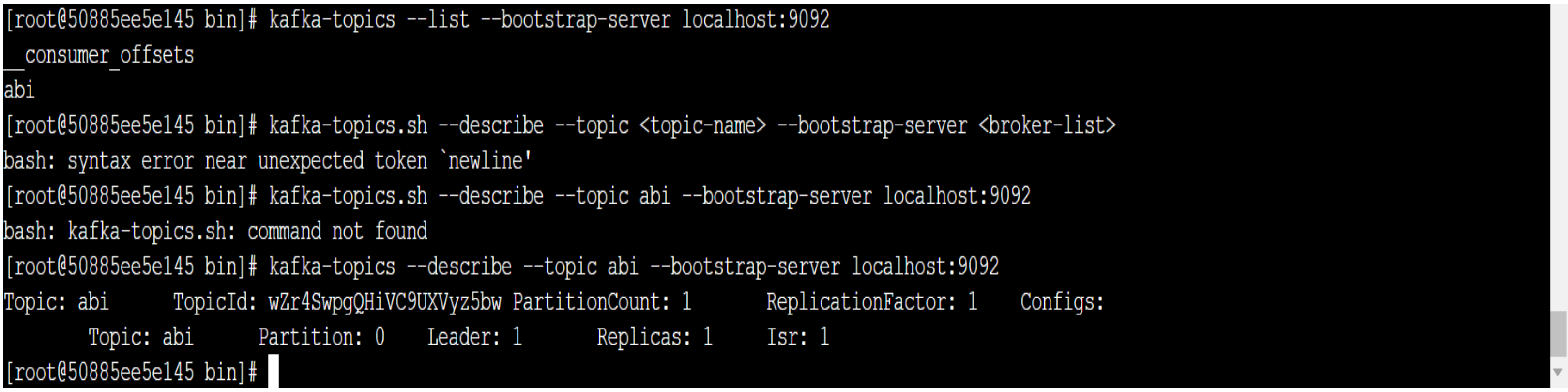


```
[root@ip-172-31-51-162 ~]# docker-compose up -d
[+] Running 3/0
 ✓ Container root-zookeeper-1  Running                                                                    0.0s
 ✓ Container root-kafka-1      Running                                                                    0.0s
 ✓ Container root-producer-1   Running                                                                    0.0s
[root@ip-172-31-51-162 ~]# docker ps
CONTAINER ID   IMAGE                          COMMAND                  CREATED       STATUS       PORTS                          NAMES
4f62923d3a84   root-producer                  "/usr/local/bin/a.sh"    2 hours ago   Up 2 hours   9092/tcp                       root-producer-
1
50885ee5e145   confluentinc/cp-kafka:7.0.0    "/etc/confluent/dock…"   6 hours ago   Up 6 hours   9092/tcp                       root-kafka-1
62787fd6653f   confluentinc/cp-zookeeper:7.0.0   "/etc/confluent/dock…"   6 hours ago   Up 6 hours   2181/tcp, 2888/tcp, 3888/tcp   root-zookeeper
-1
[root@ip-172-31-51-162 ~]#
```

Step 4   access inside the container of kafka service by using the command
    docker exec -it <container_id> /bin/bash

Step 5 change the advertised.listeners(PLAINTEXT:// <ip-address>:9092) to host name or ip address in server properties by using commands
    Vim /etc/kafka/server.properties
    advertised.listeners=PLAINTEXT://100.25.200.52:9092

Step 6 Create the topic by using the command
    /bin/kafka-topics.sh --bootstrap-server localhost:9092 --create --topic <mytopic> --partitions 1 --replication-factor1



```
[root@50885ee5e145 bin]# kafka-topics --list --bootstrap-server localhost:9092
__consumer_offsets
abi
[root@50885ee5e145 bin]# kafka-topics.sh --describe --topic <topic-name> --bootstrap-server <broker-list>
bash: syntax error near unexpected token `newline'
[root@50885ee5e145 bin]# kafka-topics.sh --describe --topic abi --bootstrap-server localhost:9092
bash: kafka-topics.sh: command not found
[root@50885ee5e145 bin]# kafka-topics --describe --topic abi --bootstrap-server localhost:9092
Topic: abi      TopicId: wZr4SwpgQHiVC9UXVyz5bw PartitionCount: 1       ReplicationFactor: 1    Configs:
        Topic: abi      Partition: 0    Leader: 1       Replicas: 1     Isr: 1
[root@50885ee5e145 bin]#
```

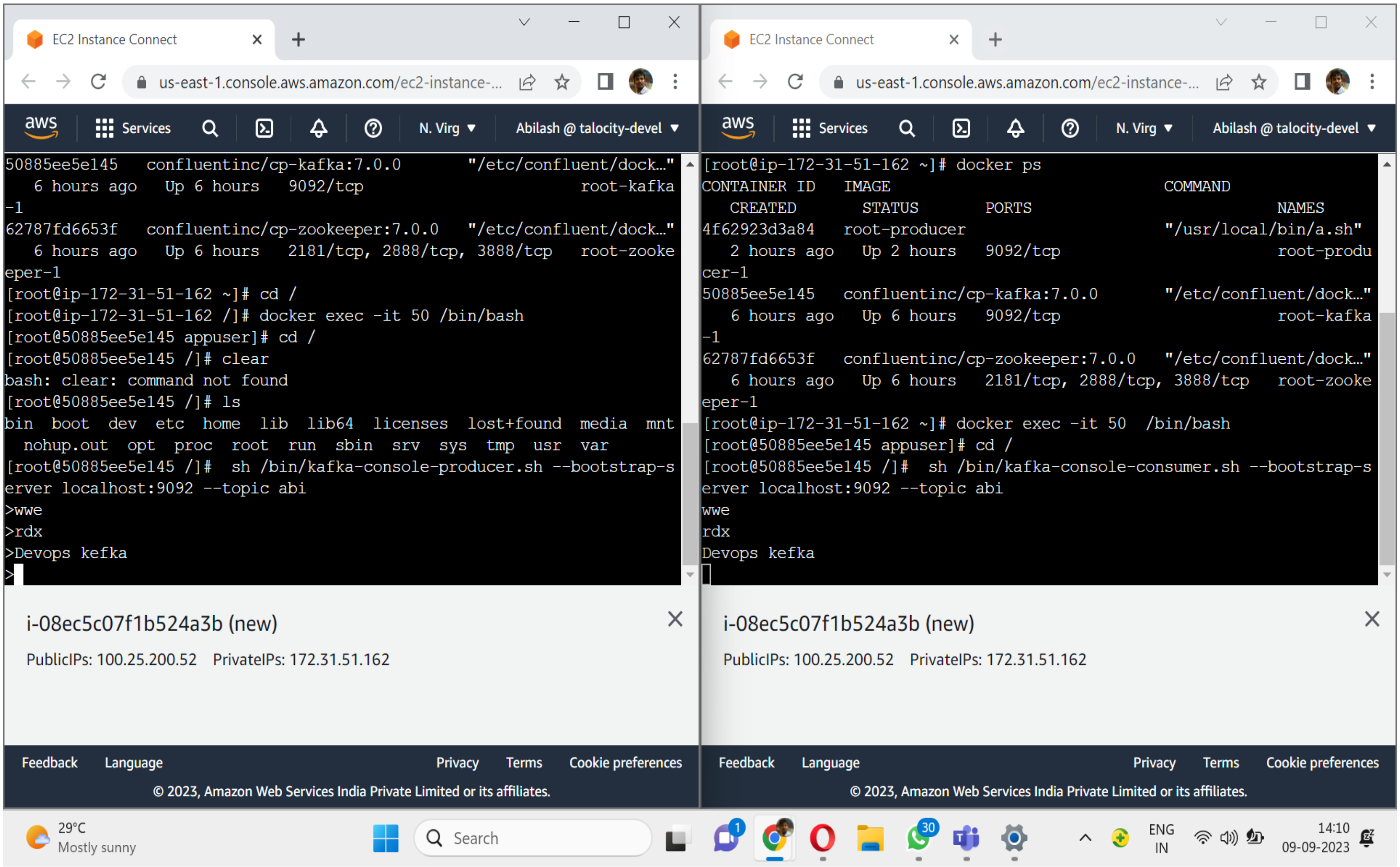
Above image Kafka topic is successfully created.

Step 7 create a Kafka producer in Apache Kafka, you can use Kafka's command-line producer tool, kafka-console-producer

  sh /bin/kafka-console-producer.sh --bootstrap-server localhost:9092 --topic abi

Step 8 create a Kafka consumer in Apache Kafka, to receive the data by using below command

sh /bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic abi



```
50885ee5e145   confluentinc/cp-kafka:7.0.0       "/etc/confluent/dock…"
   6 hours ago    Up 6 hours    9092/tcp                       root-kafka
-1
62787fd6653f   confluentinc/cp-zookeeper:7.0.0   "/etc/confluent/dock…"
   6 hours ago    Up 6 hours    2181/tcp, 2888/tcp, 3888/tcp   root-zooke
eper-1
[root@ip-172-31-51-162 ~]# cd /
[root@ip-172-31-51-162 /]# docker exec -it 50 /bin/bash
[root@50885ee5e145 appuser]# cd /
[root@50885ee5e145 /]# clear
bash: clear: command not found
[root@50885ee5e145 /]# ls
bin  boot  dev  etc  home  lib  lib64  licenses  lost+found  media  mnt
 nohup.out  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
[root@50885ee5e145 /]#  sh /bin/kafka-console-producer.sh --bootstrap-s
erver localhost:9092 --topic abi
>wwe
>rdx
>Devops kefka
>
```

i-08ec5c07f1b524a3b (new)

PublicIPs: 100.25.200.52   PrivateIPs: 172.31.51.162

© 2023, Amazon Web Services India Private Limited or its affiliates.

```
[root@ip-172-31-51-162 ~]# docker ps
CONTAINER ID   IMAGE                               COMMAND
   CREATED        STATUS        PORTS                          NAMES
4f62923d3a84   root-producer                       "/usr/local/bin/a.sh"
   2 hours ago    Up 2 hours    9092/tcp                       root-produ
cer-1
50885ee5e145   confluentinc/cp-kafka:7.0.0         "/etc/confluent/dock…"
   6 hours ago    Up 6 hours    9092/tcp                       root-kafka
-1
62787fd6653f   confluentinc/cp-zookeeper:7.0.0     "/etc/confluent/dock…"
   6 hours ago    Up 6 hours    2181/tcp, 2888/tcp, 3888/tcp   root-zooke
eper-1
[root@ip-172-31-51-162 ~]# docker exec -it 50  /bin/bash
[root@50885ee5e145 appuser]# cd /
[root@50885ee5e145 /]#  sh /bin/kafka-console-consumer.sh --bootstrap-s
erver localhost:9092 --topic abi
wwe
rdx
Devops kefka
```

i-08ec5c07f1b524a3b (new)

PublicIPs: 100.25.200.52   PrivateIPs: 172.31.51.162

© 2023, Amazon Web Services India Private Limited or its affiliates.

In the above images can see that the data is been sent in the producer side & it is received in consumer side

PTO

Create a simple shell that sends 1000 messages to this kafka topic. Create another script to consume these messages.

```
#!/bin/bash
# Kafka topic to produce messages to
TOPIC=<topic-name>

# Number of messages to produce
NUM_MESSAGES=1000

# Kafka broker address
BROKER=localhost:9092

# Produce messages
for ((i=1; i<=NUM_MESSAGES; i++)); do
  a="Message $i"
  echo $a | kafka-console-producer.sh --broker-list $BROKER --topic $TOPIC
done
```

Save the script & configure in the Dockerfile & in above docker-compose file

```
FROM confluentinc/cp-kafka:7.0.0
USER root
COPY a.sh /bin/a.sh
RUN chmod +x /bin/a.sh
CMD ["/bin/a.sh"]
```