1. **RUNNING KAFKA ON COMMAND LINE:**

- **Install kafka from the website: https://kafka.apache.org/downloads**

- This command calculates the **SHA-512 checksum** of the `kafka_2.13-3.9.1` file.:

  You can compare this hash with a **published SHA-512 hash** from the Kafka website to:

  - Verify **file integrity** (i.e., the file wasn't corrupted during download).

  - Ensure **authenticity** (i.e., it wasn't tampered with or modified).

  shasum -a 512 kafka_2.13-3.9.1

- **extracts** the contents of the `kafka_2.13-3.9.1.tgz` archive file.
  tar -xvf kafka_2.13-3.9.1.tgz

2. **START ZOOKEEPER:**
   - Start zookeeper in a new tab.

     Move to the kafka directory where we unpacked the .tgz file.

     For example: *cd ~/Downloads/kafka_2.13-3.9.1*
   - Ls shows these files:

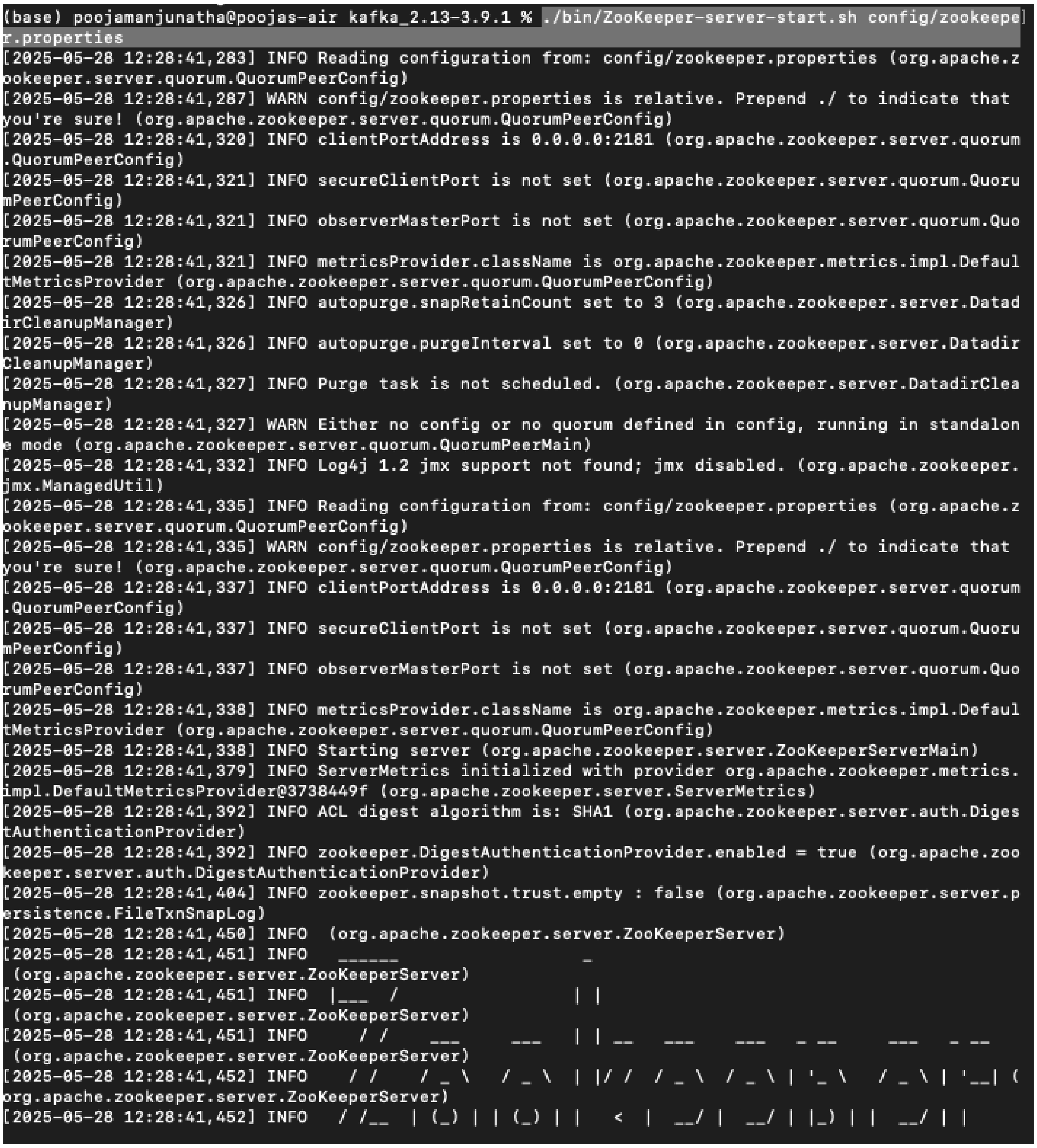
```
(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 % ls
LICENSE         bin         libs        site-docs
NOTICE          config      licenses
```

1. All of the libraries are in the libs directory
2. Example configuration files in config directory
3. Bin directory has files that help run kafka, zookeeper and admin commands.

ALWAYS RUN ZOOKEEPER FIRST AND END IT LAST.

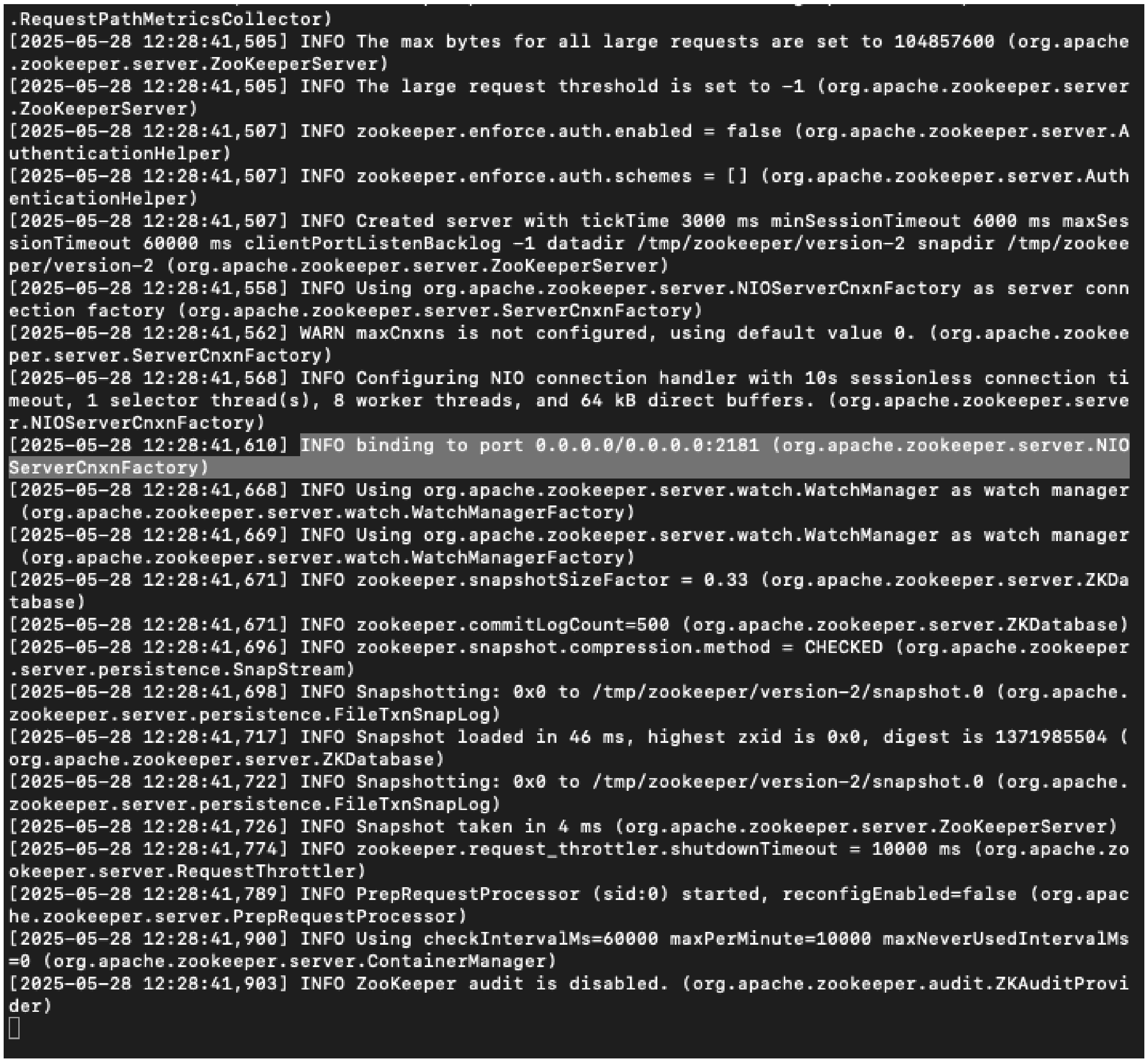
./bin/ZooKeeper-server-start.sh config/zookeeper.properties

This command starts the zookeeper and we pass default values present in the config file for it to be up and running.



```
(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 % ./bin/ZooKeeper-server-start.sh config/zookeepe
r.properties
[2025-05-28 12:28:41,283] INFO Reading configuration from: config/zookeeper.properties (org.apache.z
ookeeper.server.quorum.QuorumPeerConfig)
[2025-05-28 12:28:41,287] WARN config/zookeeper.properties is relative. Prepend ./ to indicate that
you're sure! (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
[2025-05-28 12:28:41,320] INFO clientPortAddress is 0.0.0.0:2181 (org.apache.zookeeper.server.quorum
.QuorumPeerConfig)
[2025-05-28 12:28:41,321] INFO secureClientPort is not set (org.apache.zookeeper.server.quorum.Quoru
mPeerConfig)
[2025-05-28 12:28:41,321] INFO observerMasterPort is not set (org.apache.zookeeper.server.quorum.Quo
rumPeerConfig)
[2025-05-28 12:28:41,321] INFO metricsProvider.className is org.apache.zookeeper.metrics.impl.Defaul
tMetricsProvider (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
[2025-05-28 12:28:41,326] INFO autopurge.snapRetainCount set to 3 (org.apache.zookeeper.server.Datad
irCleanupManager)
[2025-05-28 12:28:41,326] INFO autopurge.purgeInterval set to 0 (org.apache.zookeeper.server.Datadir
CleanupManager)
[2025-05-28 12:28:41,327] INFO Purge task is not scheduled. (org.apache.zookeeper.server.DatadirClea
nupManager)
[2025-05-28 12:28:41,327] WARN Either no config or no quorum defined in config, running in standalon
e mode (org.apache.zookeeper.server.quorum.QuorumPeerMain)
[2025-05-28 12:28:41,332] INFO Log4j 1.2 jmx support not found; jmx disabled. (org.apache.zookeeper.
jmx.ManagedUtil)
[2025-05-28 12:28:41,335] INFO Reading configuration from: config/zookeeper.properties (org.apache.z
ookeeper.server.quorum.QuorumPeerConfig)
[2025-05-28 12:28:41,335] WARN config/zookeeper.properties is relative. Prepend ./ to indicate that
you're sure! (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
[2025-05-28 12:28:41,337] INFO clientPortAddress is 0.0.0.0:2181 (org.apache.zookeeper.server.quorum
.QuorumPeerConfig)
[2025-05-28 12:28:41,337] INFO secureClientPort is not set (org.apache.zookeeper.server.quorum.Quoru
mPeerConfig)
[2025-05-28 12:28:41,337] INFO observerMasterPort is not set (org.apache.zookeeper.server.quorum.Quo
rumPeerConfig)
[2025-05-28 12:28:41,338] INFO metricsProvider.className is org.apache.zookeeper.metrics.impl.Defaul
tMetricsProvider (org.apache.zookeeper.server.quorum.QuorumPeerConfig)
[2025-05-28 12:28:41,338] INFO Starting server (org.apache.zookeeper.server.ZooKeeperServerMain)
[2025-05-28 12:28:41,379] INFO ServerMetrics initialized with provider org.apache.zookeeper.metrics.
impl.DefaultMetricsProvider@3738449f (org.apache.zookeeper.server.ServerMetrics)
[2025-05-28 12:28:41,392] INFO ACL digest algorithm is: SHA1 (org.apache.zookeeper.server.auth.Diges
tAuthenticationProvider)
[2025-05-28 12:28:41,392] INFO zookeeper.DigestAuthenticationProvider.enabled = true (org.apache.zoo
keeper.server.auth.DigestAuthenticationProvider)
[2025-05-28 12:28:41,404] INFO zookeeper.snapshot.trust.empty : false (org.apache.zookeeper.server.p
ersistence.FileTxnSnapLog)
[2025-05-28 12:28:41,450] INFO  (org.apache.zookeeper.server.ZooKeeperServer)
[2025-05-28 12:28:41,451] INFO   ______                  _                                          
 (org.apache.zookeeper.server.ZooKeeperServer)
[2025-05-28 12:28:41,451] INFO  |___  /                 | |                                         
 (org.apache.zookeeper.server.ZooKeeperServer)
[2025-05-28 12:28:41,451] INFO     / /    ___     ___   | | __   ___    ___   _ __     ___   _ __   
 (org.apache.zookeeper.server.ZooKeeperServer)
[2025-05-28 12:28:41,452] INFO    / /    / _ \   / _ \  | |/ /  / _ \  / _ \ | '_ \   / _ \ | '__| (
org.apache.zookeeper.server.ZooKeeperServer)
[2025-05-28 12:28:41,452] INFO   / /__  | (_) | | (_) | |   <  |  __/ |  __/ | |_) | |  __/ | |
```

What we look for in this file is a message that looks like this:
INFO binding to port 0.0.0.0/0.0.0.0:2181 (org.apache.zookeeper.server.NIOServerCnxnFactory)- which means the zookeeper is up and running

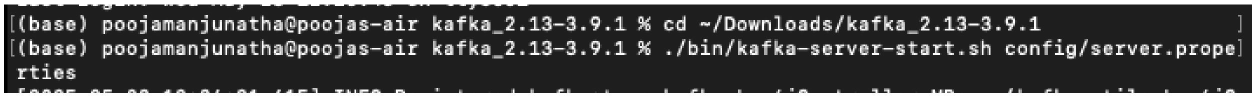


```
.RequestPathMetricsCollector)
[2025-05-28 12:28:41,505] INFO The max bytes for all large requests are set to 104857600 (org.apache
.zookeeper.server.ZooKeeperServer)
[2025-05-28 12:28:41,505] INFO The large request threshold is set to -1 (org.apache.zookeeper.server
.ZooKeeperServer)
[2025-05-28 12:28:41,507] INFO zookeeper.enforce.auth.enabled = false (org.apache.zookeeper.server.A
uthenticationHelper)
[2025-05-28 12:28:41,507] INFO zookeeper.enforce.auth.schemes = [] (org.apache.zookeeper.server.Auth
enticationHelper)
[2025-05-28 12:28:41,507] INFO Created server with tickTime 3000 ms minSessionTimeout 6000 ms maxSes
sionTimeout 60000 ms clientPortListenBacklog -1 datadir /tmp/zookeeper/version-2 snapdir /tmp/zookee
per/version-2 (org.apache.zookeeper.server.ZooKeeperServer)
[2025-05-28 12:28:41,558] INFO Using org.apache.zookeeper.server.NIOServerCnxnFactory as server conn
ection factory (org.apache.zookeeper.server.ServerCnxnFactory)
[2025-05-28 12:28:41,562] WARN maxCnxns is not configured, using default value 0. (org.apache.zookee
per.server.ServerCnxnFactory)
[2025-05-28 12:28:41,568] INFO Configuring NIO connection handler with 10s sessionless connection ti
meout, 1 selector thread(s), 8 worker threads, and 64 kB direct buffers. (org.apache.zookeeper.serve
r.NIOServerCnxnFactory)
[2025-05-28 12:28:41,610] INFO binding to port 0.0.0.0/0.0.0.0:2181 (org.apache.zookeeper.server.NIO
ServerCnxnFactory)
[2025-05-28 12:28:41,668] INFO Using org.apache.zookeeper.server.watch.WatchManager as watch manager
 (org.apache.zookeeper.server.watch.WatchManagerFactory)
[2025-05-28 12:28:41,669] INFO Using org.apache.zookeeper.server.watch.WatchManager as watch manager
 (org.apache.zookeeper.server.watch.WatchManagerFactory)
[2025-05-28 12:28:41,671] INFO zookeeper.snapshotSizeFactor = 0.33 (org.apache.zookeeper.server.ZKDa
tabase)
[2025-05-28 12:28:41,671] INFO zookeeper.commitLogCount=500 (org.apache.zookeeper.server.ZKDatabase)
[2025-05-28 12:28:41,696] INFO zookeeper.snapshot.compression.method = CHECKED (org.apache.zookeeper
.server.persistence.SnapStream)
[2025-05-28 12:28:41,698] INFO Snapshotting: 0x0 to /tmp/zookeeper/version-2/snapshot.0 (org.apache.
zookeeper.server.persistence.FileTxnSnapLog)
[2025-05-28 12:28:41,717] INFO Snapshot loaded in 46 ms, highest zxid is 0x0, digest is 1371985504 (
org.apache.zookeeper.server.ZKDatabase)
[2025-05-28 12:28:41,722] INFO Snapshotting: 0x0 to /tmp/zookeeper/version-2/snapshot.0 (org.apache.
zookeeper.server.persistence.FileTxnSnapLog)
[2025-05-28 12:28:41,726] INFO Snapshot taken in 4 ms (org.apache.zookeeper.server.ZooKeeperServer)
[2025-05-28 12:28:41,774] INFO zookeeper.request_throttler.shutdownTimeout = 10000 ms (org.apache.zo
okeeper.server.RequestThrottler)
[2025-05-28 12:28:41,789] INFO PrepRequestProcessor (sid:0) started, reconfigEnabled=false (org.apac
he.zookeeper.server.PrepRequestProcessor)
[2025-05-28 12:28:41,900] INFO Using checkIntervalMs=60000 maxPerMinute=10000 maxNeverUsedIntervalMs
=0 (org.apache.zookeeper.server.ContainerManager)
[2025-05-28 12:28:41,903] INFO ZooKeeper audit is disabled. (org.apache.zookeeper.audit.ZKAuditProvi
der)
```

3. START KAFKA

- Open a new tab in the same directory and run the command:

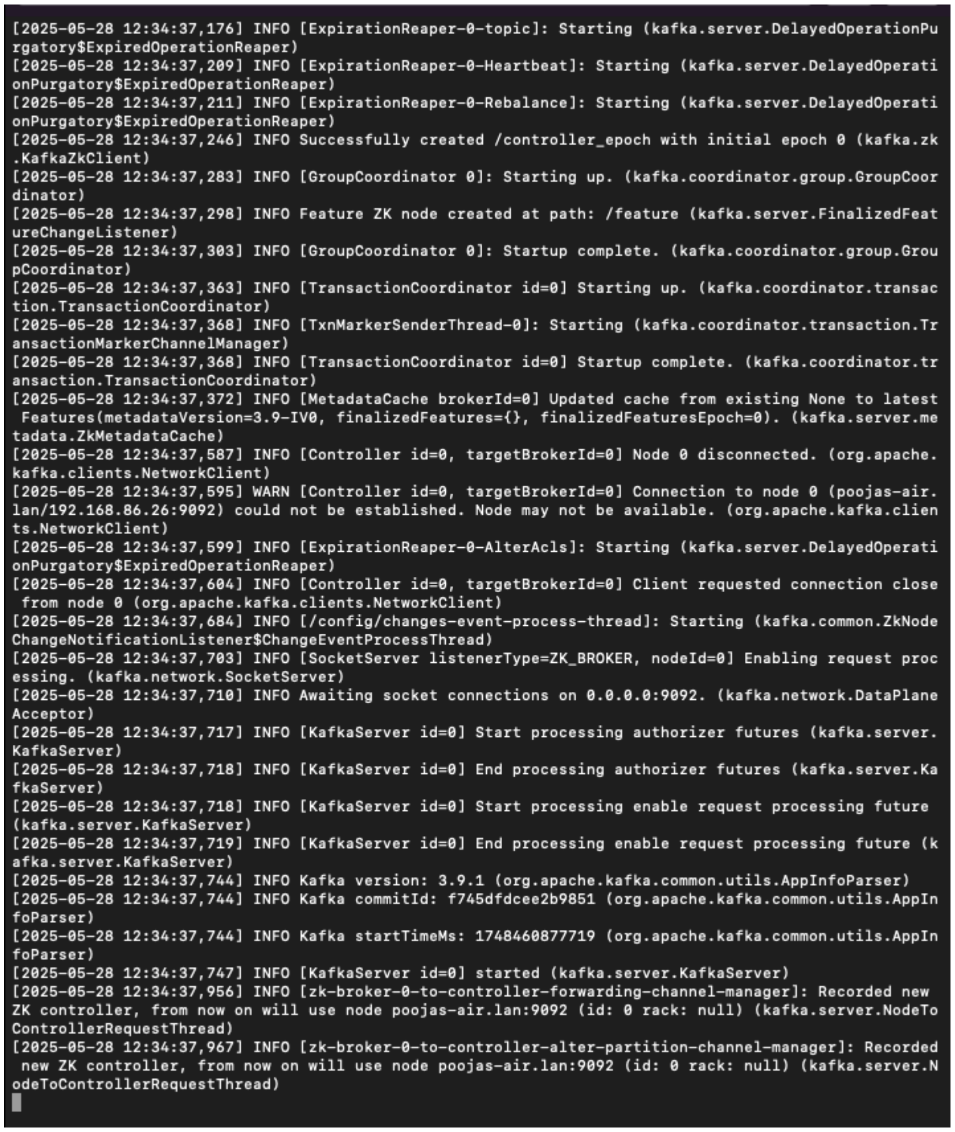
./bin/kafka-server-start.sh config/server.properties



```
[(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 % cd ~/Downloads/kafka_2.13-3.9.1          ]
[(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 % ./bin/kafka-server-start.sh config/server.prope]
 rties
```

- This time we are looking for a message that says :

2025-05-28 12:34:37,717] INFO [KafkaServer id=0] Start processing authorizer futures (kafka.server.KafkaServer)



```
[2025-05-28 12:34:37,176] INFO [ExpirationReaper-0-topic]: Starting (kafka.server.DelayedOperationPu
rgatory$ExpiredOperationReaper)
[2025-05-28 12:34:37,209] INFO [ExpirationReaper-0-Heartbeat]: Starting (kafka.server.DelayedOperati
onPurgatory$ExpiredOperationReaper)
[2025-05-28 12:34:37,211] INFO [ExpirationReaper-0-Rebalance]: Starting (kafka.server.DelayedOperati
onPurgatory$ExpiredOperationReaper)
[2025-05-28 12:34:37,246] INFO Successfully created /controller_epoch with initial epoch 0 (kafka.zk
.KafkaZkClient)
[2025-05-28 12:34:37,283] INFO [GroupCoordinator 0]: Starting up. (kafka.coordinator.group.GroupCoor
dinator)
[2025-05-28 12:34:37,298] INFO Feature ZK node created at path: /feature (kafka.server.FinalizedFeat
ureChangeListener)
[2025-05-28 12:34:37,303] INFO [GroupCoordinator 0]: Startup complete. (kafka.coordinator.group.Grou
pCoordinator)
[2025-05-28 12:34:37,363] INFO [TransactionCoordinator id=0] Starting up. (kafka.coordinator.transac
tion.TransactionCoordinator)
[2025-05-28 12:34:37,368] INFO [TxnMarkerSenderThread-0]: Starting (kafka.coordinator.transaction.Tr
ansactionMarkerChannelManager)
[2025-05-28 12:34:37,368] INFO [TransactionCoordinator id=0] Startup complete. (kafka.coordinator.tr
ansaction.TransactionCoordinator)
[2025-05-28 12:34:37,372] INFO [MetadataCache brokerId=0] Updated cache from existing None to latest
 Features(metadataVersion=3.9-IV0, finalizedFeatures={}, finalizedFeaturesEpoch=0). (kafka.server.me
tadata.ZkMetadataCache)
[2025-05-28 12:34:37,587] INFO [Controller id=0, targetBrokerId=0] Node 0 disconnected. (org.apache.
kafka.clients.NetworkClient)
[2025-05-28 12:34:37,595] WARN [Controller id=0, targetBrokerId=0] Connection to node 0 (poojas-air.
lan/192.168.86.26:9092) could not be established. Node may not be available. (org.apache.kafka.clien
ts.NetworkClient)
[2025-05-28 12:34:37,599] INFO [ExpirationReaper-0-AlterAcls]: Starting (kafka.server.DelayedOperati
onPurgatory$ExpiredOperationReaper)
[2025-05-28 12:34:37,604] INFO [Controller id=0, targetBrokerId=0] Client requested connection close
 from node 0 (org.apache.kafka.clients.NetworkClient)
[2025-05-28 12:34:37,684] INFO [/config/changes-event-process-thread]: Starting (kafka.common.ZkNode
ChangeNotificationListener$ChangeEventProcessThread)
[2025-05-28 12:34:37,703] INFO [SocketServer listenerType=ZK_BROKER, nodeId=0] Enabling request proc
essing. (kafka.network.SocketServer)
[2025-05-28 12:34:37,710] INFO Awaiting socket connections on 0.0.0.0:9092. (kafka.network.DataPlane
Acceptor)
[2025-05-28 12:34:37,717] INFO [KafkaServer id=0] Start processing authorizer futures (kafka.server.
KafkaServer)
[2025-05-28 12:34:37,718] INFO [KafkaServer id=0] End processing authorizer futures (kafka.server.Ka
fkaServer)
[2025-05-28 12:34:37,718] INFO [KafkaServer id=0] Start processing enable request processing future
(kafka.server.KafkaServer)
[2025-05-28 12:34:37,719] INFO [KafkaServer id=0] End processing enable request processing future (k
afka.server.KafkaServer)
[2025-05-28 12:34:37,744] INFO Kafka version: 3.9.1 (org.apache.kafka.common.utils.AppInfoParser)
[2025-05-28 12:34:37,744] INFO Kafka commitId: f745dfdcee2b9851 (org.apache.kafka.common.utils.AppIn
foParser)
[2025-05-28 12:34:37,744] INFO Kafka startTimeMs: 1748460877719 (org.apache.kafka.common.utils.AppIn
foParser)
[2025-05-28 12:34:37,747] INFO [KafkaServer id=0] started (kafka.server.KafkaServer)
[2025-05-28 12:34:37,956] INFO [zk-broker-0-to-controller-forwarding-channel-manager]: Recorded new
ZK controller, from now on will use node poojas-air.lan:9092 (id: 0 rack: null) (kafka.server.NodeTo
ControllerRequestThread)
[2025-05-28 12:34:37,967] INFO [zk-broker-0-to-controller-alter-partition-channel-manager]: Recorded
 new ZK controller, from now on will use node poojas-air.lan:9092 (id: 0 rack: null) (kafka.server.N
odeToControllerRequestThread)
```

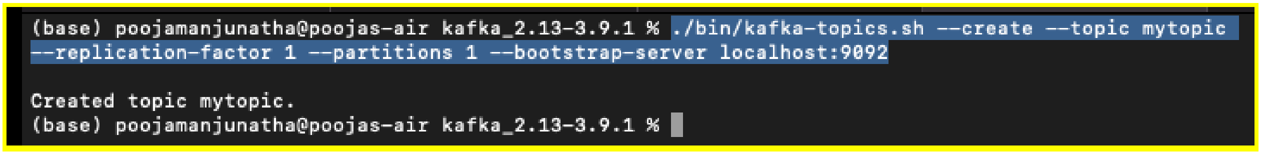
4. CREATE A TOPIC:

- Open a new tab and move to the kafka directory and stat the topic called "mytopic"

replication_factor=1
partitions=1

./bin/kafka-topics.sh --create --topic mytopic --replication-factor 1 --partitions 1 --bootstrap-server localhost:9092

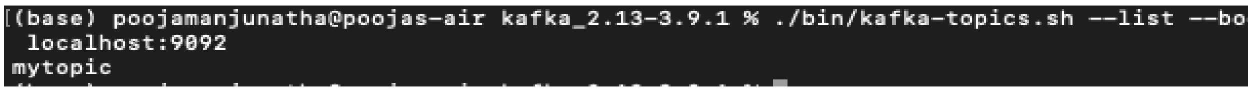
Part	Meaning
`./bin/kafka-topics.sh`	This is the Kafka script used to manage topics (create, list, delete, etc.). It's located inside the `bin/` folder of your Kafka directory.
`--create`	This tells Kafka you want to create a **new topic**.
`--topic mytopic`	This sets the **name** of the topic you're creating. In this case, the topic will be called `mytopic`.
`--replication-factor 1`	This defines **how many replicas** of the topic's data Kafka should maintain across brokers. `1` means no replication (just one copy). Useful for local setups or single-node clusters.
`--partitions 1`	This sets the number of **partitions** the topic will have. Partitions help with parallelism and scalability.
`--bootstrap-server localhost:9092`	This tells the Kafka CLI tool how to connect to your Kafka broker. In this case, it's saying: "Connect to a Kafka broker running on `localhost` at port `9092`."



```
(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 % ./bin/kafka-topics.sh --create --topic mytopic
--replication-factor 1 --partitions 1 --bootstrap-server localhost:9092

Created topic mytopic.
(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 %
```

- List all topics

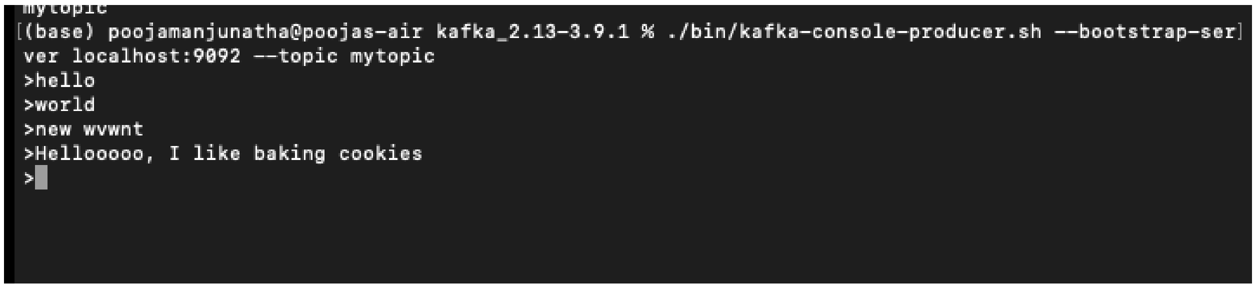
./bin/kafka-topics.sh --list --bootstrap-server localhost:9092
Mytopic


```
[(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 % ./bin/kafka-topics.sh --list --bo
 localhost:9092
mytopic
```

5. **Produce messages to the topic:**
   ./bin/kafka-console-producer.sh --bootstrap-server localhost:9092 --topic mytopic

Once the prompt appears (`>`), type messages line by line. Example:

> \> hello
> \> world
> \> new event

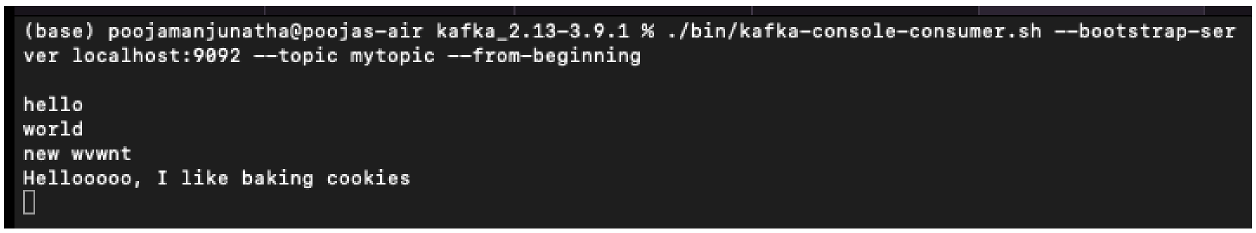

```
mytopic
[(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 % ./bin/kafka-console-producer.sh --bootstrap-ser]
ver localhost:9092 --topic mytopic
>hello
>world
>new wvwnt
>Hellooooo, I like baking cookies
>
```

6. **Consume messages from the topic:**
   ./bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic mytopic --from-beginning

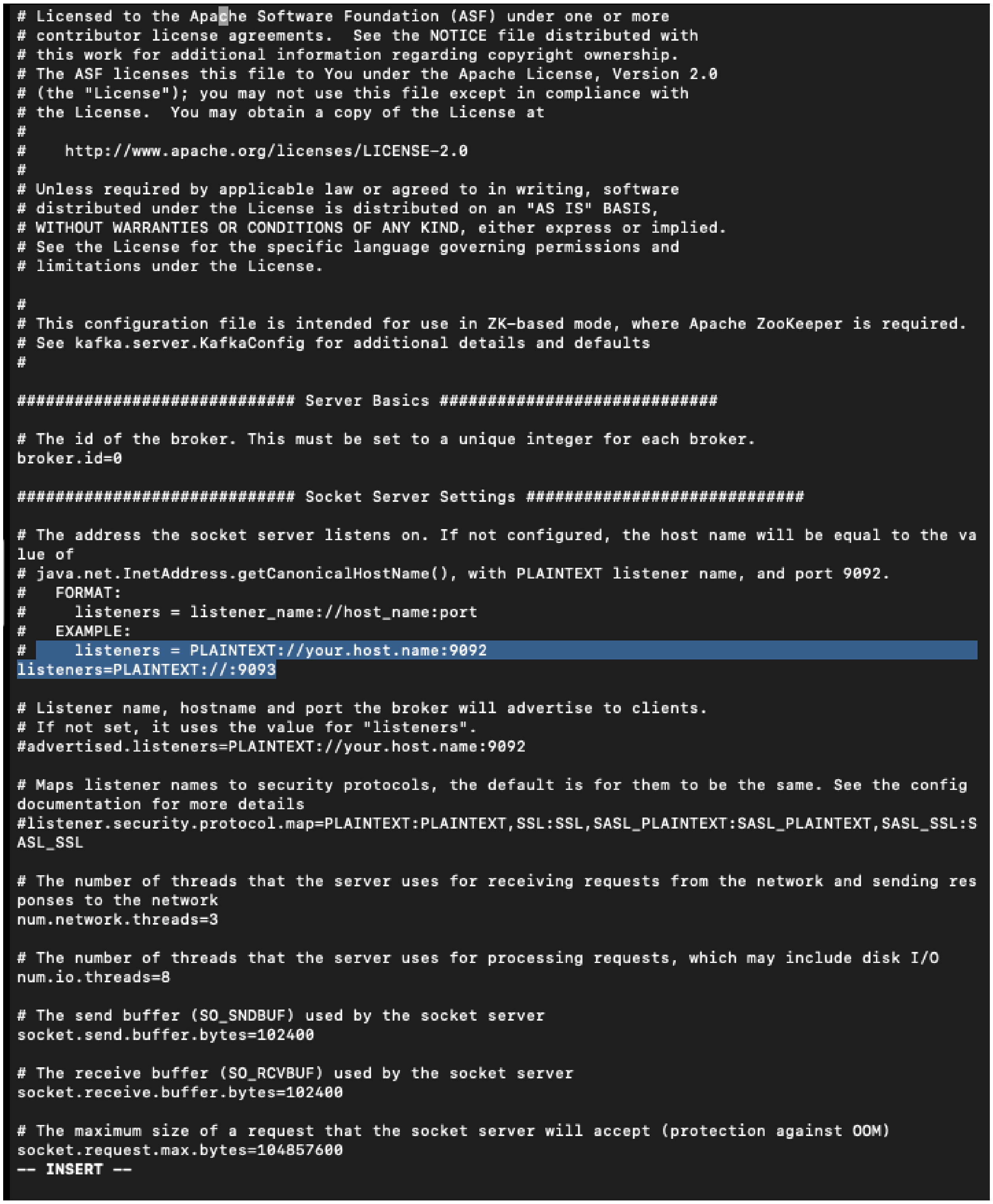
   `--bootstrap-server localhost:9092`: Connects to the Kafka broker.

   `--topic mytopic`: Specifies the topic to read messages from.

   `--from-beginning`: Tells Kafka to consume all messages from the beginning of the topic, not just new ones.

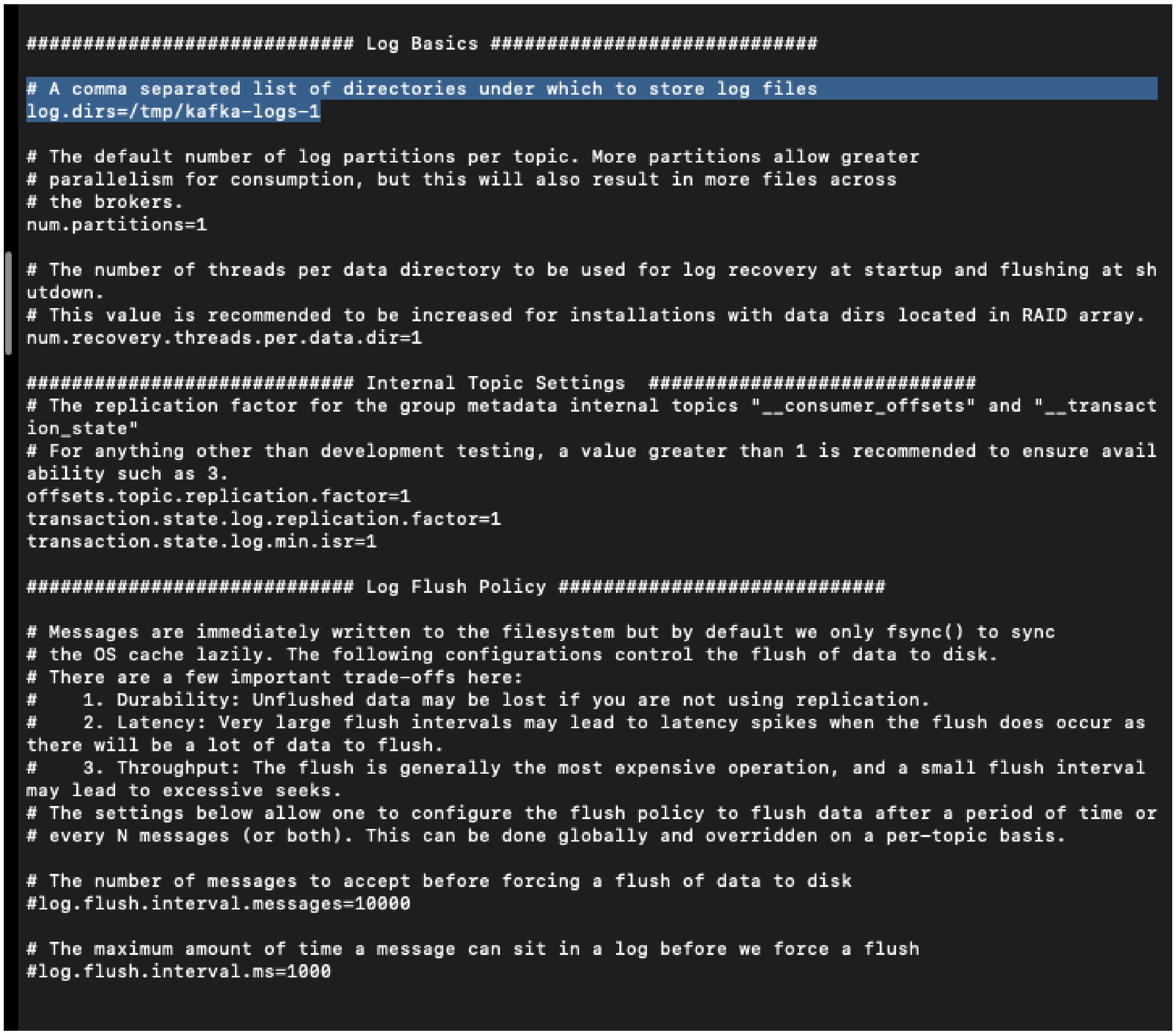

```
(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 % ./bin/kafka-console-consumer.sh --bootstrap-ser
ver localhost:9092 --topic mytopic --from-beginning

hello
world
new wvwnt
Hellooooo, I like baking cookies
```

7. **Configure second kafka broker: by copying the file:** *cp config/server.properties config/server-1.properties*



```
# Licensed to the Apache Software Foundation (ASF) under one or more
# contributor license agreements.  See the NOTICE file distributed with
# this work for additional information regarding copyright ownership.
# The ASF licenses this file to You under the Apache License, Version 2.0
# (the "License"); you may not use this file except in compliance with
# the License.  You may obtain a copy of the License at
#
#    http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

#
# This configuration file is intended for use in ZK-based mode, where Apache ZooKeeper is required.
# See kafka.server.KafkaConfig for additional details and defaults
#

############################# Server Basics #############################

# The id of the broker. This must be set to a unique integer for each broker.
broker.id=0

############################# Socket Server Settings #############################

# The address the socket server listens on. If not configured, the host name will be equal to the va
lue of
# java.net.InetAddress.getCanonicalHostName(), with PLAINTEXT listener name, and port 9092.
#   FORMAT:
#     listeners = listener_name://host_name:port
#   EXAMPLE:
#     listeners = PLAINTEXT://your.host.name:9092
listeners=PLAINTEXT://:9093

# Listener name, hostname and port the broker will advertise to clients.
# If not set, it uses the value for "listeners".
#advertised.listeners=PLAINTEXT://your.host.name:9092

# Maps listener names to security protocols, the default is for them to be the same. See the config
documentation for more details
#listener.security.protocol.map=PLAINTEXT:PLAINTEXT,SSL:SSL,SASL_PLAINTEXT:SASL_PLAINTEXT,SASL_SSL:S
ASL_SSL

# The number of threads that the server uses for receiving requests from the network and sending res
ponses to the network
num.network.threads=3

# The number of threads that the server uses for processing requests, which may include disk I/O
num.io.threads=8

# The send buffer (SO_SNDBUF) used by the socket server
socket.send.buffer.bytes=102400

# The receive buffer (SO_RCVBUF) used by the socket server
socket.receive.buffer.bytes=102400

# The maximum size of a request that the socket server will accept (protection against OOM)
socket.request.max.bytes=104857600
-- INSERT --
```

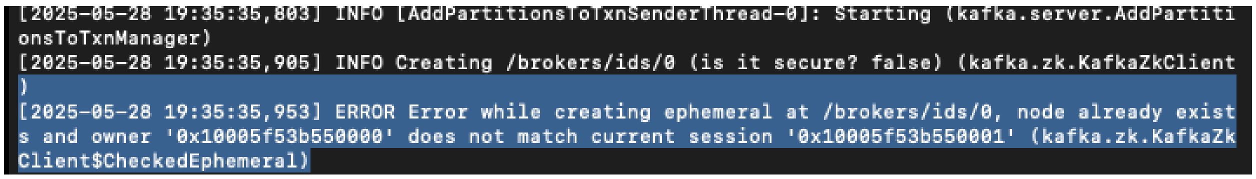
Change #listeners= PLAINTEXT://:9092 to PLAINTEXT:://:9093 as shown in the figure above



```
############################# Log Basics #############################

# A comma separated list of directories under which to store log files
log.dirs=/tmp/kafka-logs-1

# The default number of log partitions per topic. More partitions allow greater
# parallelism for consumption, but this will also result in more files across
# the brokers.
num.partitions=1

# The number of threads per data directory to be used for log recovery at startup and flushing at sh
utdown.
# This value is recommended to be increased for installations with data dirs located in RAID array.
num.recovery.threads.per.data.dir=1

############################# Internal Topic Settings  #############################
# The replication factor for the group metadata internal topics "__consumer_offsets" and "__transact
ion_state"
# For anything other than development testing, a value greater than 1 is recommended to ensure avail
ability such as 3.
offsets.topic.replication.factor=1
transaction.state.log.replication.factor=1
transaction.state.log.min.isr=1

############################# Log Flush Policy #############################

# Messages are immediately written to the filesystem but by default we only fsync() to sync
# the OS cache lazily. The following configurations control the flush of data to disk.
# There are a few important trade-offs here:
#    1. Durability: Unflushed data may be lost if you are not using replication.
#    2. Latency: Very large flush intervals may lead to latency spikes when the flush does occur as
there will be a lot of data to flush.
#    3. Throughput: The flush is generally the most expensive operation, and a small flush interval
may lead to excessive seeks.
# The settings below allow one to configure the flush policy to flush data after a period of time or
# every N messages (or both). This can be done globally and overridden on a per-topic basis.

# The number of messages to accept before forcing a flush of data to disk
#log.flush.interval.messages=10000

# The maximum amount of time a message can sit in a log before we force a flush
#log.flush.interval.ms=1000
```

Add-1 to log.dirs=/tmp/kafka-logs and change it to log.dirs=/tmp/kafka-logs-1 as mentioned above

- When you try to run this broker, it throws an error:
    - Because it requires unique broker id



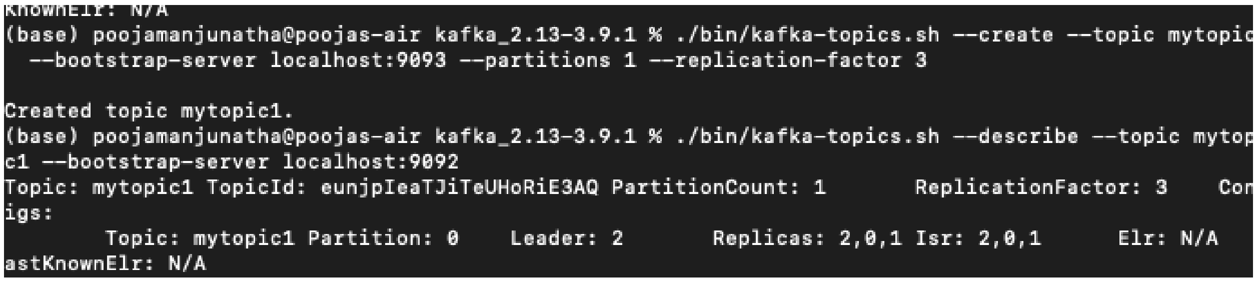
```
[2025-05-28 19:35:35,803] INFO [AddPartitionsToTxnSenderThread-0]: Starting (kafka.server.AddPartiti
onsToTxnManager)
[2025-05-28 19:35:35,905] INFO Creating /brokers/ids/0 (is it secure? false) (kafka.zk.KafkaZkClient
)
[2025-05-28 19:35:35,953] ERROR Error while creating ephemeral at /brokers/ids/0, node already exist
s and owner '0x10005f53b550000' does not match current session '0x10005f53b550001' (kafka.zk.KafkaZk
Client$CheckedEphemeral)
```

./bin/kafka-server-start.sh config/server-1.properties

- Create three brokers with the same idea, changing the port to 9093 and 9094.
- Changing broker id to 2 and 3 and logs -2,-3

8. **Created a new topic called mytopic1 with 3 replication factor:**

./bin/kafka-topics.sh --describe --topic mytopic1 --bootstrap-server localhost:9092



```
KnownElr: N/A
(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 % ./bin/kafka-topics.sh --create --topic mytopic
  --bootstrap-server localhost:9093 --partitions 1 --replication-factor 3

Created topic mytopic1.
(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 % ./bin/kafka-topics.sh --describe --topic mytop
c1 --bootstrap-server localhost:9092
Topic: mytopic1 TopicId: eunjpIeaTJiTeUHoRiE3AQ PartitionCount: 1       ReplicationFactor: 3    Con
igs:
        Topic: mytopic1 Partition: 0    Leader: 2       Replicas: 2,0,1 Isr: 2,0,1      Elr: N/A
astKnownElr: N/A
```

Field	Description
Topic	Name of the Kafka topic: `mytopic1`
TopicId	Unique ID assigned to the topic internally
PartitionCount	Number of partitions: `1`
ReplicationFactor	Number of replicas for each partition: `3`
Leader	Broker currently acting as the leader for partition 0: `Broker 2`
Replicas	List of broker IDs that hold replicas: `[2, 0, 1]`
Isr (In-Sync Replicas)	Replicas that are fully caught up and synchronized: `[2, 0, 1]`
Elr (Eligible Leader Replicas)	Deprecated (shows `N/A`)
LastKnownElr	Deprecated (shows `N/A`)

- **Leader: T**his is the broker responsible for handling all reads and writes for this partition.

- **Replicas:** These are brokers storing a copy of the partition. Even if not leaders, they ensure high availability.

- **ISR (In-Sync Replicas)**: Replicas that are up-to-date with the leader. All 3 brokers are in sync — this is ideal.

- This implies you have 3 Kafka brokers running with `broker.id`s 0, 1, and 2 — necessary to support replication factor 3.

9. Custom Kafka Logging Setup and Console Producer Test

- This command sets the `KAFKA_OPTS` environment variable to use a custom Log4j configuration file located at `config/log4j.properties`. It ensures that all Kafka logs follow the structure and log levels defined in this file, which helps in debugging and log management

```
export KAFKA_OPTS="-Dlog4j.configuration=file:config/log4j.properties"
```

- Produce messages to kafka

```
./bin/kafka-console-producer.sh --topic mytopic1 --bootstrap-server localhost:9093
```

  - This command launches the Kafka Console Producer that connects to the Kafka broker running on `localhost:9093` and sends messages to the topic named `mytopic1`.
  - After running the command, you can type messages in the terminal and press Enter to send each message to the Kafka topic.
  - The producer keeps running until you stop it (using `Ctrl+C`).

- These steps confirm that:
  - Your Kafka broker is running and reachable.
  - The topic `mytopic1` is properly created.
  - The custom logging configuration is active and logs Kafka events as specified.

Kafka requires each broker in the cluster to have a **unique, persistent identity**. This identity is defined by the broker's `broker.id`, which is used to manage partitions, replication, and coordination within the Kafka cluster.

In Kubernetes, if you use a regular **Deployment**, pods are treated as interchangeable and may get new names or IPs when restarted. This can break Kafka's expectations about broker identity and lead to issues with data consistency and cluster stability.

To solve this, **StatefulSets** are used. A StatefulSet ensures:

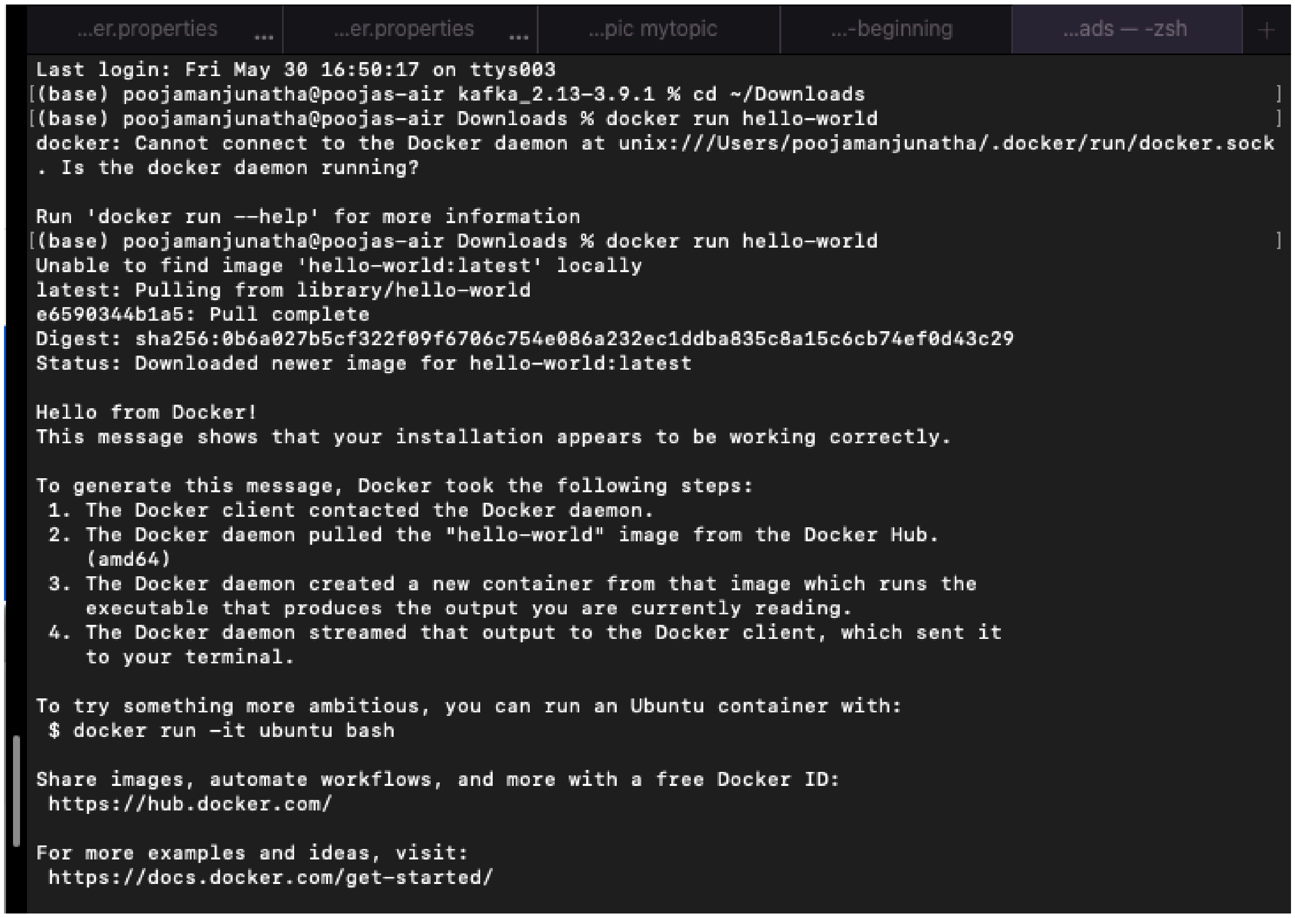
- Each Kafka broker pod has a **stable, unique name** (like `kafka-0`, `kafka-1`, etc.)
- Each broker gets its own **dedicated persistent volume**, which is not shared and is preserved across restarts
- The **broker ID can be mapped directly** from the pod name (e.g., `broker.id=0` for `kafka-0`).

This persistent identity and storage are **critical** for Kafka's correct operation and for maintaining data integrity.

9. Download docker

- Move to the folder it is downloaded in and run the command

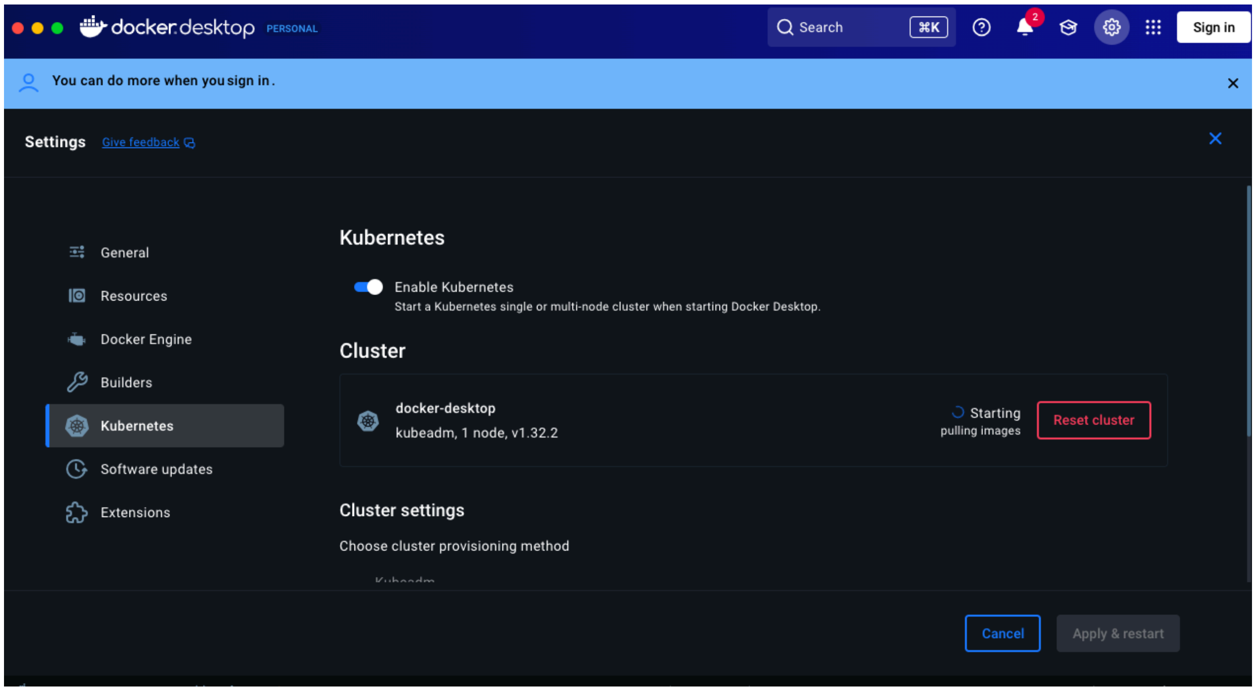
docker run hello-world



```
Last login: Fri May 30 16:50:17 on ttys003
[(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 % cd ~/Downloads                                  ]
[(base) poojamanjunatha@poojas-air Downloads % docker run hello-world                                 ]
docker: Cannot connect to the Docker daemon at unix:///Users/poojamanjunatha/.docker/run/docker.sock
. Is the docker daemon running?

Run 'docker run --help' for more information
[(base) poojamanjunatha@poojas-air Downloads % docker run hello-world                                 ]
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
e6590344b1a5: Pull complete
Digest: sha256:0b6a027b5cf322f09f6706c754e086a232ec1ddba835c8a15c6cb74ef0d43c29
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

10. Install Kubernetes

- Go to settings in docker and enable kubernetes



Kubernetes

Enable Kubernetes
Start a Kubernetes single or multi-node cluster when starting Docker Desktop.

Cluster

docker-desktop
kubeadm, 1 node, v1.32.2
Starting
pulling images
Reset cluster

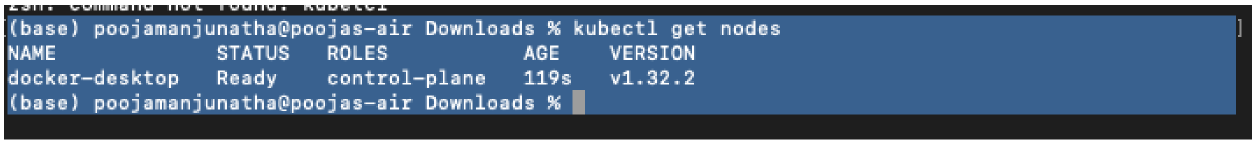
Cluster settings

Choose cluster provisioning method

- Run this command in kubernetes to check if it is working

kubectl get nodes

The output should contain docker-desktop as shown in the image below



```
(base) poojamanjunatha@poojas-air Downloads % kubectl get nodes
NAME             STATUS   ROLES           AGE    VERSION
docker-desktop   Ready    control-plane   119s   v1.32.2
(base) poojamanjunatha@poojas-air Downloads %
```

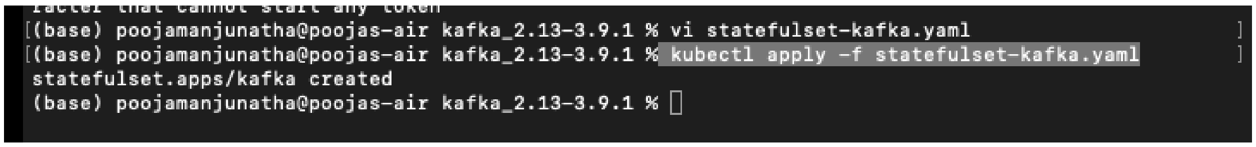
- Open a new filecalled statefulset-kafka.yaml and add the content:
  - The command to create a new file in the terminal-

    vi statefulset-kafka.yaml

The content of the file will be:

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: kafka
spec:
  selector:
    matchLabels:
      app: kafka
  serviceName: "kafka"
  replicas: 1
  template:
    metadata:
      labels:
        app: kafka
    spec:
      containers:
        - name: kafka
          image: debezium/kafka
          ports:
            - containerPort: 9092
              name: kafka
          env:
            - name: BROKER_ID
              value: "0"
            - name: ZOOKEEPER_CONNECT
              value: "host.docker.internal:2181"
```

- Run the command *kubectl apply -f statefulset-kafka.yaml*
  This applies kubernetes to the cluster

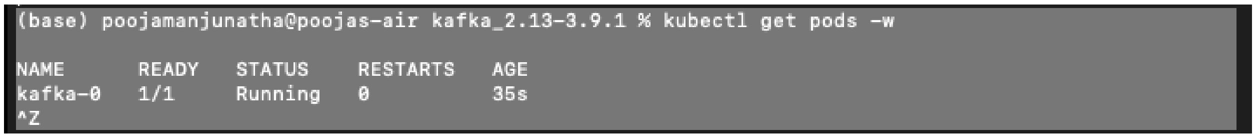


```
 factor that cannot start any token
[(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 % vi statefulset-kafka.yaml                  ]
[(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 % kubectl apply -f statefulset-kafka.yaml    ]
 statefulset.apps/kafka created
 (base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 %
```

- Run the command:`kubectl get pods` lists all the current pods in your Kubernetes cluster (in the current namespace).

- The `-w` (or `--watch`) flag tells Kubernetes to **continuously watch and stream updates** about the pods as they happen in real-time.

kafka_2.13-3.9.1 % kubectl get pods -w

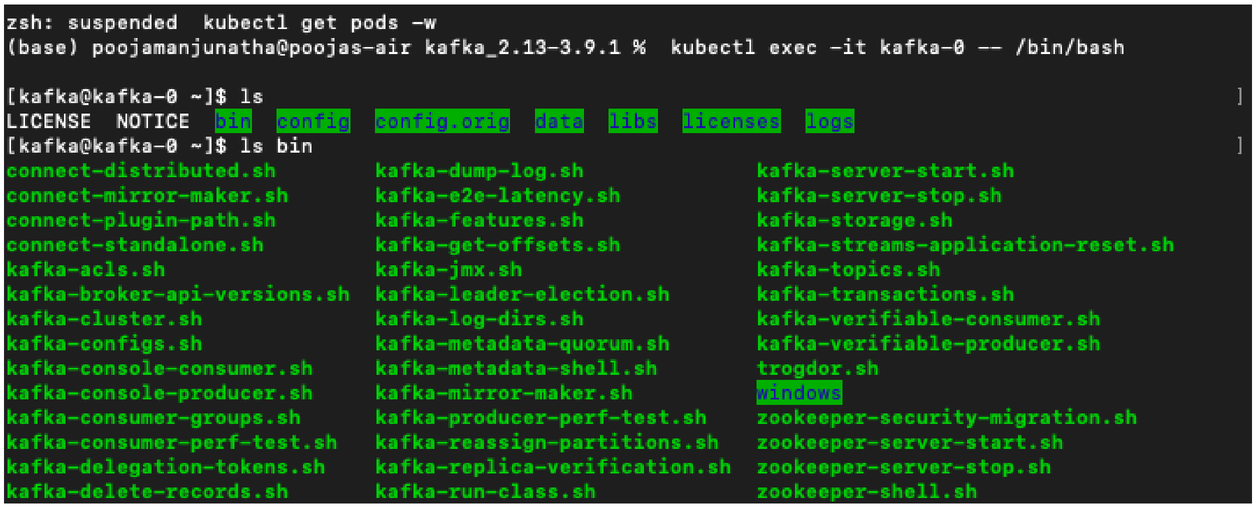


```
(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 % kubectl get pods -w

NAME       READY    STATUS     RESTARTS    AGE
kafka-0    1/1      Running    0           35s
^Z
```

- The command below is used to open an interactive Bash shell inside the running Kafka pod named `kafka-0` in your Kubernetes cluster. It allows you to directly access the container's command line interface, just like logging into a remote server.

This is helpful for performing tasks such as inspecting files, running Kafka commands, checking environment variables, or debugging issues within the pod. The `-it` flag makes the session interactive and user-friendly by enabling input and displaying output in a terminal-like environment.

kafka_2.13-3.9.1 % kubectl exec -it kafka-0 -- /bin/bash

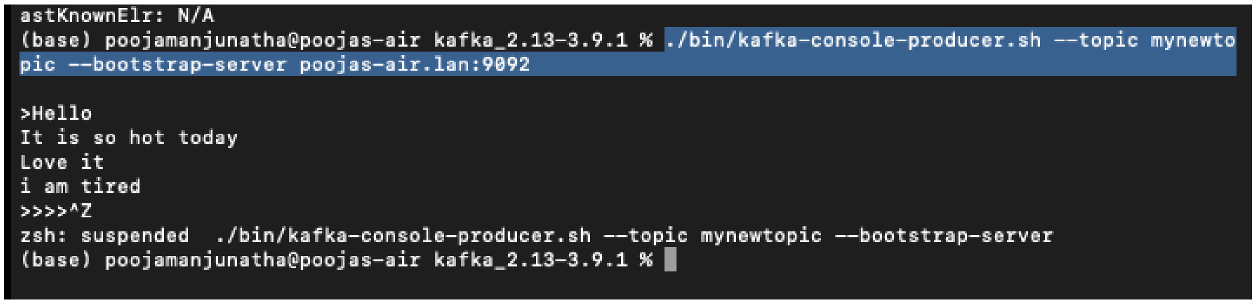


```
zsh: suspended  kubectl get pods -w
(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 %  kubectl exec -it kafka-0 -- /bin/bash

[kafka@kafka-0 ~]$ ls
LICENSE  NOTICE  bin  config  config.orig  data  libs  licenses  logs
[kafka@kafka-0 ~]$ ls bin
connect-distributed.sh         kafka-dump-log.sh              kafka-server-start.sh
connect-mirror-maker.sh        kafka-e2e-latency.sh           kafka-server-stop.sh
connect-plugin-path.sh         kafka-features.sh              kafka-storage.sh
connect-standalone.sh          kafka-get-offsets.sh           kafka-streams-application-reset.sh
kafka-acls.sh                  kafka-jmx.sh                   kafka-topics.sh
kafka-broker-api-versions.sh   kafka-leader-election.sh       kafka-transactions.sh
kafka-cluster.sh               kafka-log-dirs.sh              kafka-verifiable-consumer.sh
kafka-configs.sh               kafka-metadata-quorum.sh       kafka-verifiable-producer.sh
kafka-console-consumer.sh      kafka-metadata-shell.sh        trogdor.sh
kafka-console-producer.sh      kafka-mirror-maker.sh          windows
kafka-consumer-groups.sh       kafka-producer-perf-test.sh    zookeeper-security-migration.sh
kafka-consumer-perf-test.sh    kafka-reassign-partitions.sh   zookeeper-server-start.sh
kafka-delegation-tokens.sh     kafka-replica-verification.sh  zookeeper-server-stop.sh
kafka-delete-records.sh        kafka-run-class.sh             zookeeper-shell.sh
```

- Create a new topic inside the container, called "mynewtopic" and then describe it using the command

./bin/kafka-topics.sh --describe --topic mynewtopic --bootstrap-server poojas-air.lan:9092

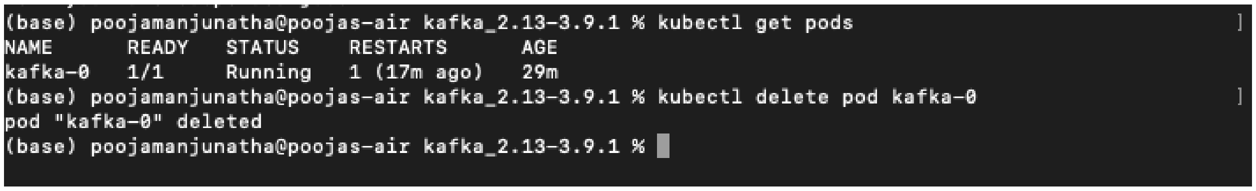
- Produce messages in this topic:



```
astKnownElr: N/A
(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 % ./bin/kafka-console-producer.sh --topic mynewto
pic --bootstrap-server poojas-air.lan:9092

>Hello
It is so hot today
Love it
i am tired
>>>>^Z
zsh: suspended  ./bin/kafka-console-producer.sh --topic mynewtopic --bootstrap-server
(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 %
```

Send and produce messages

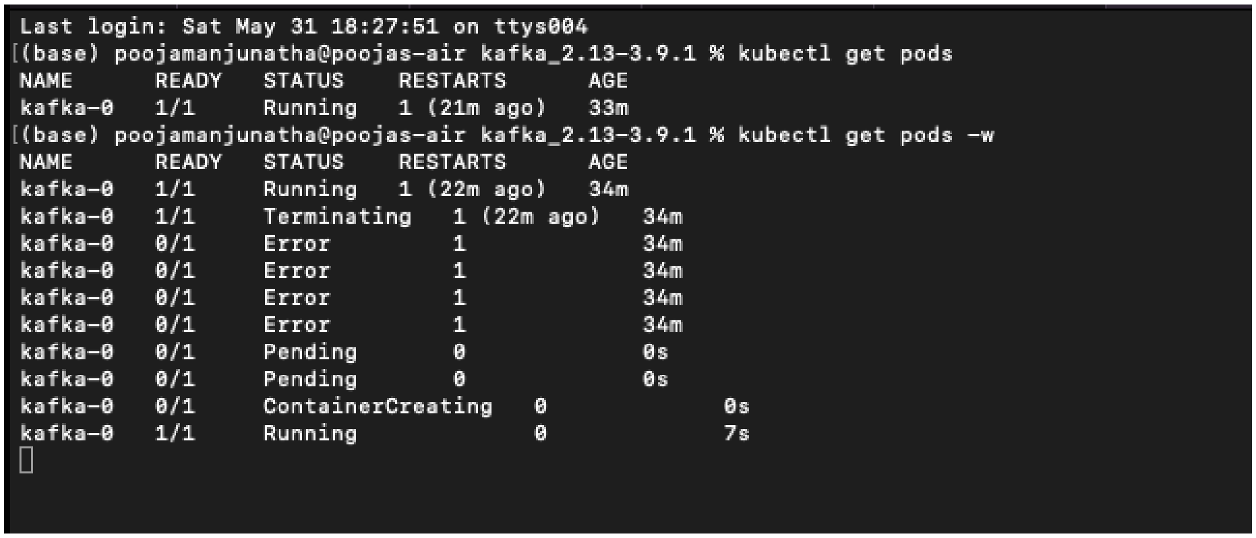
- To check the status of the pod- delete the pod
  kubectl delete pod kafka-0

- 

```
(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 % kubectl get pods
NAME      READY   STATUS    RESTARTS      AGE
kafka-0   1/1     Running   1 (17m ago)   29m
(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 % kubectl delete pod kafka-0
pod "kafka-0" deleted
(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 %
```

- Run this command in another tab and see the status

kubectl get pods -w



```
Last login: Sat May 31 18:27:51 on ttys004
[(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 % kubectl get pods
NAME      READY   STATUS    RESTARTS      AGE
kafka-0   1/1     Running   1 (21m ago)   33m
[(base) poojamanjunatha@poojas-air kafka_2.13-3.9.1 % kubectl get pods -w
NAME      READY   STATUS    RESTARTS      AGE
kafka-0   1/1     Running   1 (22m ago)   34m
kafka-0   1/1     Terminating   1 (22m ago)   34m
kafka-0   0/1     Error         1             34m
kafka-0   0/1     Error         1             34m
kafka-0   0/1     Error         1             34m
kafka-0   0/1     Error         1             34m
kafka-0   0/1     Pending       0             0s
kafka-0   0/1     Pending       0             0s
kafka-0   0/1     ContainerCreating   0             0s
kafka-0   1/1     Running             0             7s
```

Kubernetes, when a pod (like your Kafka pod `kafka-0`) is terminated (intentionally or due to an error), the **controller managing the pod**, typically a **StatefulSet** (for Kafka) or **Deployment**, will automatically recreate it to maintain the desired state.

However, this automatic recreation **does not retain data** unless:

1. **Persistent Volumes (PVs)** are properly configured and mounted via **Persistent Volume Claims (PVCs)**.

2. Kafka is configured to write logs and data to those volumes.

If Kafka is **not** using persistent storage, then on pod restart, it will lose all topic data (e.g., messages, logs, consumer offsets), and it will start fresh — which explains why your consumer might not receive anything.

To summarize:

- Yes, the pod is recreated automatically.

- But unless persistent storage is used, **data is lost** on each termination/restart.
- You can check if a PVC is mounted with: