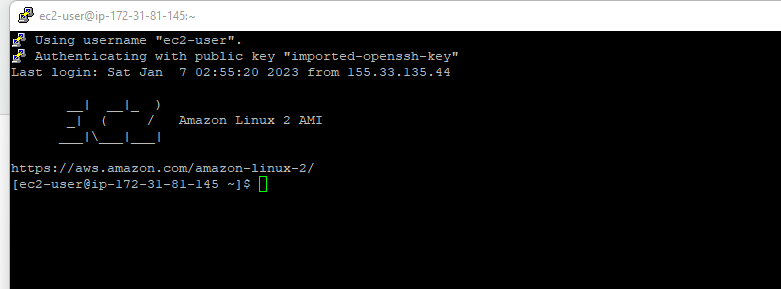
Stock Market Real-Time Data Analysis Using Kafka

End-To-End Data Engineering Project

* Kafka -Distributed Event Store and Stream Processing platform
* Producers – Data generation platforms – Logs, sensors, logistics, EMR, Transactions, Google Maps
* Brokers – Single member server of kafka cluster (node or server basically a single machine is called broker), used for receiving and storing messages
* Consumer – Who consumes the data and uses for any activity, pulls messages form topics (0.. N)
* Zookeeper – Resource Manager – Distributed System so making sure every resource work in alignment and are synchronous, stores the ACLs and secrets keys and access, store configurations
* Topics – Logical representation of messages inside Kafka (like Shipping Orders, return orders, Payments etc.) -- > Categorical Messages
* Brokers can have partitions (0 .. N) based on requirements, ordering guarantee
* Each log file is a separate partition. Data -> Topic -> Partitioned using log file
* As messages keep on coming it gets appended to The Log file
* Broker replication for fault tolerance like hdfs replication
* Producers -> Write messages in any language Python, JS, Java etc
* Timeline

  Description automatically generated
* Login in EC2 instance using AWS console
* 
* "C:\ExD\projects\Stock-Market-RT-Analysis\command\_kafka.txt" – Run the kafka commands
* Text

  Description automatically generated
* Once all the java and kafka are installed, start the Zookeeper in the running EC2 instance
* Text

  Description automatically generated with medium confidence
* Now open another EC2 instance and connect to kafka server as we need to let Zookeeper running in one instance
* Graphical user interface, text

  Description automatically generated
* Start the kafka server in the another windowText

  Description automatically generated
* Stop both zookeeper and kafka producer as we cannot connect to the servers running in private DNS from local machine.
* A picture containing graphical user interface

  Description automatically generated
* Graphical user interface, text

  Description automatically generated
* Update the ip in config with public ip
* Text

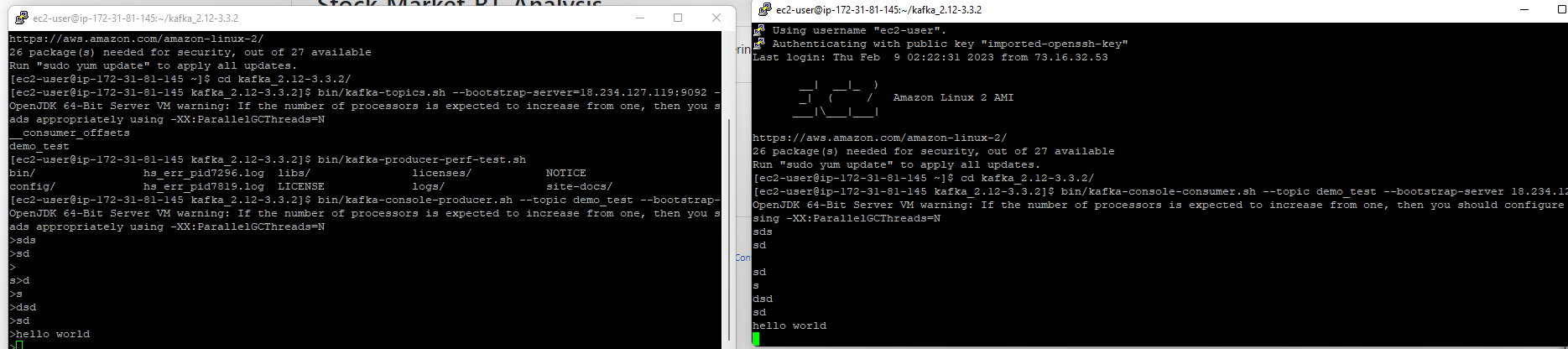
  Description automatically generated
* Save Ctrl+X -> Y -> Enter
* Now run both the zookeeper and kafka server again using the commands
* Text

  Description automatically generated with medium confidence
* Now allow local machine access to the EC2 instance. Add inbound all traffic rule in SecGroup of EC2 instance as below:
* Graphical user interface, application, Word

  Description automatically generated
* Open another EC2 instance for creating the topics and creating producers and consumers
* A computer screen capture

  Description automatically generated with medium confidence
* On same window start the producer
* A picture containing graphical user interface

  Description automatically generated
* Open another EC2 instance and start the consumer as below:
* Text

  Description automatically generated
* And now go to Producer and start publishing messages and we can see that consumer is receiving the messages as well
* 
* Load Jupyter in C drive as below command
* Text

  Description automatically generated
* Graphical user interface, text, application, email

  Description automatically generated
* If the kafka cluster broke and gives “Broker not available” then restart the zookeeper, KafkaServer, Producer and Consumer and then stream the data form Producer
* Now, we can push data from local machine to S3 bucket that we created. For that we need to install awscli (after creating IAM user and configuring the user in local machine) \*\* already done by me in past
* Text

  Description automatically generated
* Check the KafkaProducer and KafkaConsumer python file for generating and consuming data scripts
* Now, we are going to crawl through the data using AWS Glue as below create crawler, place “/” at the end in s3 location
* Graphical user interface, text, application, email

  Description automatically generated
* Below is the Glue Crawler config.
* Graphical user interface, application, email

  Description automatically generated
* Open the Athena.. Go to Settings and add the s3 location to sore the query data
* Graphical user interface, application, Teams

  Description automatically generated
* Graphical user interface, application

  Description automatically generated