

**ĐẠI HỌC BÁCH KHOA HÀ NỘI**  
**TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**



**SOICT**

## **BÁO CÁO**

Lưu trữ và xử lý dữ liệu lớn

*Phân tích dữ liệu giao dịch tiền ảo*

### **Nhóm HDSD**

<b>Sinh viên thực hiện</b>	<b>Mã sinh viên</b>
Nguyễn Trọng Hải	20183730
Võ Việt Dũng	20183723
Lê Hữu Tiến Dũng	20183719
Ngô Đình Sáng	20183819

Giảng viên: **TS. Đào Thành Chung**

*Hà Nội, 1 – 2022*

## MỤC LỤC

<b>MỤC LỤC</b> .....	2
<b>I. ĐẶT VẤN ĐỀ</b> .....	3
<b>II. MÔ TẢ BÀI TOÁN</b> .....	3
<b>III. TRIỂN KHAI CÀI ĐẶT</b> .....	3
1. Sơ đồ luồng hoạt động.....	3
2. Cài đặt cụm 3 máy ảo Hadoop Yarn .....	3
3. Cài đặt Spark .....	8
4. Cài đặt Kafka.....	10
<b>IV. LẬP TRÌNH VÀ THỰC THI</b> .....	10
1. Crawl và lưu trữ dữ liệu .....	10
1.1 Mã nguồn kafkaproducer.py.....	10
1.2 Mã nguồn kafkaconsumer.py.....	11
1.3 Dữ liệu được lưu trữ .....	11
2. Visualize dữ liệu.....	12
2.1 Tổng giao dịch trong 24h.....	12
2.2 Biểu đồ sự thay đổi % theo ngày trong 15 ngày.....	13
2.3 Biểu đồ sự thay đổi giá trị theo ngày trong 15 ngày .....	15
3. Kết quả application đã chạy.....	16
<b>V. KẾT LUẬN</b> .....	17
<b>VI. PHỤ LỤC</b> .....	18

## I. ĐẶT VẤN ĐỀ

Xuất phát từ nhu cầu đầu tư ngày càng tăng cao, thời gian gần đây, dự đoán xu hướng tiền ảo đang rất được quan tâm.

Sự phát triển bùng nổ của Bitcoin cũng như các đồng tiền kỹ thuật số khác đã đưa thêm vào mạng Internet rất nhiều dữ liệu mới và ngày càng trở lên rất lớn.

Ứng dụng kiến thức lưu trữ và xử lý dữ liệu lớn đã được học, nhóm muốn xử lý một đề thực tiễn, liên quan tới xu hướng của Top 200 đồng tiền ảo.

## II. MÔ TẢ BÀI TOÁN

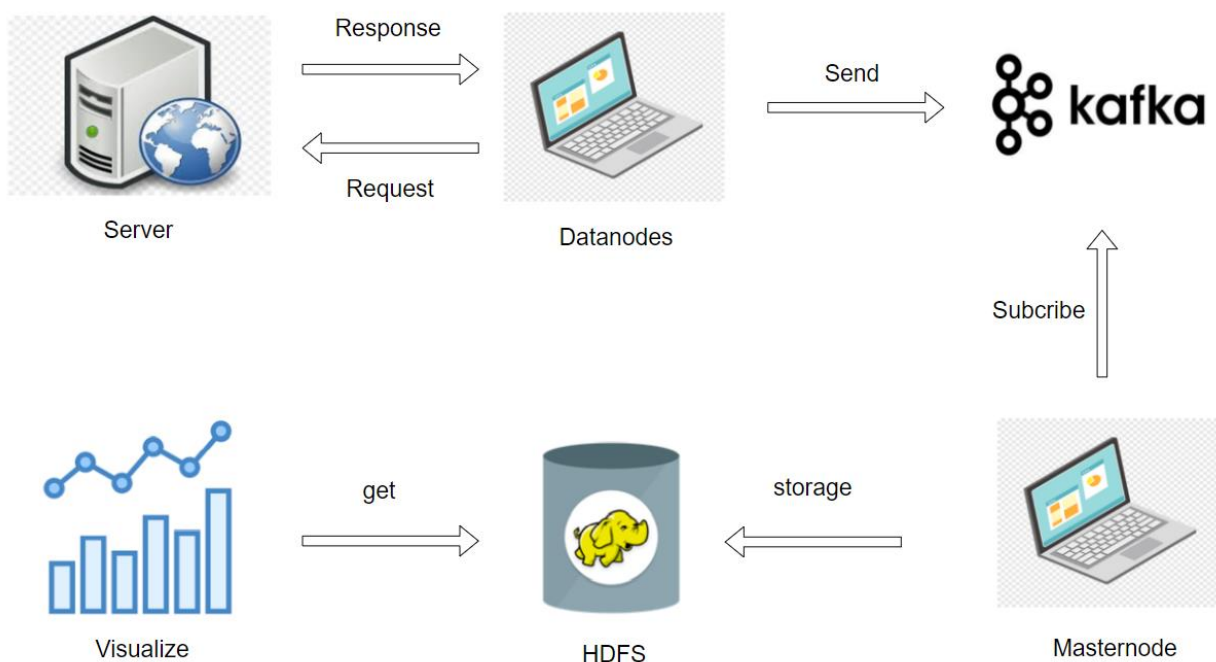
Lấy dữ liệu lịch sử các đồng tiền ảo từ ngày 28/4/2013 đến 29/12/2021.

Lưu trữ dữ liệu phân toán dưới dạng CSV.

Truy xuất dữ liệu từ file CSV để hiển thị.

## III. TRIỂN KHAI CÀI ĐẶT

### 1. Sơ đồ luồng hoạt động



### 2. Cài đặt cụm 3 máy ảo Hadoop Yarn

- masternode: 192.168.1.1
- datanode1: 192.168.1.2
- datanode2: 192.168.1.3

```

HadoopMasterNode [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal T7 14:31
tronghai@tronghai: ~
File Edit View Search Terminal Help
GNU nano 2.9.3 /etc/hosts
127.0.0.1 localhost
#127.0.1.1 tronghai
192.168.1.1 masternode
192.168.1.2 datanode1
192.168.1.3 datanode2
# The following lines are desirable for IPv6 capable hosts
::1 ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

```

- Sinh SSH-key ở masternode

```

hadoop@masternode:~$
hadoop@masternode:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/hadoop/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/hadoop/.ssh/id_rsa.
Your public key has been saved in /home/hadoop/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:uDJe3vemQKCUo+wzJRywrtamDtomYhDmvd/VrJnjv4Q hadoop@masternode
The key's randomart image is:
+---[RSA 2048]-----+
|.
| o .
|. . + .
|o+ + o o
|oo* o . S
|oo++ o +
|o++++ o .E +
|+o+* =...o* .
|o.o.o...o*+*o
+----[SHA256]-----+

```

- Copy SSH-key từ masternode đến 2 datanode

```
hadoop@masternode:~$ ssh-copy-id hadoop@datanode1
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/hadoop/.ssh/id_rsa.pub"
The authenticity of host 'datanode1 (192.168.1.2)' can't be established.
ECDSA key fingerprint is SHA256:h/F31SPUnJkoS923fs8DeRWyoxmxseF1lc6aRLqfLS0.
Are you sure you want to continue connecting (yes/no)? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter
out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
hadoop@datanode1's password:

Number of key(s) added: 1

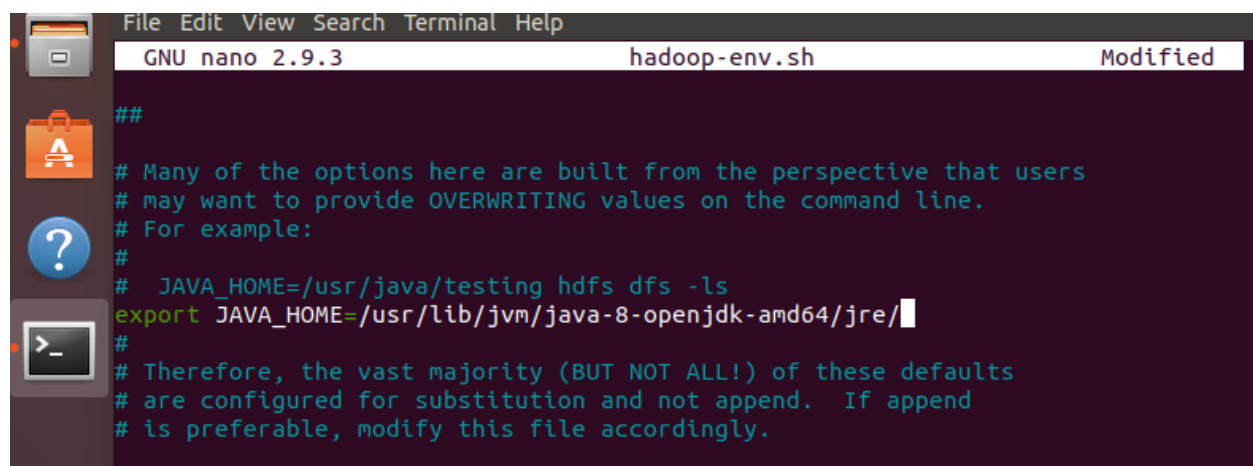
Now try logging into the machine, with:  "ssh 'hadoop@datanode1'"
and check to make sure that only the key(s) you wanted were added.
```

```
hadoop@masternode:~$ ssh-copy-id hadoop@datanode2
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/hadoop/.ssh/id_rsa.pub"
The authenticity of host 'datanode2 (192.168.1.3)' can't be established.
ECDSA key fingerprint is SHA256:h/F31SPUnJkoS923fs8DeRWyoxmxseF1lc6aRLqfLS0.
Are you sure you want to continue connecting (yes/no)? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter
out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
hadoop@datanode2's password:

Number of key(s) added: 1

Now try logging into the machine, with:  "ssh 'hadoop@datanode2'"
and check to make sure that only the key(s) you wanted were added.
```

#### - Cài đặt hadoop-env.sh



```
File Edit View Search Terminal Help
GNU nano 2.9.3 hadoop-env.sh Modified
##
# Many of the options here are built from the perspective that users
# may want to provide OVERWRITING values on the command line.
# For example:
#
# JAVA_HOME=/usr/java/testing hdfs dfs -ls
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64/jre/
#
# Therefore, the vast majority (BUT NOT ALL!) of these defaults
# are configured for substitution and not append. If append
# is preferable, modify this file accordingly.
```

#### - Cài đặt core-site.xml

```
tronghai@masternode: ~/hadoop/etc/hadoop
File Edit View Search Terminal Help
GNU nano 2.9.3 core-site.xml

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. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
  <property>
    <name>fs.default.name</name>
    <value>hdfs://masternode:9000</value>
  </property>
</configuration>
```

## - Cài đặt hdfs-site.xml

```
hadoop@masternode: ~/hadoop/etc/hadoop
File Edit View Search Terminal Help
GNU nano 2.9.3 hdfs-site.xml

<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
Licensed 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. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>/home/hadoop/hadoop/data/nameNode</value>
  </property>

  <property>
    <name>dfs.datanode.data.dir</name>
    <value>/home/hadoop/hadoop/data/dataNode</value>
  </property>

  <property>
    <name>dfs.support.append</name>
    <value>true</value>
  </property>

  <property>
    <name>dfs.replication</name>
    <value>2</value>
  </property>
</configuration>
```

## - Cài đặt yarn-site.xml

```
tronghai@masternode: ~/hadoop/etc/hadoop
File Edit View Search Terminal Help
GNU nano 2.9.3 yarn-site.xml

<?xml version="1.0"?>
<!--
Licensed 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. See accompanying LICENSE file.
-->
<configuration>

<!-- Site specific YARN configuration properties -->

    <property>
        <name>yarn.acl.enable</name>
        <value>0</value>
    </property>

    <property>
        <name>yarn.resourcemanager.hostname</name>
        <value>192.168.1.1</value>
    </property>

    <property>
        <name>yarn.nodemanager.aux-services</name>
        <value>mapreduce_shuffle</value>
    </property>
</configuration>
```

- Cài đặt worker

```
tronghai@masternode: ~/hadoop/etc/hadoop
File Edit View Search Terminal Help
GNU nano 2.9.3 workers Modified

datanode1
datanode2
```

- Cài đặt yarn-site.xml trên 2 datanode

```

HadoopDatanode1 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
GNU nano 2.9.3 yarn-site.xml Modified

<!-- Site specific YARN configuration properties -->

<property>
  <name>yarn.acl.enable</name>
  <value>0</value>
</property>

<property>
  <name>yarn.resourcemanager.hostname</name>
  <value>192.168.1.1</value>
</property>

<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>

<property>
  <name>yarn.nodemanager.disk-health-checker.max-disk-utilization-per-disk-percentage</name>
  <value>98</value>
</property>

```

```

HadoopDatanode2 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
GNU nano 2.9.3 yarn-site.xml Modified

<!-- Site specific YARN configuration properties -->

<property>
  <name>yarn.acl.enable</name>
  <value>0</value>
</property>

<property>
  <name>yarn.resourcemanager.hostname</name>
  <value>192.168.1.1</value>
</property>

<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>

<property>
  <name>yarn.nodemanager.disk-health-checker.max-disk-utilization-per-disk-percentage</name>
  <value>98</value>
</property>

```

### - Khởi động yarn

```

hadoop@masternode:~$ yarn node -list
2022-01-03 21:41:24,194 INFO client.RMProxy: Connecting to ResourceManager at /192.168.1.1:8032
Total Nodes:2
      Node-Id          Node-State Node-Http-Address      Number-of-Running-Containers
datanode2:38579      RUNNING    datanode2:8042          0
datanode1:45233      RUNNING    datanode1:8042          0
hadoop@masternode:~$

```

## 3. Cài đặt Spark

- Spark 2.4.7

### - Cấu hình .profile



```

hadoop@masternode: ~
File Edit View Search Terminal Help
GNU nano 2.9.3 .profile

# ~/.profile: executed by the command interpreter for login shells.
# This file is not read by bash(1), if ~/.bash_profile or ~/.bash_login
# exists.
# see /usr/share/doc/bash/examples/startup-files for examples.
# the files are located in the bash-doc package.

# the default umask is set in /etc/profile; for setting the umask
# for ssh logins, install and configure the libpam-umask package.
#umask 022

# if running bash
if [ -n "$BASH_VERSION" ]; then
    # include .bashrc if it exists
    if [ -f "$HOME/.bashrc" ]; then
        . "$HOME/.bashrc"
    fi
fi

# set PATH so it includes user's private bin if it exists
if [ -d "$HOME/bin" ] ; then
    PATH="$HOME/bin:$PATH"
fi

# set PATH so it includes user's private bin if it exists
if [ -d "$HOME/.local/bin" ] ; then
    PATH="$HOME/.local/bin:$PATH"
fi

PATH=/home/hadoop/hadoop/bin:/home/hadoop/hadoop/sbin:/home/hadoop/spark/bin:$PATH
export HADOOP_CONF_DIR=/home/hadoop/hadoop/etc/hadoop
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export SPARK_HOME=/home/hadoop/spark
export PYSPARK_PYTHON=/usr/bin/python3
export LD_LIBRARY_PATH=/home/hadoop/hadoop/lib/native:$LD_LIBRARY_PATH

```

## - Cấu hình spark/conf/spark-default.conf

```

hadoop@masternode: ~
File Edit View Search Terminal Help
GNU nano 2.9.3 /home/hadoop/spark/conf/spark-defaults.conf

# Default system properties included when running spark-submit.
# This is useful for setting default environmental settings.

# Example:
# spark.master                spark://master:7077
# spark.eventLog.enabled      true
# spark.eventLog.dir          hdfs://namenode:8021/directory
# spark.serializer            org.apache.spark.serializer.KryoSerializer
# spark.driver.memory         5g
# spark.executor.extraJavaOptions -XX:+PrintGCDetails -Dkey=value -Dnumbers="one two three"

spark.master                yarn
spark.driver.memory         2g
spark.yarn.am.memory        2g
spark.executor.memory       2g

spark.eventLog.enabled      true
spark.eventLog.dir          hdfs://masternode:9000/spark-logs
spark.history.provider       org.apache.spark.deploy.history.FsHistoryProvider
spark.history.fs.logDirectory hdfs://masternode:9000/spark-logs
spark.history.fs.update.interval 10s
spark.history.ui.port       18080

```

## - Khởi động thử Spark 2.4.7 bằng pyspark

```

hadoop@masternode:~$ pyspark
/home/hadoop/spark/bin/pyspark: line 45: python: command not found
Python 3.6.9 (default, Dec 8 2021, 21:08:43)
[GCC 8.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
2022-01-03 21:47:50,645 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
2022-01-03 21:47:56,658 WARN yarn.Client: Neither spark.yarn.jars nor spark.yarn.archive is set, falling back to uploading libraries under SPARK_HOME.
Welcome to

  ____      _
 / ___|  __| | | |
 \___ \  / _ \ |_| |
  ___) |/ ___ \  __/
 /_____/|___ \___/

 version 2.4.7

Using Python version 3.6.9 (default, Dec 8 2021 21:08:43)
SparkSession available as 'spark'.
>>>

```

## 4. Cài đặt Kafka

- Kafka 2.8.1

- Cài đặt Kafka vào thư mục mong muốn là có thể sử dụng được.
- Mặc định kafka giao tiếp ở cổng 9092, zookeeper giao tiếp ở cổng 2181.
- Tạo topic mới bằng kafka.
- Topic “hdsd” tại masternode:9092

```
hadoop@masternode:~$ cd kafka
hadoop@masternode:~/kafka$ bin/kafka-topics.sh --create --topic hdsd --bootstrap-server masternode:9092
Created topic hdsd.
hadoop@masternode:~/kafka$
```

- Xem danh sách các topic tại masternode:2181

```
hadoop@masternode:~/kafka$ bin/kafka-topics.sh --list --zookeeper masternode:2181
hdsd
hdsd-test
hadoop@masternode:~/kafka$
```

## IV. LẬP TRÌNH VÀ THỰC THI

### 1. Crawl và lưu trữ dữ liệu

#### 1.1 Mã nguồn kafkaproducer.py

- File **kafkaproducer.py** nằm ở các máy datanode1 và datanode2 để tiến hành crawl dữ liệu và đẩy về masternode qua kafka.
- Cần install **kafka\_python** để sử dụng kafka
- Sử dụng **BeautifulSoup** để lấy dữ liệu json.

```
from kafka import KafkaProducer
import requests
import time
from bs4 import BeautifulSoup
from datetime import timedelta, date

producer = KafkaProducer(bootstrap_servers='masternode:9092')

url_base = 'https://coinmarketcap.com/historical/'
def daterange(start_date, end_date):
    for n in range(int((end_date - start_date).days)):
        yield start_date + timedelta(n)

start_date = date(2021, 12, 27)
end_date = date(2021, 12, 29)

def runaround(single_date):
    time.sleep(20)
    url = url_base + single_date.strftime("%Y%m%d")
```

```

print(url)
r = requests.get(url)
soup = BeautifulSoup(r.text, 'html.parser')
return soup.find('script', type='application/json')

for single_date in daterange(start_date, end_date):
    url = url_base + single_date.strftime("%Y%m%d")
    print(url)
    r = requests.get(url)
    soup = BeautifulSoup(r.text, 'html.parser')
    s = soup.find('script', type='application/json')
    while s is None:
        s = runaround(single_date)
    s=s.string
    start = s.find('["id":1,')
    s=s[start:]
    end = s.find(',"page":1')
    s=s[:end]
    producer.send('hdsd', bytes(s,'utf-8'))
    producer.flush()
    time.sleep(5)

```

## 1.2 Mã nguồn kafkaconsumer.py

- File **kafkaconsumer.py** nằm ở masternode để nhận dữ liệu json từ các datanode qua kafka và lưu lại vào file csv tại local.

```

from kafka import KafkaConsumer
import pandas as pd
from json import loads

consumer = KafkaConsumer(bootstrap_servers='masternode:9092')
consumer.subscribe('hdsd')
for message in consumer:
    print(message)
    message = message.value
    json = loads(message)
    df = pd.json_normalize(json)
    df.to_csv("bigdata.csv", mode='a', header=False)

```

## 1.3 Dữ liệu được lưu trữ

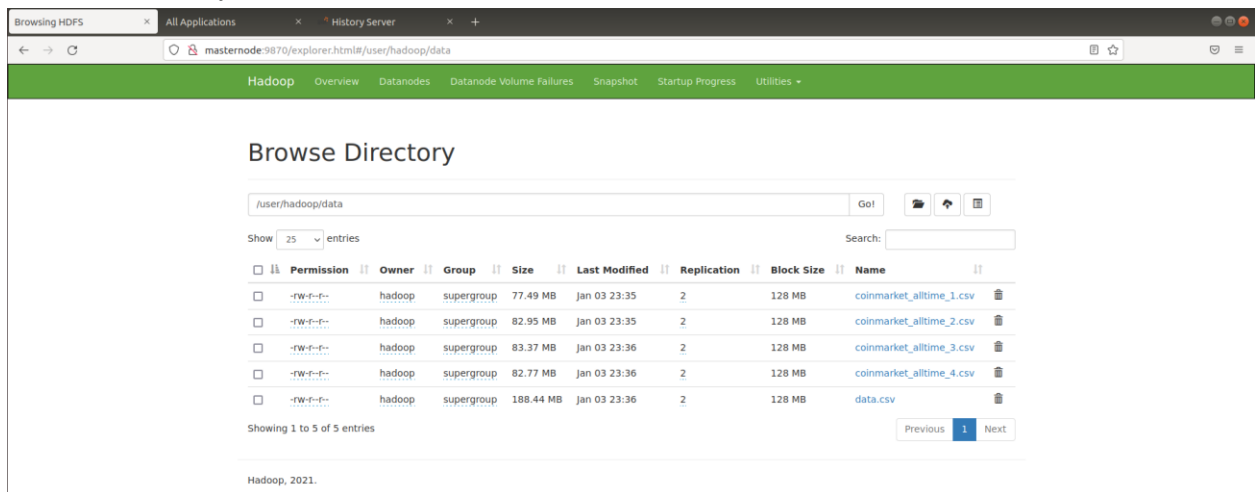
- Dữ liệu sau đó được đẩy lên HDFS.

```

hadoop@masternode:~$ hdfs dfs -ls data
Found 5 items
-rw-r--r-- 2 hadoop supergroup 81253832 2022-01-03 23:35 data/coinmarket_alltime_1.csv
-rw-r--r-- 2 hadoop supergroup 86978001 2022-01-03 23:35 data/coinmarket_alltime_2.csv
-rw-r--r-- 2 hadoop supergroup 87417187 2022-01-03 23:36 data/coinmarket_alltime_3.csv
-rw-r--r-- 2 hadoop supergroup 86788245 2022-01-03 23:36 data/coinmarket_alltime_4.csv
-rw-r--r-- 2 hadoop supergroup 197592315 2022-01-03 23:36 data/data.csv
hadoop@masternode:~$

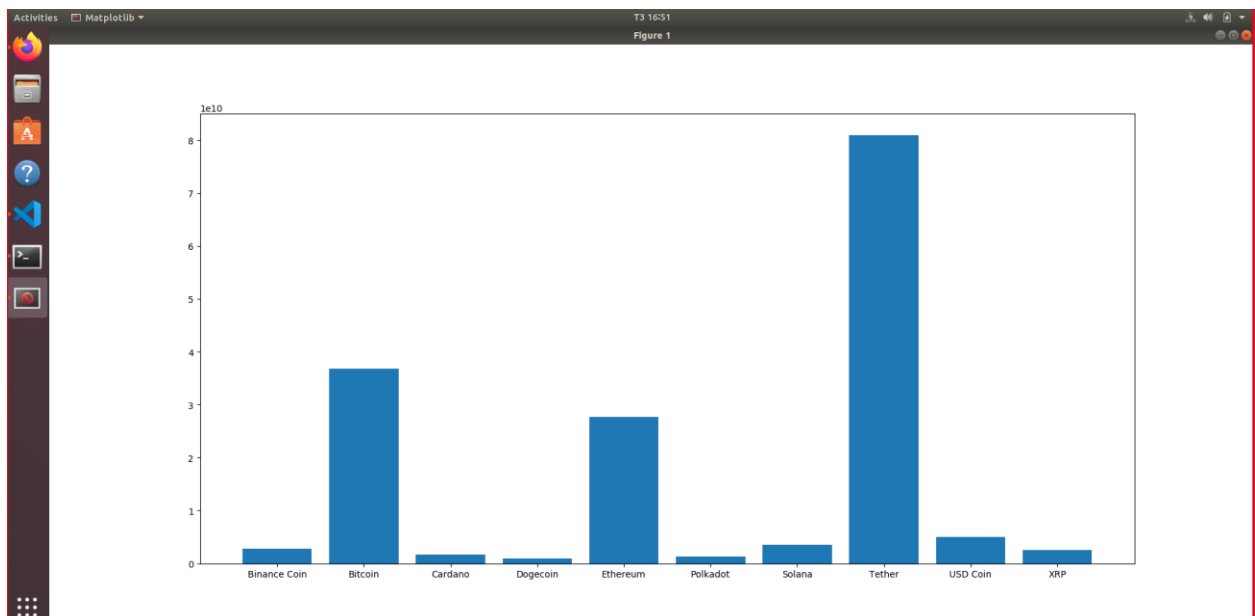
```

- Dữ liệu cỡ 512MB



## 2. Visualize dữ liệu

### 2.1 Tổng giao dịch trong 24h



- Mã nguồn **visualize1.py**

```
import sys
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from pyspark.sql.session import SparkSession
from pyspark.sql import SQLContext
from pyspark import SparkConf, SparkContext

conf = SparkConf().setAppName("HDSD_Visual")
sc = SparkContext(conf=conf)
sc.setLogLevel("WARN")
spark = SparkSession(sc)
sqlContext = SQLContext(sc)
```

```
data = spark.read.csv(sys.argv[1], sep=',',inferSchema=True, header=True)
df = data.toPandas()
df.head()

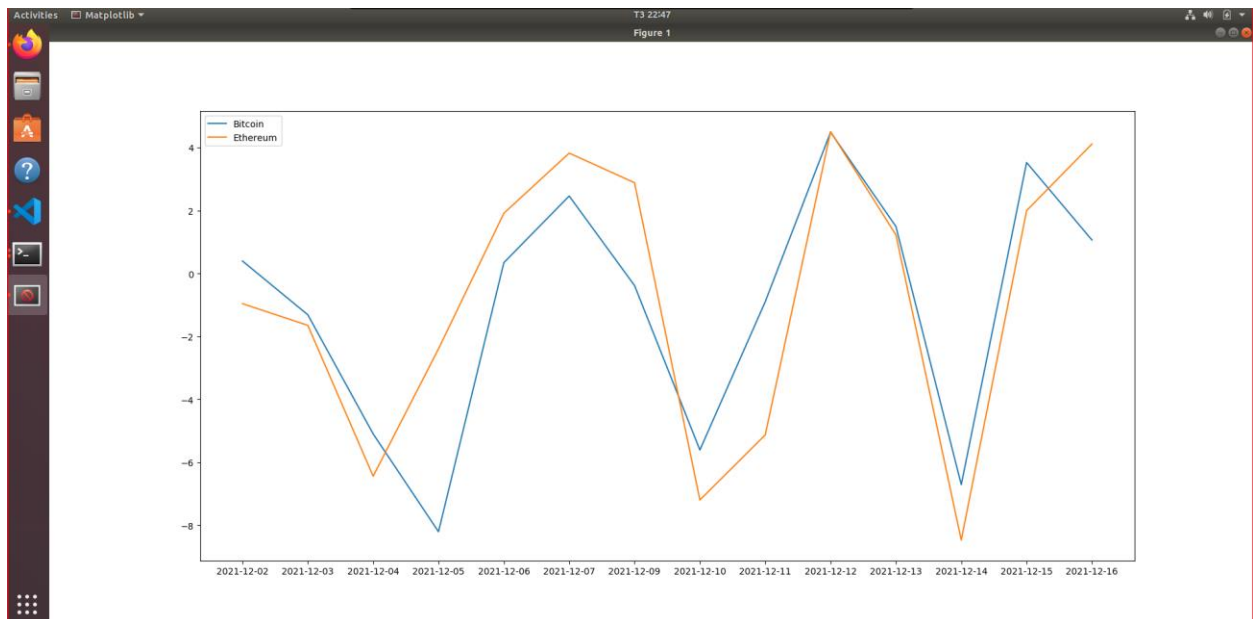
name = df['name'].head(20)
volume_24h = df['quote.USD.volume_24h'].head(20)
percent_change_1h = df['quote.USD.percent_change_1h'].head(20)

fig = plt.figure(figsize=(20, 20))
plt.bar(name[0:10], volume_24h[0:10])

# Show Plot
plt.show()
```

- `spark-submit visualize1.py hdfs://masternode:9000/user/hadoop/data/data.csv`

## 2.2 Biểu đồ sự thay đổi % theo ngày trong 15 ngày



- Mã nguồn **visualize2.py**

```
import sys
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from pyspark.sql.session import SparkSession
from pyspark.sql import SQLContext
from pyspark import SparkConf, SparkContext

conf = SparkConf().setAppName("HDSD_Visual")
sc = SparkContext(conf=conf)
sc.setLogLevel("WARN")
spark = SparkSession(sc)
```

```
sqlContext = SQLContext(sc)

column = ['name', 'quote.USD.percent_change_24h', 'last_updated']
data = spark.read.csv(sys.argv[1], sep=',', inferSchema=True, header=True)
df = data.toPandas()
df.head()

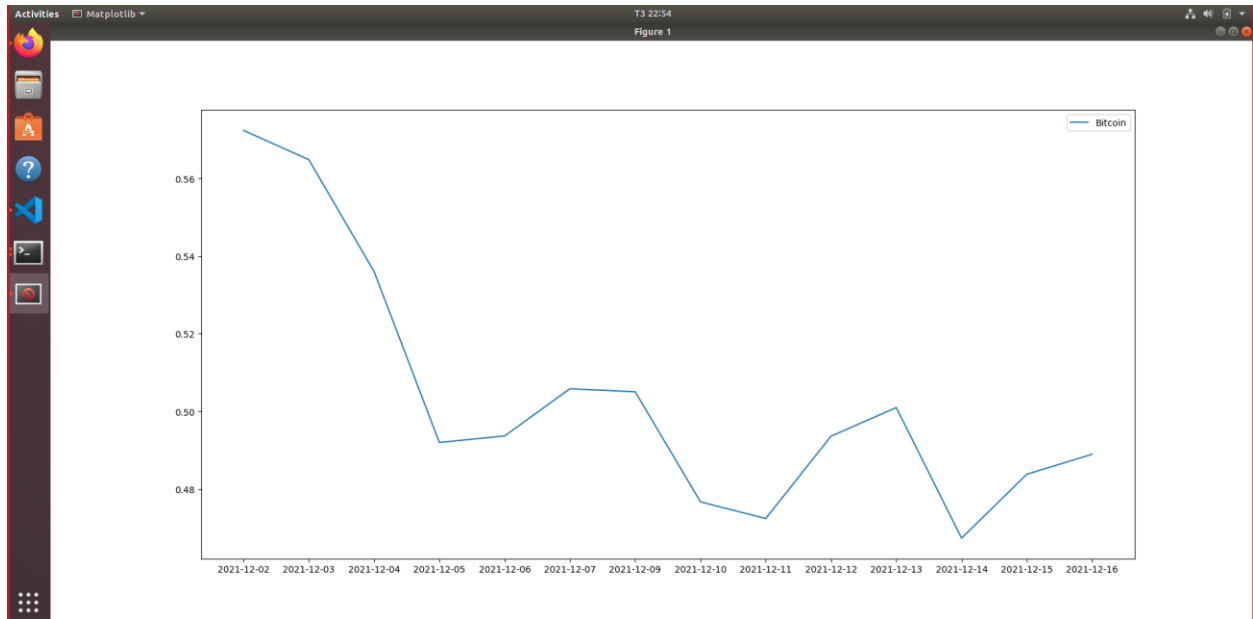
dataLine = df[column]
dataLine = dataLine.to_numpy()
dataLine.shape
percent_change_24h_Bitcoin = []
date_Bitcoin = []
percent_change_24h_Ethereum = []
date_Ethereum = []
for line in dataLine:
    if (line[0] == "Bitcoin"):
        percent_change_24h_Bitcoin.append(line[1])
        day = line[2]
        day = str(day)
        date_Bitcoin.append(day[0:10])
    elif (line[0] == "Ethereum"):
        percent_change_24h_Ethereum.append(line[1])
        day = line[2]
        day = str(day)
        date_Ethereum.append(day[0:10])

# Figure Size
fig = plt.figure(figsize=(30, 20))

plt.plot(date_Bitcoin[0:15], percent_change_24h_Bitcoin[0:15], label='Bitcoin')
plt.plot(date_Ethereum[0:15],
         percent_change_24h_Ethereum[0:15], label='Ethereum')
plt.legend()
# Show Plot
plt.show()
```

- *spark-submit visualize2.py hdfs://masternode:9000/user/hadoop/data/data.csv*

## 2.3 Biểu đồ sự thay đổi giá trị theo ngày trong 15 ngày



- Mã nguồn **visualize3.py**

```
import sys
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from pyspark.sql.session import SparkSession
from pyspark.sql import SQLContext
from pyspark import SparkConf, SparkContext

conf = SparkConf().setAppName("HDSD_Visual")
sc = SparkContext(conf=conf)
sc.setLogLevel("WARN")
spark = SparkSession(sc)
sqlContext = SQLContext(sc)

column = ['name', 'quote.USD.price', 'last_updated']
data = spark.read.csv(sys.argv[1], sep=',', inferSchema=True, header=True)
df = data.toPandas()
df.head()

dataLine = df[column]
dataLine = dataLine.to_numpy()
dataLine.shape
price_Bitcoin = []
date_Bitcoin = []
price_Ethereum = []
date_Ethereum = []
for line in dataLine:
    if (line[0] == "Bitcoin"):
        price_Bitcoin.append(line[1]/100000)
```

```

    day = line[2]
    day = str(day)
    date_Bitcoin.append(day[0:10])
    elif (line[0] == "Ethereum"):
        price_Ethereum.append(line[1])
        day = line[2]
        day = str(day)
        date_Ethereum.append(day[0:10])

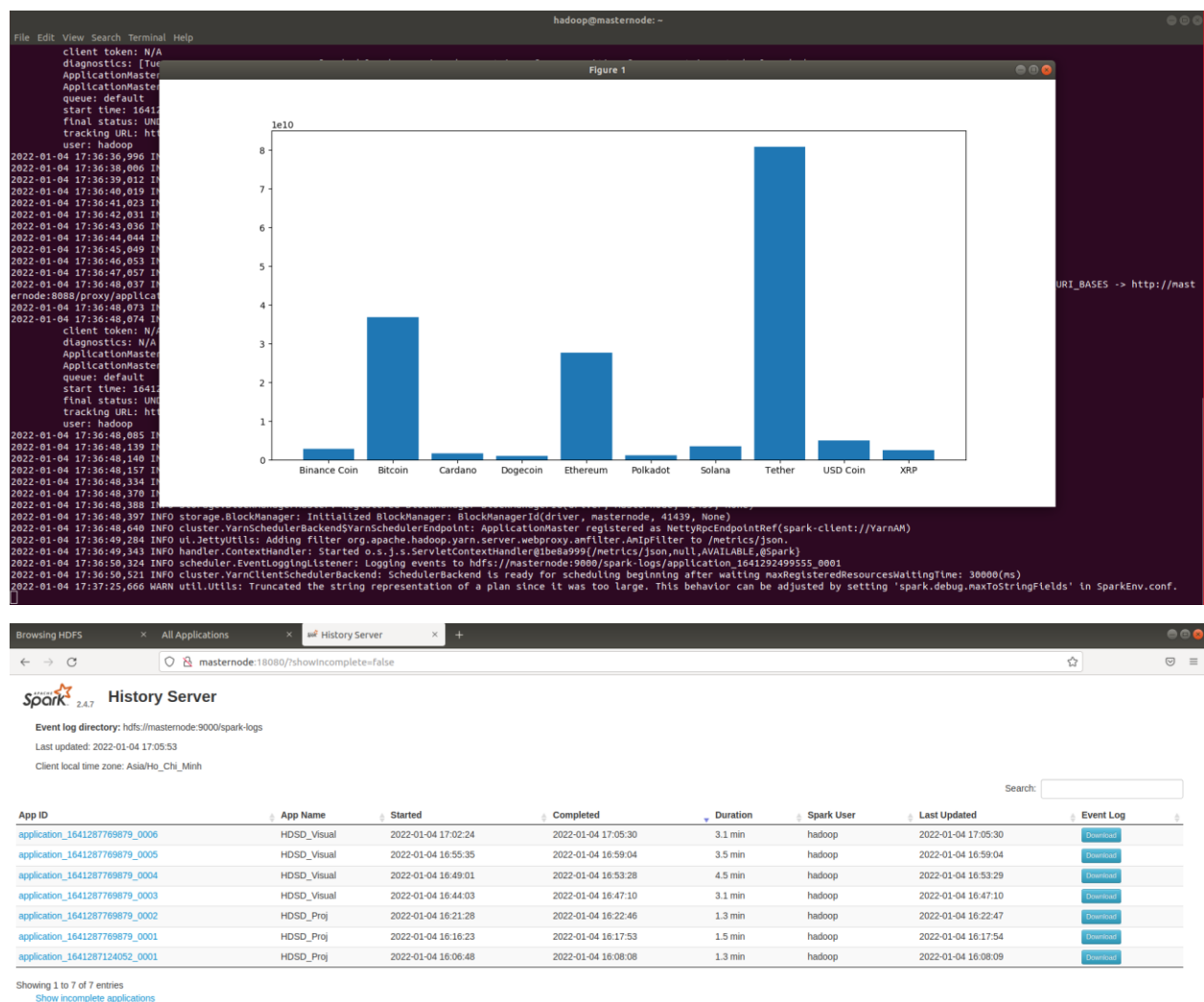
# Figure Size
fig = plt.figure(figsize=(30, 20))

plt.plot(date_Bitcoin[0:15], price_Bitcoin[0:15], label='Bitcoin')
plt.legend()
# Show Plot
plt.show()

```

- `spark-submit visualize3.py hdfs://masternode:9000/user/hadoop/data/data.csv`

### 3. Kết quả application đã chạy





**All Applications**

Cluster Metrics		Apps Pending		Apps Running		Apps Completed		Containers Running		Used Resources		Total Resources		Reserved Resources	
6	0	0	0	6	0	0	0	<memory:0, vCores:0>	<memory:16384, vCores:16>	<memory:0, vCores:0>	<memory:0, vCores:0>	<memory:0, vCores:0>	<memory:0, vCores:0>	<memory:0, vCores:0>	<memory:0, vCores:0>

Cluster Nodes Metrics		Decommissioning Nodes		Decommissioned Nodes		Lost Nodes		Unhealthy Nodes	
2	0	0	0	0	0	0	0	0	0

Scheduler Metrics		Scheduling Resource Type		Minimum Allocation		Maximum Allocation	
Capacity Scheduler	[memory-mb (unit=Mi), vcores]	<memory:1024, vCores:1>	<memory:8192, vCores:4>				0

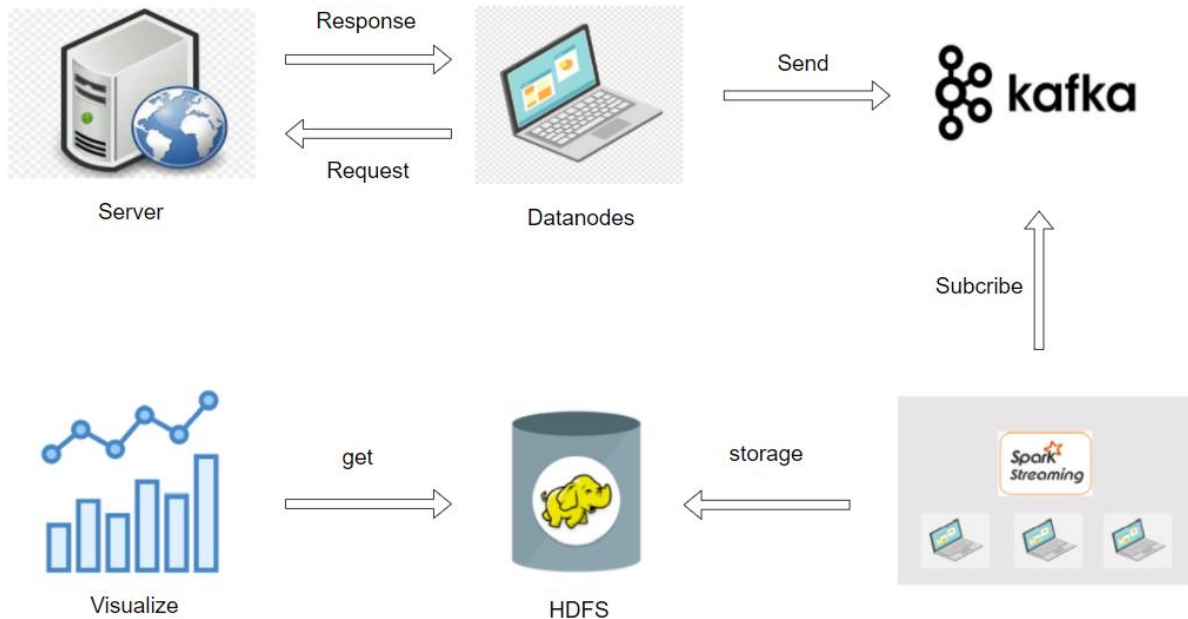
ID	User	Name	Application Type	Queue	Application Priority	StartTime	LaunchTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU Vcores	Allocated Memory MB	Allocated GPUs	Reserved CPU Vcores	Reserved Memory MB
application_1641287769879_0006	hadoop	HDSD_Visual	SPARK	default	0	Tue Jan 4 17:02:52 +0700 2022	Tue Jan 4 17:02:53 +0700 2022	Tue Jan 4 17:05:30 +0700 2022	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	N/A
application_1641287769879_0005	hadoop	HDSD_Visual	SPARK	default	0	Tue Jan 4 16:56:09 +0700 2022	Tue Jan 4 16:56:10 +0700 2022	Tue Jan 4 16:59:04 +0700 2022	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	N/A
application_1641287769879_0004	hadoop	HDSD_Visual	SPARK	default	0	Tue Jan 4 16:49:34 +0700 2022	Tue Jan 4 16:49:35 +0700 2022	Tue Jan 4 16:53:29 +0700 2022	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	N/A
application_1641287769879_0003	hadoop	HDSD_Visual	SPARK	default	0	Tue Jan 4 16:44:36 +0700 2022	Tue Jan 4 16:44:37 +0700 2022	Tue Jan 4 16:47:10 +0700 2022	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	N/A
application_1641287769879_0002	hadoop	HDSD_Proj	SPARK	default	0	Tue Jan 4 16:21:54 +0700 2022	Tue Jan 4 16:21:55 +0700 2022	Tue Jan 4 16:22:47 +0700 2022	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	N/A
application_1641287769879_0001	hadoop	HDSD_Proj	SPARK	default	0	Tue Jan 4 16:16:49 +0700 2022	Tue Jan 4 16:16:52 +0700 2022	Tue Jan 4 16:17:54 +0700 2022	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	N/A

## V. KẾT LUẬN

- Nhóm đã vận dụng và tự tìm hiểu một số kiến thức trong lưu trữ và xử lý dữ liệu lớn.
- Cấu hình được Hadoop Yarn, Spark, Kafka.
- Lưu trữ dữ liệu trên HDFS và visualize sử dụng spark-submit trên Yarn.
- Qua phân tích lịch sử tiền ảo, nhóm nhận thấy tiền ảo biến động mạnh, xu hướng vô cùng khó đoán trước.
- Source: <https://github.com/tronghaiit2/CryptoCurrencyHistory>

## VI. PHỤ LỤC

### - Sử dụng Kafka kết hợp với SparkStreaming



### - Đẩy dữ liệu qua kafka - spark streaming

The screenshot shows a terminal window with the following content:

```

File Edit View Search Terminal Help
Time: 2022-01-04 10:00:42
-----
id      name    platform.slug    platform.token_address
0       1       Bitcoin         NaN
1       1027    Ethereum        NaN
2       1839    Binance Coin    NaN
3       825     Tether          NaN
4       5426    Solana          NaN
...
195     4041    MX TOKEN        ethereum 0x11eef04c884e24d9b7b4760e7476d06dd7f797f36
196     3673    ASD             ethereum 0xff742d05420b6aca4481f635ad8341f81a6308c2
197     2297    StormX          ethereum 0xb59375c6a42802e8258962efb95551a5b722803
198     2758    Unlbrigt        ethereum 0x8400d94a5cbf8a0d041a3788e395285d61c9e5e
199     3441    Dlvl            NaN
-----
[200 rows x 35 columns]
Time: 2022-01-04 10:00:45
-----
Time: 2022-01-04 10:00:48
-----
id      name    platform.slug    platform.token_address
0       1       Bitcoin         NaN
1       1027    Ethereum        NaN
2       1839    Binance Coin    NaN
3       825     Tether          NaN
4       5426    Solana          NaN
...
195     3673    ASD             ethereum 0xff742d05420b6aca4481f635ad8341f81a6308c2
196     4041    MX TOKEN        ethereum 0x11eef04c884e24d9b7b4760e7476d06dd7f797f36
197     2297    StormX          ethereum 0xb59375c6a42802e8258962efb95551a5b722803
198     3441    Dlvl            NaN
199     7064    Bakerytokens    binance coin 0xE02df9e3e6220e8d069fb838b799E3F168902c5
-----
[200 rows x 35 columns]
Time: 2022-01-04 10:00:51
-----
PZ
(2)+ Stopped spark-submit --packages org.apache.spark:spark-streaming-kafka-0-8_2.11:2.4.7 kafkasparkstreaming.py hdfs://masternode:9000/user/hadoop/data/data.csv

```

### - Source sparkstreaming.py

- File **sparkstreaming.py** nằm ở masternode để nhận dữ liệu json từ các datanode qua kafka và lưu lại vào file csv tại HDFS.

```

import sys
import os
import pandas as pd
from json import loads
packages = "org.apache.spark:spark-streaming-kafka-0-8_2.11:2.4.7"

```

```
os.environ["PYSPARK_SUBMIT_ARGS"] = ("--packages {0} pyspark-  
shell".format/packages))  
from pyspark.sql.types import *  
from pyspark import SparkContext, SparkConf  
from pyspark.streaming import StreamingContext  
from pyspark.streaming.kafka import KafkaUtils  
from pyspark.sql import SQLContext  
from pyspark.sql.session import SparkSession  
  
def store_data(sqlContext, value):  
    data = str(value)  
    if data is not None:  
        json = loads(data)  
        df = pd.json_normalize(json)  
        df.to_csv(sys.argv[1], mode='a', header=False)  
        return df  
  
conf = SparkConf().setAppName("HDSD_Proj")  
sc = SparkContext(conf=conf)  
sc.setLogLevel("WARN")  
spark = SparkSession(sc)  
sqlContext = SQLContext(sc)  
ssc = StreamingContext(sc, 3)  
  
KAFKA_BROKER = "masternode:9092"  
KAFKA_TOPIC = "hdsd"  
print("HDSD")  
kafkaStream = KafkaUtils.createDirectStream(ssc,[KAFKA_TOPIC],{"metadata.broker.l  
ist":KAFKA_BROKER})  
lines = kafkaStream.map(lambda value: store_data(sqlContext, value[1]))  
lines.pprint()  
  
ssc.start()  
ssc.awaitTermination()
```