Group Project - 1

Group Number: 2

Brian Honea (responsible for part B ONLY SHOWN in this version)

Harshita Kala

Dorsey Kirpatrick

Basil Muhammad

<u>Project Part B: Flume Ingestion</u>

Screen shot of python code: I used Twelve Data API to pull data for AAPL, IBM, GOOGL, MSFT, and TSLA. I took time to figure out how to get the API to communicate appropriately and finally got it to work. I understand this submission is late.

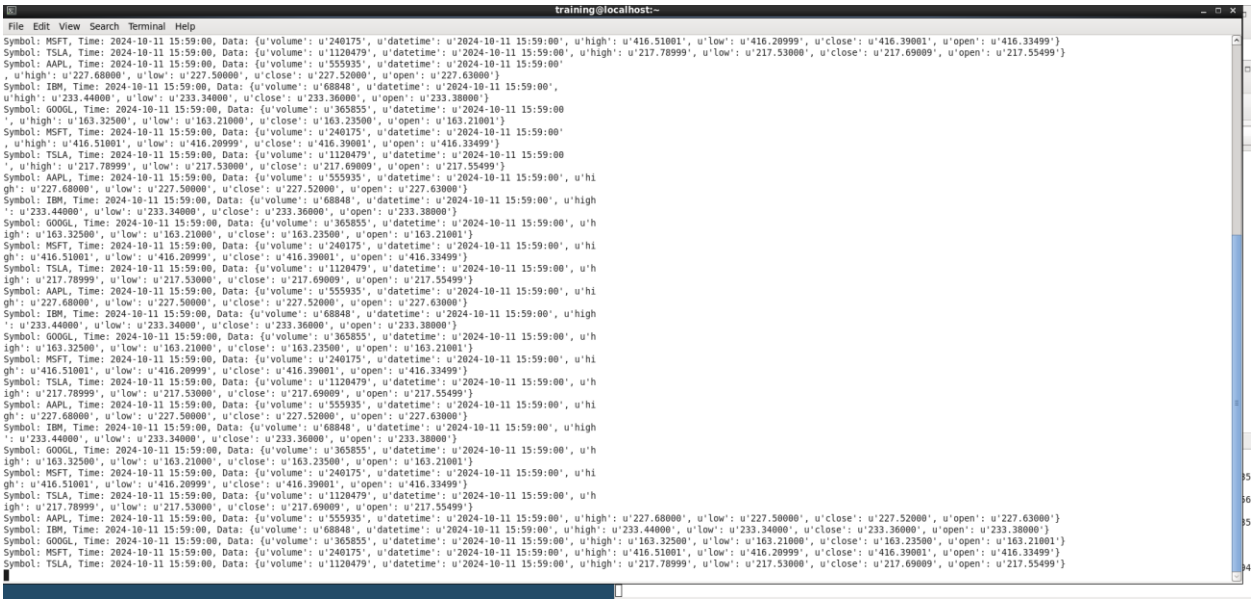
cloudera-training-capspark-student-rev_cdh5.4.3a - VMware Workstation 17 Player (Non-commercial use only)

Player

```python
import requests
import json
import socket
import time
from datetime import datetime

# Replace with your Twelve Data API key
API_KEY = "3953f49cbf664ed386a217&ac1c91d78"

# Function to fetch stock data from Twelve Data API
def fetch_stock_data(symbol):
    url = "https://api.twelvedata.com/time_series?symbol={}&interval=1min&apikey={}".format(symbol, API_KEY)
    response = requests.get(url)

    # Check if the response is successful
    if response.status_code == 200:
        try:
            data = response.json()
            if 'values' in data:
                return data
            else:
                print("Error in response:", data)
                return {}
        except ValueError as e:
            print("JSON Decode Error:", e)
            return {}
    else:
        print("Error fetching data for {}: Status Code: {}, Response Content: {}".format(symbol, response.status_code, response.text))
        return {}

# Define the stock symbols to monitor
symbols = ["AAPL", "IBM", "GOOGL", "MSFT", "TSLA"]

# Define the localhost server settings
HOST = 'localhost'
PORT = 12345  # You can choose any available port

# Function to send data to localhost server
def send_data_to_localhost(data):
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.connect((HOST, PORT))
    s.sendall(data + "\n")  # Send data with a newline character
    s.close()

# Function to determine if the market is open (9:30 am to 4:00 pm ET)
def is_market_open():
    return True
    #now = datetime.now()
    #market_open = now.replace(hour=9, minute=30, second=0, microsecond=0)
    #market_close = now.replace(hour=16, minute=0, second=0, microsecond=0)

    # Checking if it's a weekday (Mon-Fri)
    #if now.weekday() < 5:
        #return market_open <= now <= market_close
    #return False

# Main loop to fetch stock data and send to localhost
```

Python output in console:

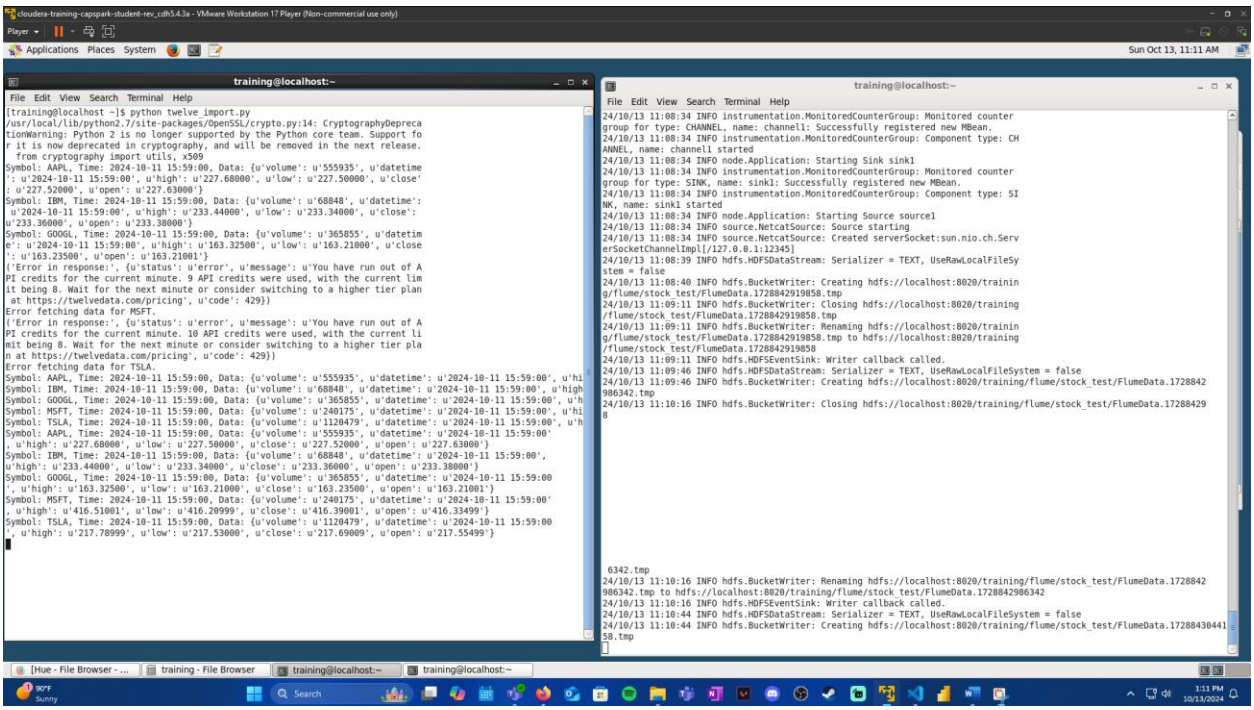
As you can see the python file is running as it should in the VM server. It is outputting the data gathered from the API for AAPL, IBM, GOOGL, MSFT, and TSLA.



```
Symbol: MSFT, Time: 2024-10-11 15:59:00, Data: {u'volume': u'240175', u'datetime': u'2024-10-11 15:59:00', u'high': u'416.51001', u'low': u'416.20999', u'close': u'416.39001', u'open': u'416.33499'}
Symbol: TSLA, Time: 2024-10-11 15:59:00, Data: {u'volume': u'1120479', u'datetime': u'2024-10-11 15:59:00', u'high': u'217.78999', u'low': u'217.53000', u'close': u'217.69009', u'open': u'217.55499'}
Symbol: AAPL, Time: 2024-10-11 15:59:00, Data: {u'volume': u'555935', u'datetime': u'2024-10-11 15:59:00'
, u'high': u'227.68000', u'low': u'227.50000', u'close': u'227.52000', u'open': u'227.63000'}
Symbol: IBM, Time: 2024-10-11 15:59:00, Data: {u'volume': u'68848', u'datetime': u'2024-10-11 15:59:00',
u'high': u'233.44000', u'low': u'233.34000', u'close': u'233.36000', u'open': u'233.38000'}
Symbol: GOOGL, Time: 2024-10-11 15:59:00, Data: {u'volume': u'365855', u'datetime': u'2024-10-11 15:59:00
', u'high': u'163.32500', u'low': u'163.21000', u'close': u'163.23500', u'open': u'163.21001'}
Symbol: MSFT, Time: 2024-10-11 15:59:00, Data: {u'volume': u'240175', u'datetime': u'2024-10-11 15:59:00'
, u'high': u'416.51001', u'low': u'416.20999', u'close': u'416.39001', u'open': u'416.33499'}
Symbol: TSLA, Time: 2024-10-11 15:59:00, Data: {u'volume': u'1120479', u'datetime': u'2024-10-11 15:59:00
', u'high': u'217.78999', u'low': u'217.53000', u'close': u'217.69009', u'open': u'217.55499'}
Symbol: AAPL, Time: 2024-10-11 15:59:00, Data: {u'volume': u'555935', u'datetime': u'2024-10-11 15:59:00', u'hi
gh': u'227.68000', u'low': u'227.50000', u'close': u'227.52000', u'open': u'227.63000'}
Symbol: IBM, Time: 2024-10-11 15:59:00, Data: {u'volume': u'68848', u'datetime': u'2024-10-11 15:59:00', u'high
': u'233.44000', u'low': u'233.34000', u'close': u'233.36000', u'open': u'233.38000'}
Symbol: GOOGL, Time: 2024-10-11 15:59:00, Data: {u'volume': u'365855', u'datetime': u'2024-10-11 15:59:00', u'h
igh': u'163.32500', u'low': u'163.21000', u'close': u'163.23500', u'open': u'163.21001'}
Symbol: MSFT, Time: 2024-10-11 15:59:00, Data: {u'volume': u'240175', u'datetime': u'2024-10-11 15:59:00', u'hi
gh': u'416.51001', u'low': u'416.20999', u'close': u'416.39001', u'open': u'416.33499'}
Symbol: TSLA, Time: 2024-10-11 15:59:00, Data: {u'volume': u'1120479', u'datetime': u'2024-10-11 15:59:00', u'h
igh': u'217.78999', u'low': u'217.53000', u'close': u'217.69009', u'open': u'217.55499'}
Symbol: AAPL, Time: 2024-10-11 15:59:00, Data: {u'volume': u'555935', u'datetime': u'2024-10-11 15:59:00', u'hi
gh': u'227.68000', u'low': u'227.50000', u'close': u'227.52000', u'open': u'227.63000'}
Symbol: IBM, Time: 2024-10-11 15:59:00, Data: {u'volume': u'68848', u'datetime': u'2024-10-11 15:59:00', u'high
': u'233.44000', u'low': u'233.34000', u'close': u'233.36000', u'open': u'233.38000'}
Symbol: GOOGL, Time: 2024-10-11 15:59:00, Data: {u'volume': u'365855', u'datetime': u'2024-10-11 15:59:00', u'h
igh': u'163.32500', u'low': u'163.21000', u'close': u'163.23500', u'open': u'163.21001'}
Symbol: MSFT, Time: 2024-10-11 15:59:00, Data: {u'volume': u'240175', u'datetime': u'2024-10-11 15:59:00', u'hi
gh': u'416.51001', u'low': u'416.20999', u'close': u'416.39001', u'open': u'416.33499'}
Symbol: TSLA, Time: 2024-10-11 15:59:00, Data: {u'volume': u'1120479', u'datetime': u'2024-10-11 15:59:00', u'h
igh': u'217.78999', u'low': u'217.53000', u'close': u'217.69009', u'open': u'217.55499'}
Symbol: AAPL, Time: 2024-10-11 15:59:00, Data: {u'volume': u'555935', u'datetime': u'2024-10-11 15:59:00', u'hi
gh': u'227.68000', u'low': u'227.50000', u'close': u'227.52000', u'open': u'227.63000'}
Symbol: IBM, Time: 2024-10-11 15:59:00, Data: {u'volume': u'68848', u'datetime': u'2024-10-11 15:59:00', u'high
': u'233.44000', u'low': u'233.34000', u'close': u'233.36000', u'open': u'233.38000'}
Symbol: GOOGL, Time: 2024-10-11 15:59:00, Data: {u'volume': u'365855', u'datetime': u'2024-10-11 15:59:00', u'h
igh': u'163.32500', u'low': u'163.21000', u'close': u'163.23500', u'open': u'163.21001'}
Symbol: MSFT, Time: 2024-10-11 15:59:00, Data: {u'volume': u'240175', u'datetime': u'2024-10-11 15:59:00', u'hi
gh': u'416.51001', u'low': u'416.20999', u'close': u'416.39001', u'open': u'416.33499'}
Symbol: TSLA, Time: 2024-10-11 15:59:00, Data: {u'volume': u'1120479', u'datetime': u'2024-10-11 15:59:00', u'h
igh': u'217.78999', u'low': u'217.53000', u'close': u'217.69009', u'open': u'217.55499'}
Symbol: AAPL, Time: 2024-10-11 15:59:00, Data: {u'volume': u'555935', u'datetime': u'2024-10-11 15:59:00', u'high': u'227.68000', u'low': u'227.50000', u'close': u'227.52000', u'open': u'227.63000'}
Symbol: IBM, Time: 2024-10-11 15:59:00, Data: {u'volume': u'68848', u'datetime': u'2024-10-11 15:59:00', u'high': u'233.44000', u'low': u'233.34000', u'close': u'233.36000', u'open': u'233.38000'}
Symbol: GOOGL, Time: 2024-10-11 15:59:00, Data: {u'volume': u'365855', u'datetime': u'2024-10-11 15:59:00', u'high': u'163.32500', u'low': u'163.21000', u'close': u'163.23500', u'open': u'163.21001'}
Symbol: MSFT, Time: 2024-10-11 15:59:00, Data: {u'volume': u'240175', u'datetime': u'2024-10-11 15:59:00', u'high': u'416.51001', u'low': u'416.20999', u'close': u'416.39001', u'open': u'416.33499'}
Symbol: TSLA, Time: 2024-10-11 15:59:00, Data: {u'volume': u'1120479', u'datetime': u'2024-10-11 15:59:00', u'high': u'217.78999', u'low': u'217.53000', u'close': u'217.69009', u'open': u'217.55499'}
```

Flume and Python code communicating:

Updated this code, now the API correctly pulls data and creates tmp files in HDFS. With this API I am allowed 800 API calls per day, so I will run this week to get a sufficient amount of data for the future project.



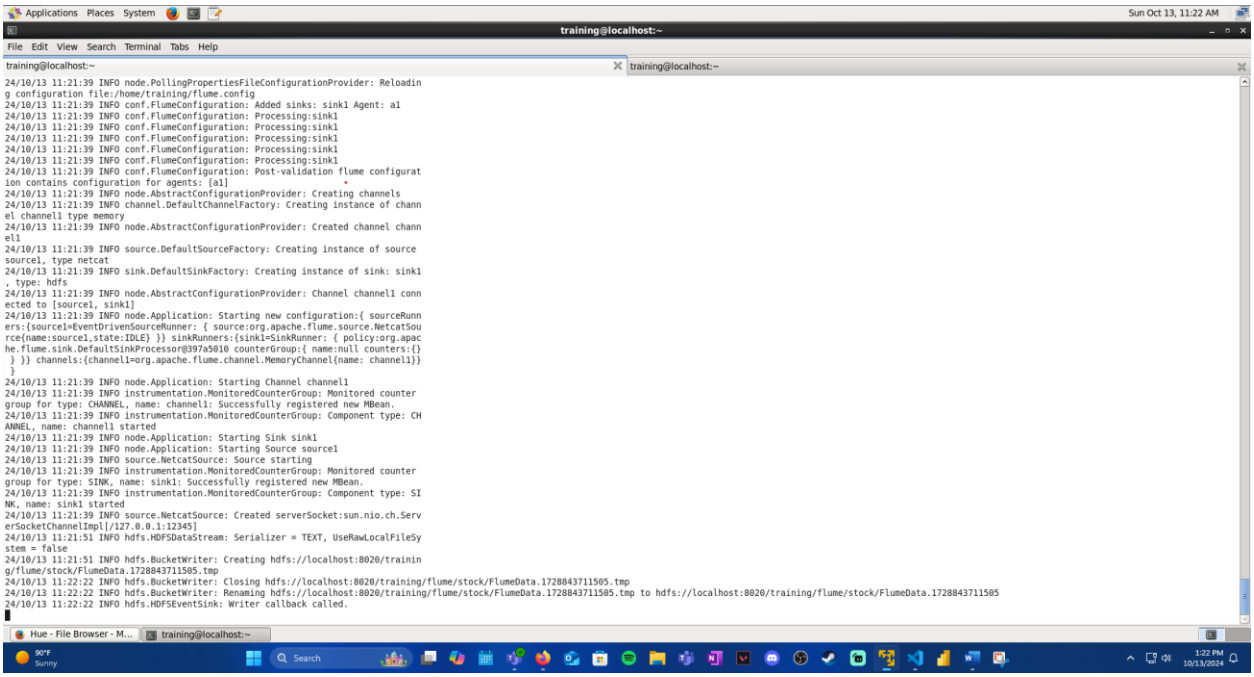
```
[training@localhost ~]$ python twelve_import.py
/usr/local/lib/python2.7/site-packages/OpenSSL/crypto.py:14: CryptographyDepreca
tionWarning: Python 2 is no longer supported by the Python core team. Support fo
r it is now deprecated in cryptography, and will be removed in the next release.
  from cryptography import utils, x509
Symbol: AAPL, Time: 2024-10-11 15:59:00, Data: {u'volume': u'555935', u'datetime
': u'2024-10-11 15:59:00', u'high': u'227.68000', u'low': u'227.50000', u'close'
: u'227.52000', u'open': u'227.63000'}
Symbol: IBM, Time: 2024-10-11 15:59:00, Data: {u'volume': u'68848', u'datetime':
 u'2024-10-11 15:59:00', u'high': u'233.44000', u'low': u'233.34000', u'close':
u'233.36000', u'open': u'233.38000'}
Symbol: GOOGL, Time: 2024-10-11 15:59:00, Data: {u'volume': u'365855', u'datetim
e': u'2024-10-11 15:59:00', u'high': u'163.32500', u'low': u'163.21000', u'close
': u'163.23500', u'open': u'163.21001'}
('Error in response:', {u'status': u'error', u'message': u'You have run out of A
PI credits for the current minute. 9 API credits were used, with the current lim
it being 8. Wait for the next minute or consider switching to a higher tier plan
 at https://twelvedata.com/pricing', u'code': 429})
Error fetching data for MSFT.
('Error in response:', {u'status': u'error', u'message': u'You have run out of A
PI credits for the current minute. 10 API credits were used, with the current li
mit being 8. Wait for the next minute or consider switching to a higher tier pla
n at https://twelvedata.com/pricing', u'code': 429})
Error fetching data for TSLA.
Symbol: AAPL, Time: 2024-10-11 15:59:00, Data: {u'volume': u'555935', u'datetime': u'2024-10-11 15:59:00', u'hi
Symbol: IBM, Time: 2024-10-11 15:59:00, Data: {u'volume': u'68848', u'datetime': u'2024-10-11 15:59:00', u'high
Symbol: GOOGL, Time: 2024-10-11 15:59:00, Data: {u'volume': u'365855', u'datetime': u'2024-10-11 15:59:00', u'h
Symbol: MSFT, Time: 2024-10-11 15:59:00, Data: {u'volume': u'240175', u'datetime': u'2024-10-11 15:59:00', u'hi
Symbol: TSLA, Time: 2024-10-11 15:59:00, Data: {u'volume': u'1120479', u'datetime': u'2024-10-11 15:59:00', u'h
Symbol: AAPL, Time: 2024-10-11 15:59:00, Data: {u'volume': u'555935', u'datetime': u'2024-10-11 15:59:00'
, u'high': u'227.68000', u'low': u'227.50000', u'close': u'227.52000', u'open': u'227.63000'}
Symbol: IBM, Time: 2024-10-11 15:59:00, Data: {u'volume': u'68848', u'datetime': u'2024-10-11 15:59:00',
u'high': u'233.44000', u'low': u'233.34000', u'close': u'233.36000', u'open': u'233.38000'}
Symbol: GOOGL, Time: 2024-10-11 15:59:00, Data: {u'volume': u'365855', u'datetime': u'2024-10-11 15:59:00
', u'high': u'163.32500', u'low': u'163.21000', u'close': u'163.23500', u'open': u'163.21001'}
Symbol: MSFT, Time: 2024-10-11 15:59:00, Data: {u'volume': u'240175', u'datetime': u'2024-10-11 15:59:00'
, u'high': u'416.51001', u'low': u'416.20999', u'close': u'416.39001', u'open': u'416.33499'}
Symbol: TSLA, Time: 2024-10-11 15:59:00, Data: {u'volume': u'1120479', u'datetime': u'2024-10-11 15:59:00
', u'high': u'217.78999', u'low': u'217.53000', u'close': u'217.69009', u'open': u'217.55499'}
```

```
24/10/13 11:08:34 INFO instrumentation.MonitoredCounterGroup: Monitored counter
group for type: CHANNEL, name: channel1: Successfully registered new MBean.
24/10/13 11:08:34 INFO instrumentation.MonitoredCounterGroup: Component type: CH
ANNEL, name: channel1 started
24/10/13 11:08:34 INFO node.Application: Starting Sink sink1
24/10/13 11:08:34 INFO instrumentation.MonitoredCounterGroup: Monitored counter
group for type: SINK, name: sink1: Successfully registered new MBean.
24/10/13 11:08:34 INFO instrumentation.MonitoredCounterGroup: Component type: SI
NK, name: sink1 started
24/10/13 11:08:34 INFO node.Application: Starting Source source1
24/10/13 11:08:34 INFO source.NetcatSource: Source starting
24/10/13 11:08:34 INFO source.NetcatSource: Created serverSocket:sun.nio.ch.Serv
erSocketChannelImpl[/127.0.0.1:12345]
24/10/13 11:08:39 INFO hdfs.HDFSDataStream: Serializer = TEXT, UseRawLocalFileSy
stem = false
24/10/13 11:08:40 INFO hdfs.BucketWriter: Creating hdfs://localhost:8020/trainin
g/flume/stock_test/FlumeData.1728842919858.tmp
24/10/13 11:09:11 INFO hdfs.BucketWriter: Closing hdfs://localhost:8020/training
/flume/stock_test/FlumeData.1728842919858.tmp
24/10/13 11:09:11 INFO hdfs.BucketWriter: Renaming hdfs://localhost:8020/trainin
g/flume/stock_test/FlumeData.1728842919858.tmp to hdfs://localhost:8020/training
/flume/stock_test/FlumeData.1728842919858
24/10/13 11:09:11 INFO hdfs.HDFSEventSink: Writer callback called.
24/10/13 11:09:46 INFO hdfs.HDFSDataStream: Serializer = TEXT, UseRawLocalFileSystem = false
24/10/13 11:09:46 INFO hdfs.BucketWriter: Creating hdfs://localhost:8020/training/flume/stock_test/FlumeData.1728842
986342.tmp
24/10/13 11:10:16 INFO hdfs.BucketWriter: Closing hdfs://localhost:8020/training/flume/stock_test/FlumeData.17288429
8








 6342.tmp
24/10/13 11:10:16 INFO hdfs.BucketWriter: Renaming hdfs://localhost:8020/training/flume/stock_test/FlumeData.1728842
986342.tmp to hdfs://localhost:8020/training/flume/stock_test/FlumeData.1728842986342
24/10/13 11:10:16 INFO hdfs.HDFSEventSink: Writer callback called.
24/10/13 11:10:44 INFO hdfs.HDFSDataStream: Serializer = TEXT, UseRawLocalFileSystem = false
24/10/13 11:10:44 INFO hdfs.BucketWriter: Creating hdfs://localhost:8020/training/flume/stock_test/FlumeData.17288430441
58.tmp
```

Flume Configuration calling in console:

Used the command to setup and initialize the Flume agent.

```
flume-ng agent --conf /home/training --conf-file /home/training/flume.config --name a1 -Dflume.root.logger=INFO, console
```



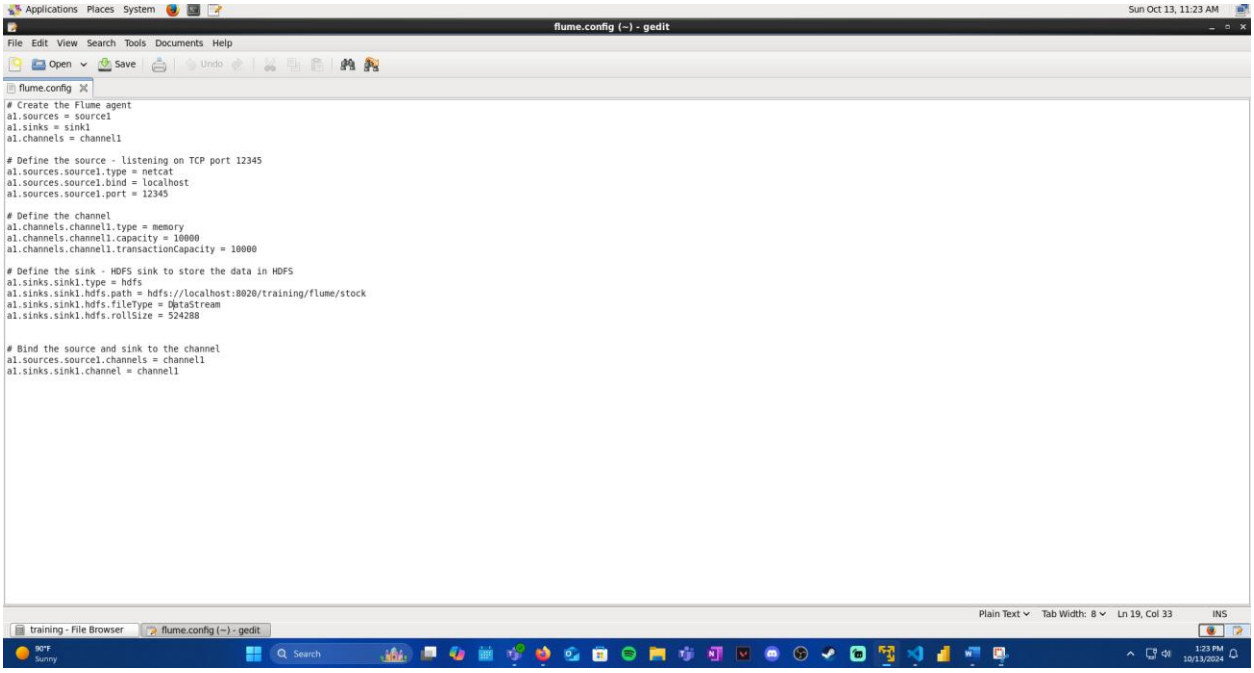
```
24/10/13 11:21:39 INFO node.PollingPropertiesFileConfigurationProvider: Reloadin
g configuration file:/home/training/flume.config
24/10/13 11:21:39 INFO conf.FlumeConfiguration: Added sinks: sink1 Agent: a1
24/10/13 11:21:39 INFO conf.FlumeConfiguration: Processing:sink1
24/10/13 11:21:39 INFO conf.FlumeConfiguration: Processing:sink1
24/10/13 11:21:39 INFO conf.FlumeConfiguration: Processing:sink1
24/10/13 11:21:39 INFO conf.FlumeConfiguration: Processing:sink1
24/10/13 11:21:39 INFO conf.FlumeConfiguration: Processing:sink1
24/10/13 11:21:39 INFO conf.FlumeConfiguration: Post-validation flume configurat
ion contains configuration for agents: [a1]
24/10/13 11:21:39 INFO node.AbstractConfigurationProvider: Creating channels
24/10/13 11:21:39 INFO channel.DefaultChannelFactory: Creating instance of chann
el channel1 type memory
24/10/13 11:21:39 INFO node.AbstractConfigurationProvider: Created channel chann
el1
24/10/13 11:21:39 INFO source.DefaultSourceFactory: Creating instance of source
source1, type netcat
24/10/13 11:21:39 INFO sink.DefaultSinkFactory: Creating instance of sink: sink1
, type: hdfs
24/10/13 11:21:39 INFO node.AbstractConfigurationProvider: Channel channel1 conn
ected to [source1, sink1]
24/10/13 11:21:39 INFO node.Application: Starting new configuration:{ sourceRunn
ers:{source1=EventDrivenSourceRunner: { source:org.apache.flume.source.NetcatSou
rce{name:source1,state:IDLE} }} sinkRunners:{sink1=SinkRunner: { policy:org.apac
he.flume.sink.DefaultSinkProcessor@397a5010 counterGroup:{ name:null counters:{}
 } }} channels:{channel1=org.apache.flume.channel.MemoryChannel{name: channel1}}
 }
24/10/13 11:21:39 INFO node.Application: Starting Channel channel1
24/10/13 11:21:39 INFO instrumentation.MonitoredCounterGroup: Monitored counter
group for type: CHANNEL, name: channel1: Successfully registered new MBean.
24/10/13 11:21:39 INFO instrumentation.MonitoredCounterGroup: Component type: CH
ANNEL, name: channel1 started
24/10/13 11:21:39 INFO node.Application: Starting Sink sink1
24/10/13 11:21:39 INFO node.Application: Starting Source source1
24/10/13 11:21:39 INFO source.NetcatSource: Source starting
24/10/13 11:21:39 INFO instrumentation.MonitoredCounterGroup: Monitored counter
group for type: SINK, name: sink1: Successfully registered new MBean.
24/10/13 11:21:39 INFO instrumentation.MonitoredCounterGroup: Component type: SI
NK, name: sink1 started
24/10/13 11:21:39 INFO source.NetcatSource: Created serverSocket:sun.nio.ch.Serv
erSocketChannelImpl[/127.0.0.1:12345]
24/10/13 11:21:51 INFO hdfs.HDFSDataStream: Serializer = TEXT, UseRawLocalFileSy
stem = false
24/10/13 11:21:51 INFO hdfs.BucketWriter: Creating hdfs://localhost:8020/trainin
g/flume/stock/FlumeData.1728843711505.tmp
24/10/13 11:22:22 INFO hdfs.BucketWriter: Closing hdfs://localhost:8020/training/flume/stock/FlumeData.1728843711505.tmp
24/10/13 11:22:22 INFO hdfs.BucketWriter: Renaming hdfs://localhost:8020/training/flume/stock/FlumeData.1728843711505.tmp to hdfs://localhost:8020/training/flume/stock/FlumeData.1728843711505
24/10/13 11:22:22 INFO hdfs.HDFSEventSink: Writer callback called.
```

Flume Configuration File:

I realized I had an error in my last file, I specified the HDFS path as “/user/training/~” where I needed to remove “User” from the path. Now it correctly outputs to HDFS using “/training/flume/stock”. I tried many variations to the sink setup, and no matter what I try I cannot seem to get the files to only role when the data reaches 512 KB as the instructions specify. I have tried adding other options and even a buffer but it would still roll before the file reached 512KB, as you will see in my HDFS screenshot below. I believe this may be due to the data not coming over fast enough/ at a larger size. I tried with this code and still had it rolling into HDFS before reaching 512KB in size:

Define the sink - HDFS sink to store the data in HDFS a1.sinks.sink1.type = hdfs a1.sinks.sink1.hdfs.path = hdfs://localhost:8020/training/flume/stock a1.sinks.sink1.hdfs.fileType = DataStream a1.sinks.sink1.hdfs.rollSize = 524288 # Set to 512 KB a1.sinks.sink1.hdfs.rollInterval = 0 # Do not roll based on time a1.sinks.sink1.hdfs.rollCount = 0 # Do not roll based on event count a1.sinks.sink1.hdfs.batchSize = 10000 # Larger batch size to ensure more data is buffered

a1. sinks.sink1.hdfs.bufferSize = 131072 # 128 KB buffer size for better performance



Here is the file I ended up executing the code with

Here is my HDFS file output: