

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



BÁO CÁO

LƯU TRỮ VÀ XỬ LÝ DỮ LIỆU LỚN

Đề tài: Xây dựng hệ thống Crawl dữ liệu nhà đất

Visualize, analyze dữ liệu thu thập được

GVHD: TS. Đào Thành Chung

Nhóm Sinh viên thực hiện:

Phạm Văn Hạnh – MSSV: 20183525

Bùi Mạnh Trường – MSSV: 20183646

Phùng Minh Hiếu – MSSV: 20183536

Lương Hoàng Đức – MSSV: 20183712

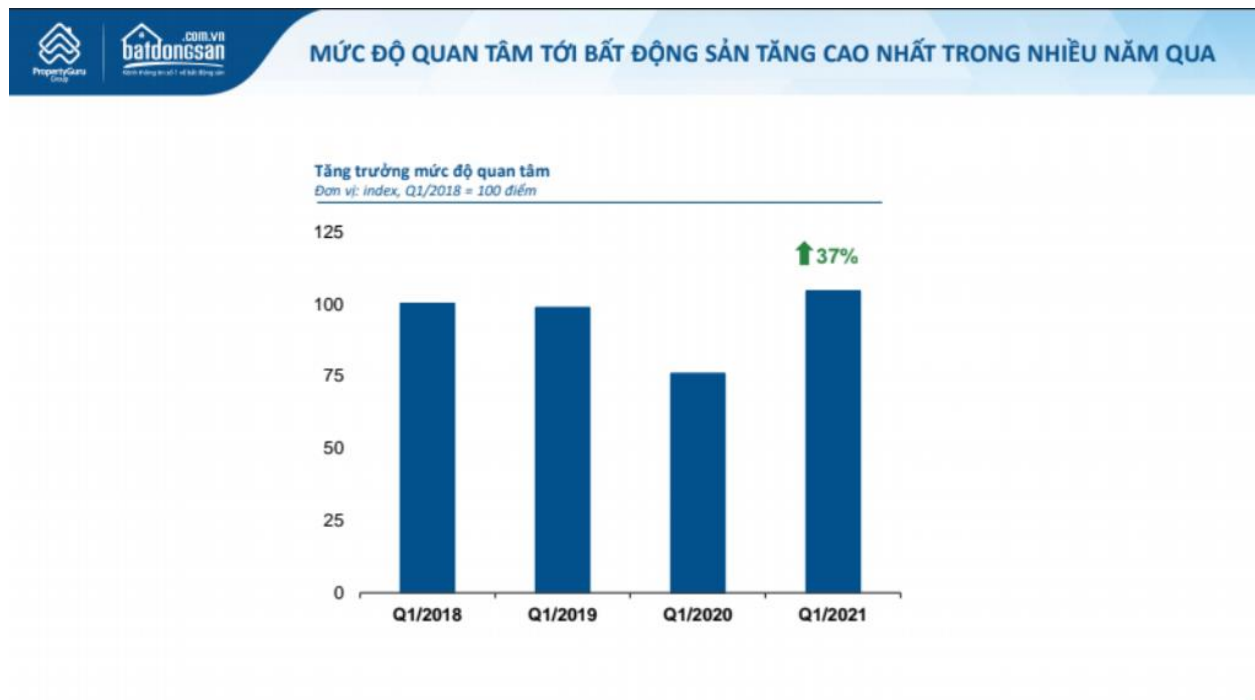
Hà Nội, 01 – 2022

ĐẶT VẤN ĐỀ

1. Giới thiệu bài toán.
2. Mô hình hệ thống.
3. Cấu hình hệ thống, quản lý, chạy chương trình.
4. Kết quả đạt được.
5. Visualize dữ liệu.
6. Tài liệu tham khảo.

1. Giới thiệu bài toán

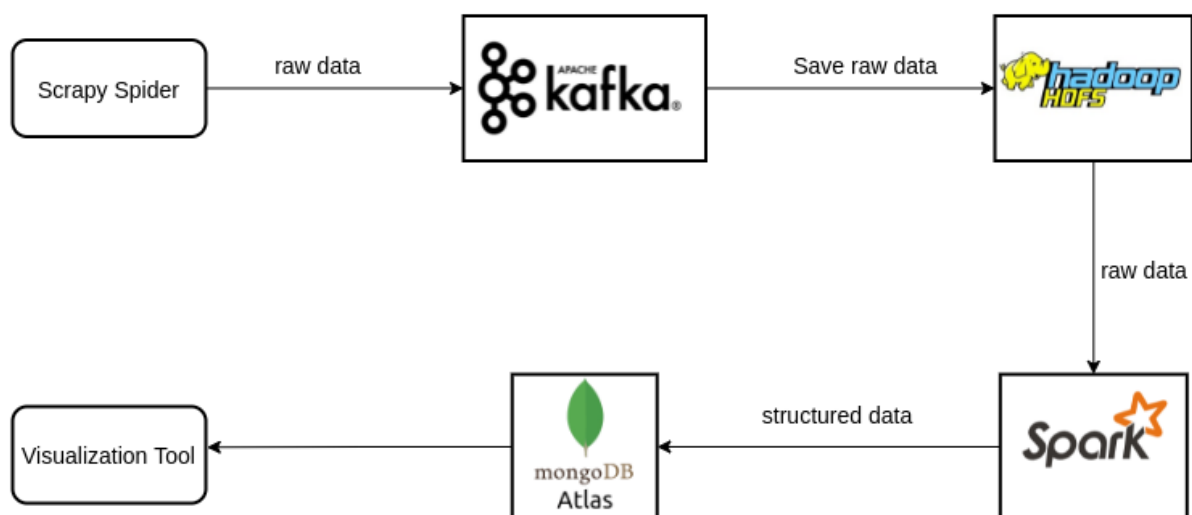
- Số lượng người đổ dồn về thành thị, đặc biệt là Hà Nội ngày càng tăng. Điều này dẫn đến sự bùng nổ về nhu cầu tìm nhà. Trong đó, phải kể đến các nhà đất mặt phố khi chúng ngày càng khan hiếm và đắt đỏ
- Nhu cầu tăng, dẫn đến mặc dù có nhiều tin tức, từ rất nhiều nguồn, nhưng chúng ta lại rất khó để tìm cho mình một căn hộ ưng ý, phù hợp với các tiêu chí của người mua nhà mà lại hợp túi tiền
- Vì vậy nhóm quyết định chọn bài toán : ***“Thu thập dữ liệu lớn thông tin về các căn hộ và phân tích nhu cầu sử dụng và mua bán”*** trong ngành này.



2. Mô hình hệ thống

Mô hình hệ thống của nhóm đưa ra nhằm đáp ứng nhu cầu crawl được lượng dữ liệu lớn, giả lập có khả năng chịu lỗi, tính khả mở, có thể scale trong tương lai.

- **Crawler:** Nhóm sử dụng web-crawling framework là Scrapy và xây dựng các spider để crawl dữ liệu từ alonthadat.com
- **Kafka:** Là hàng đợi để tránh việc lưu vào HDFS bị quá tải hoặc trong trường hợp lưu vào HDFS bị lỗi sẽ gây ảnh hưởng đến hệ thống đang chạy, data tại đây được lưu dưới dạng raw để tránh mất mát thông tin.
- **Cụm Spark:** Đảm nhiệm việc xử lý, clean dữ liệu, trích xuất ra dữ liệu có cấu trúc rồi đẩy vào MongoDB
- **MongoDB:** Nhóm sử dụng dưới dạng cloud trên MongoDB Atlas, replica set 3 nodes
- **Visualization tool:** MongoDB Charts connect với MongoDB để xem và phân tích dữ liệu



Hệ thống gồm:

- 1 Scrapy Spider để crawl dữ liệu từ trang alonthadat.com
- Cụm Apache Kafka gồm: 1 Zookeeper node, 3 kafka nodes tương ứng với 3 broker
- Cụm Hadoop gồm: 1 namenode, 1 datanode, 1 resource manager node, 1 node manager, 1 history node
- Cụm spark gồm: 1 master node, 2 worker nodes (2 Cores, 3GB Ram/ worker nodes)
- Cụm MongoDB Atlas được cấu hình Replica Set là 3 nodes

Mỗi phần được đóng gói thành một docker service bằng cách sử dụng docker-compose, các service được cấu hình trong cùng network để có thể dễ dàng giao tiếp với nhau.

3. Các bước cài đặt cụ thể:

3.1. Cài đặt docker và docker-compose:

- Cài đặt docker:
 - Step 1: Update Software Repositories
`sudo apt-get update`
 - Step 2: Uninstall Old Versions of Docker
`sudo apt-get remove docker docker-engine docker.io`
 - Step 3: Install Docker on Ubuntu 18.04
`sudo apt install docker.io`
 - Step 4: Start and Automate Docker
`sudo systemctl start docker`
`sudo systemctl enable docker`
- Cài đặt docker-compose
 - Step 1: Download the Docker Compose binary into the /usr/local/bin directory with the following [curl](#) command:
`sudo curl -L "https://github.com/docker/compose/releases/download/1.23.1/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose`
 - Step 2: Once the download is complete, apply executable [permissions](#) to the Compose binary:
`sudo chmod +x /usr/local/bin/docker-compose`
 - Step 3: Verify the installation by running the following command which will display the Compose version:
`docker-compose --version`

3.2 Cấu hình Scrapy Spider để đẩy dữ liệu vào Kafka

```
class AlonhadatPipeline:
    def process_item(self, item: AlonhadatNewsItem, spider: Spider):

        def delivery_report(err, msg):
            """ Called once for each message produced to indicate delivery result.
                Triggered by poll() or flush(). """
            if err is not None:
                print('Message delivery failed: {}'.format(err))
            else:
                print('Message delivered to {} [{}]'
                      .format(msg.topic(), msg.partition()))

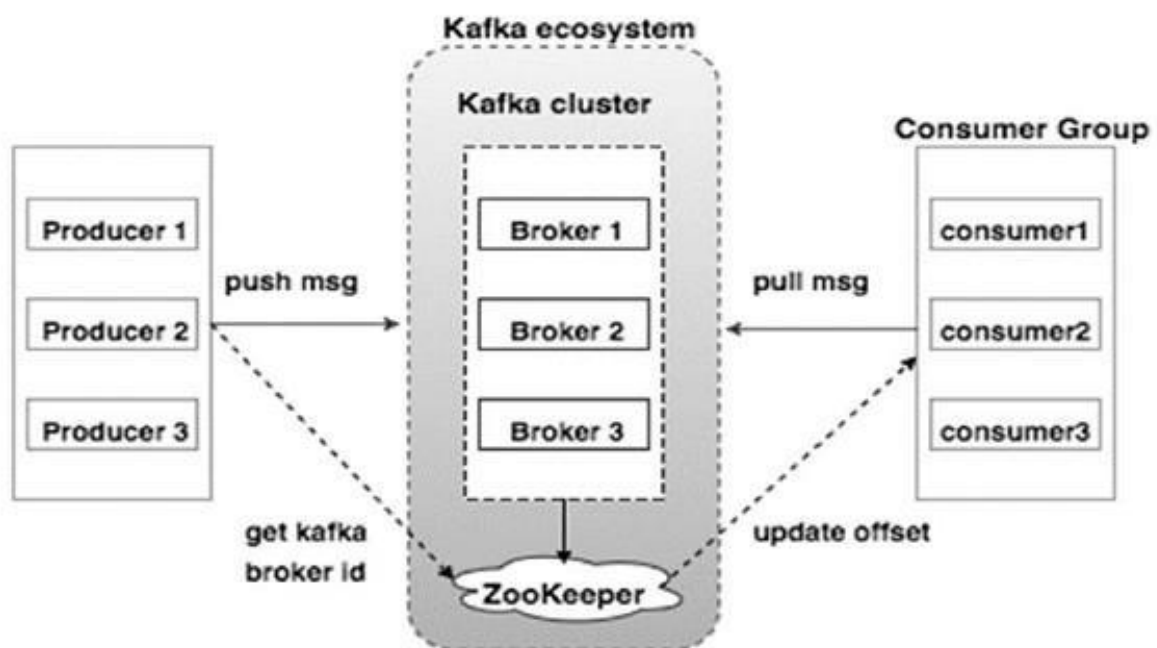
        spider.producer.poll(0)
        spider.producer.produce(spider.topic,
                                json.dumps(dataclasses.asdict(item)).encode('utf-8'),
                                callback=delivery_report)

        spider.producer.flush()
```

Viết Pipeline Class để lưu đẩy dữ liệu vào kafa. Sau đó config ITEM_PIPELINES trong setting.py của Scrapy.

3.3 Cấu hình kafka

Sử dụng docker-compose file để xây dựng kafka service với 1 Zookeeper node, 3 Kafka nodes



```

version: '2.1'

services:
  zoo1:
    image: confluentinc/cp-zookeeper:6.2.1
    hostname: zoo1
    container_name: zoo1
    ports:
      - "2181:2181"
    environment:
      ZOOKEEPER_CLIENT_PORT: 2181
      ZOOKEEPER_SERVER_ID: 1
      ZOOKEEPER_SERVERS: zoo1:2888:3888

  kafka1:
    image: confluentinc/cp-kafka:6.2.1
    hostname: kafka1
    ports:
      - "9092:9092"
    environment:
      KAFKA_ADVERTISED_LISTENERS: LISTENER_DOCKER_INTERNAL://kafka1:19092,LISTENER_DOCKER_EXTERNAL://{DOCKER_HOST_IP:-127.0.0.1}:9092
      KAFKA_LISTENER_SECURITY_PROTOCOL_MAP: LISTENER_DOCKER_INTERNAL:PLAINTEXT,LISTENER_DOCKER_EXTERNAL:PLAINTEXT
      KAFKA_INTER_BROKER_LISTENER_NAME: LISTENER_DOCKER_INTERNAL
      KAFKA_ZOOKEEPER_CONNECT: "zoo1:2181"
      KAFKA_BROKER_ID: 1
      KAFKA_LOG4J_LOGGERS: "kafka.controller=INFO,kafka.producer.async.DefaultEventHandler=INFO,state.change.logger=INFO"
    depends_on:
      - zoo1

  kafka2:
    image: confluentinc/cp-kafka:6.2.1
    hostname: kafka2
    ports:
      - "9093:9093"
    environment:
      KAFKA_ADVERTISED_LISTENERS: LISTENER_DOCKER_INTERNAL://kafka2:19093,LISTENER_DOCKER_EXTERNAL://{DOCKER_HOST_IP:-127.0.0.1}:9093
      KAFKA_LISTENER_SECURITY_PROTOCOL_MAP: LISTENER_DOCKER_INTERNAL:PLAINTEXT,LISTENER_DOCKER_EXTERNAL:PLAINTEXT
      KAFKA_INTER_BROKER_LISTENER_NAME: LISTENER_DOCKER_INTERNAL
      KAFKA_ZOOKEEPER_CONNECT: "zoo1:2181"
      KAFKA_BROKER_ID: 2
      KAFKA_LOG4J_LOGGERS: "kafka.controller=INFO,kafka.producer.async.DefaultEventHandler=INFO,state.change.logger=INFO"
    depends_on:
      - zoo1

  kafka3:
    image: confluentinc/cp-kafka:6.2.1
    hostname: kafka3
    ports:
      - "9094:9094"
    environment:
      KAFKA_ADVERTISED_LISTENERS: LISTENER_DOCKER_INTERNAL://kafka3:19094,LISTENER_DOCKER_EXTERNAL://{DOCKER_HOST_IP:-127.0.0.1}:9094
      KAFKA_LISTENER_SECURITY_PROTOCOL_MAP: LISTENER_DOCKER_INTERNAL:PLAINTEXT,LISTENER_DOCKER_EXTERNAL:PLAINTEXT
      KAFKA_INTER_BROKER_LISTENER_NAME: LISTENER_DOCKER_INTERNAL

```

Tạo **crawled_news** topic với cấu hình 3 partitions tương ứng được phân tán trên 3 kafka (broker), 3 replicas tương ứng mỗi dữ liệu được tạo 3 bản sao

Kafka: Kafka connection										
Topics		Name	Replicas	Partitions	In sync replicas	Replication factor	Under replicated partitions	Partitions	Configuration	
Consumers	III	crawled_news	9	3	9	3	0	III	Partition id	Leader
		mytopic	1	1	1	1	0		In sync replicas count	Replicas count
								0	3	3
								1	1	3
								2	2	3
Add producer										
Add consumer										

3.4 Cấu hình cụm HDFS

Sử dụng docker-compose file để xây dựng HDFS service với 1 namenode, 1 datanode, 1 resource manager node, 1 node manager, 1 history node

```

version: "3"

services:
  namenode:
    image: bde2020/hadoop-namenode:2.0.0-hadoop3.2.1-java8
    container_name: namenode
    restart: always
    ports:
      - 9870:9870
      - 9000:9000
    volumes:
      - hadoop_namenode:/hadoop/dfs/name
    environment:
      - CLUSTER_NAME=test
    env_file:
      - ./hadoop.env

  datanode:
    image: bde2020/hadoop-datanode:2.0.0-hadoop3.2.1-java8
    container_name: datanode
    restart: always
    ports:
      - 9864:9864
    volumes:
      - hadoop_datanode:/hadoop/dfs/data
    environment:
      SERVICE_PRECONDITION: "namenode:9870"
    env_file:
      - ./hadoop.env

  resourcemanager:
    image: bde2020/hadoop-resourcemanager:2.0.0-hadoop3.2.1-java8
    container_name: resourcemanager
    restart: always
    ports:
      - 8088:8088
    environment:
      SERVICE_PRECONDITION: "namenode:9000 namenode:9870 datanode:9864"
    env_file:
      - ./hadoop.env

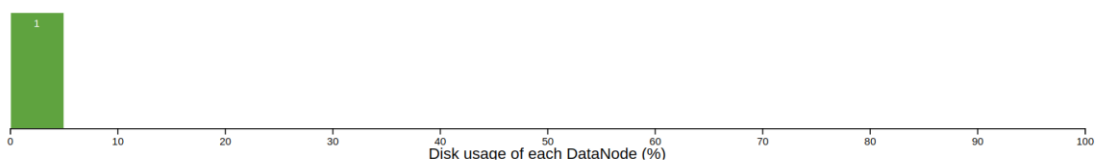
  nodemanager1:
    image: bde2020/hadoop-nodemanager:2.0.0-hadoop3.2.1-java8
    container_name: nodemanager
    restart: always
    environment:
      SERVICE_PRECONDITION: "namenode:9000 namenode:9870 datanode:9864 resourcemanager:8088"
    env_file:
      - ./hadoop.env

  historyserver:
    image: bde2020/hadoop-historyserver:2.0.0-hadoop3.2.1-java8
    container_name: historyserver
    restart: always

```

Monitoring nodes

Datanode usage histogram



In operation

Show 25 entries Search:

Node	Http Address	Last contact	Last Block Report	Capacity	Blocks	Block pool used	Version
✓ 7f5946e6547b:9866 (172.22.0.3:9866)	http://7f5946e6547b:9864	0s	28m	468.44 GB <div><div></div></div>	64811	6.94 GB (1.48%)	3.2.1

Showing 1 to 1 of 1 entries Previous 1 Next

3.5 Cấu hình cụm Spark

Sử dụng docker-compose file để xây dựng cụm Spark với 1 master node, 2 worker nodes (2 Cores, 3GB Ram/ worker nodes)

```
version: "3.3"
services:
  spark-master:
    image: cluster-apache-spark:3.0.2
    ports:
      - "9100:8080"
      - "7077:7077"
    volumes:
      - ./apps:/opt/spark-apps
      - ./data:/opt/spark-data
    environment:
      - SPARK_LOCAL_IP=spark-master
      - SPARK_WORKLOAD=master
  spark-worker-a:
    image: cluster-apache-spark:3.0.2
    ports:
      - "9110:8080"
      - "7000:7000"
    depends_on:
      - spark-master
    environment:
      - SPARK_MASTER=spark://spark-master:7077
      - SPARK_WORKER_CORES=2
      - SPARK_WORKER_MEMORY=3G
      - SPARK_DRIVER_MEMORY=2G
      - SPARK_EXECUTOR_MEMORY=2G
      - SPARK_WORKLOAD=worker
      - SPARK_LOCAL_IP=spark-worker-a
    volumes:
      - ./apps:/opt/spark-apps
      - ./data:/opt/spark-data
  spark-worker-b:
    image: cluster-apache-spark:3.0.2
    ports:
      - "9120:8080"
      - "7001:7000"
    depends_on:
      - spark-master
    environment:
      - SPARK_MASTER=spark://spark-master:7077
      - SPARK_WORKER_CORES=2
      - SPARK_WORKER_MEMORY=3G
      - SPARK_DRIVER_MEMORY=2G
      - SPARK_EXECUTOR_MEMORY=2G
      - SPARK_WORKLOAD=worker
      - SPARK_LOCAL_IP=spark-worker-b
    volumes:
      - ./apps:/opt/spark-apps
      - ./data:/opt/spark-data
networks:
  default:
    external: true
    name: bigdata
```

Monitoring worker nodes và spark jobs

The screenshot shows the Spark Master web interface. At the top, there's a navigation bar with links like Apps, Algorithm and..., Colab, Discrete Math, Ielts, Other, Tal Lieu Hoc Tap, AI, IT Tips, Hadoop/Spark, ScrapeWeb, Docker, and ML Deployment. Below this, the main header displays the Spark logo and the URL: spark://1be97d8f228b:7077. The status section indicates: URL: spark://1be97d8f228b:7077, Alive Workers: 2, Cores in use: 4 Total, 0 Used, Memory in use: 6.0 GiB Total, 0.0 B Used, Resources in use, Applications: 0 Running, 5 Completed, Drivers: 0 Running, 0 Completed, and Status: ALIVE. A section titled 'Workers (2)' contains a table with columns: Worker Id, Address, State, Cores, Memory, and Resources. It lists two workers, both in ALIVE state with 2 cores and 3.0 GiB memory. Another section titled 'Running Applications (0)' has a table with columns: Application ID, Name, Cores, Memory per Executor, Resources Per Executor, Submitted Time, User, State, and Duration. Finally, a section titled 'Completed Applications (5)' has a similar table showing three completed applications of type CleanData.

Spark Master at spark://1be97d8f228b:7077

URL: spark://1be97d8f228b:7077
Alive Workers: 2
Cores in use: 4 Total, 0 Used
Memory in use: 6.0 GiB Total, 0.0 B Used
Resources in use:
Applications: 0 Running, 5 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers (2)

Worker Id	Address	State	Cores	Memory	Resources
worker-20220105180359-172.22.0.12-7000	172.22.0.12:7000	ALIVE	2 (0 Used)	3.0 GiB (0.0 B Used)	
worker-20220105180400-172.22.0.13-7000	172.22.0.13:7000	ALIVE	2 (0 Used)	3.0 GiB (0.0 B Used)	

Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	------------------------	----------------	------	-------	----------

Completed Applications (5)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
app-202201052000112-0004	CleanData	4	1024.0 MiB		2022/01/05 20:01:12	root	FINISHED	12 min
app-20220105194139-0003	CleanData	4	1024.0 MiB		2022/01/05 19:41:39	root	FINISHED	12 min
app-20220105193807-0002	CleanData	4	1024.0 MiB		2022/01/05 19:38:07	root	FINISHED	1.3 min

3.6 Sử dụng spark để clean dữ liệu

Sử dụng spark để trích xuất dữ liệu thành các trường:

- **Url,**
- **Title,**
- **Realestate_Type,**
- **Area (m2),**
- **Price (B),**
- **m/m2,**
- **Location,**
- **District,**
- **Province**

Realestate_Type có thể là: *nha-mat-tien, nha-trong-hem, biet-thu-nha-lien-ke, can-ho-chung-cu, phong-tro-nha-tro, van-phong, kho-xuong, nha-hang-khach-san, shop-kiot-quan, trang-trai, mat-bang, dat-tho-cu-dat-o, dat-nen, dat-nong-lam-nghiep, cac-loai-khac*

Chương trình clean dữ liệu:

```
cleandata.py
1 import re
2 import sys
3 from bs4 import BeautifulSoup
4 from pyspark.sql import SparkSession
5
6
7 def extract_info(body: str):
8     soup = BeautifulSoup(body, "lxml")
9     title = soup.title.text.replace('\n', '').replace('\t', '')
10
11     location = soup.find('div', {'class': 'address'}).find('span', class_='value').text.strip()
12     raw_price = soup.find('span', class_='price').find('span', class_='value').text.strip().lower()
13     words = raw_price.replace(',', '.').split()
14     price = None
15     if 'tỷ' in words:
16         try:
17             price = float(words[0])
18         except:
19             pass
20     elif 'triệu' in words:
21         try:
22             price = float(words[0]) / 1000
23         except:
24             pass
25
26     raw_area = soup.find('span', class_='square').find('span', class_='value').text.strip().lower()
27     words = raw_area.replace(',', '.').split()
28     area = None
29     try:
30         area = float(words[0])
31     except:
32         pass
```

```

33
34     return price, area, title, location
35
36
37 def clean_data(x):
38     body = x.body
39     unit = None
40     price = None
41     area = None
42     title = None
43     location = None
44     url = x.url
45     district = x.district
46     province = x.province
47     realestate_type = x.realestate_type
48     if isinstance(body, str):
49         try:
50             price, area, title, location = extract_info(body)
51
52             if price is not None and area is not None:
53                 unit = (price / area) * 1000
54         except:
55             pass
56     return (url, title, realestate_type, area, price, unit, location, district, province)
57
58
59 spark = SparkSession.builder\
60     .appName('CleanData')\
61     .config("spark.mongodb.output.uri", "mongodb+srv://manhtruong:123456aA@cloudcluster.kq0xr.mongodb.net/myfirstData")\
62     .getOrCreate()
63
64 columns = ["Url", "Title", "Type", "Area (m2)", "Price (B)", "m/m2", "Location", "District", "Province"]
65 raw_df = spark.read.json("hdfs://namenode:9000/alonhadatnews_json")
66 rdd_raw = raw_df.rdd
67
68 cleaned_rdd = rdd_raw.map(lambda x: clean_data(x))
69 cleaned_df = cleaned_rdd.toDF(columns)
70 # cleaned_df.write.mode('overwrite').csv('/opt/spark-data/cleaned_data.csv')
71 cleaned_df.write.format("mongo").mode("append").save()
72

```

Vào spark master node chạy spark-submit với câu lệnh

```

"/opt/spark/bin/spark-submit --master spark://spark-master:7077 --archives /opt/spark-data/bigdata_venv.tar.gz#enviroment /opt/spark-data/cleandata.py
"

```

4. Kết quả đạt được

- **Crawling:**

- Nhóm đã crawl được **48582** tin đăng bán bất động sản

```


Using Python version 3.7.3 (default, Jan 22 2021 20:04:44)
SparkSession available as 'spark'.
>>> raw_df = spark.read.json("hdfs://namenode:9000/alonhadatnews_json")
22/01/06 04:08:24 WARN SharedInMemoryCache: Evicting cached table partition metadata from memory due to
leCacheSize = 262144000 bytes). This may impact query planning performance.
>>> raw_df.count()
48582
>>>

```

- Tổng lượng dữ liệu lưu trong HDFS ~ **7GB**

Configured Capacity:	468.44 GB
Configured Remote Capacity:	0 B
DFS Used:	7.28 GB (1.55%)
Non DFS Used:	322.36 GB
DFS Remaining:	118.82 GB (25.36%)
Block Pool Used:	7.28 GB (1.55%)
DataNodes usages% (Min/Median/Max/stdDev):	1.55% / 1.55% / 1.55% / 0.00%
Live Nodes	1 (Decommissioned: 0, In Maintenance: 0)
Dead Nodes	0 (Decommissioned: 0, In Maintenance: 0)
Decommissioning Nodes	0
Entering Maintenance Nodes	0
Total Datanode Volume Failures	0 (0 B)
Number of Under-Replicated Blocks	67861
Number of Blocks Pending Deletion (including replicas)	0

- **Spark job**
- Chương trình “CleanData” Spark job hoàn thành


Spark Master at spark://1be97d8f228b:7077
Quit

URL: spark://1be97d8f228b:7077
 Alive Workers: 2
 Cores in use: 4 Total, 0 Used
 Memory in use: 6.0 GiB Total, 0.0 B Used
 Resources in use:
 Applications: 0 Running, 5 Completed
 Drivers: 0 Running, 0 Completed
 Status: ALIVE

▼ Workers (2)

Worker Id	Address	State	Cores	Memory	Resources
worker-20220105180359-172.22.0.12-7000	172.22.0.12:7000	ALIVE	2 (0 Used)	3.0 GiB (0.0 B Used)	
worker-20220105180400-172.22.0.13-7000	172.22.0.13:7000	ALIVE	2 (0 Used)	3.0 GiB (0.0 B Used)	



























▼ Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	------------------------	----------------	------	-------	----------

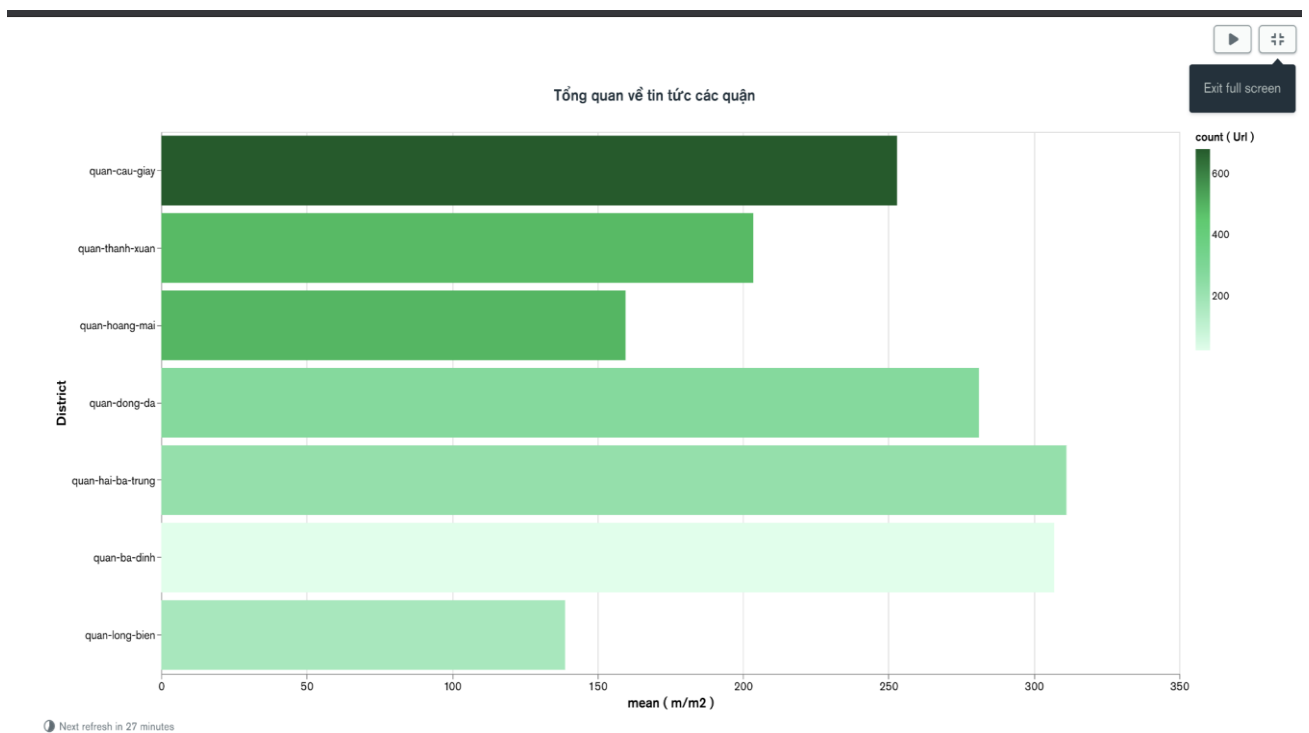
▼ Completed Applications (5)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
app-20220105200112-0004	CleanData	4	1024.0 MiB		2022/01/05 20:01:12	root	FINISHED	12 min
app-20220105194139-0003	CleanData	4	1024.0 MiB		2022/01/05 19:41:39	root	FINISHED	12 min
app-20220105193807-0002	CleanData	4	1024.0 MiB		2022/01/05 19:38:07	root	FINISHED	1.3 min

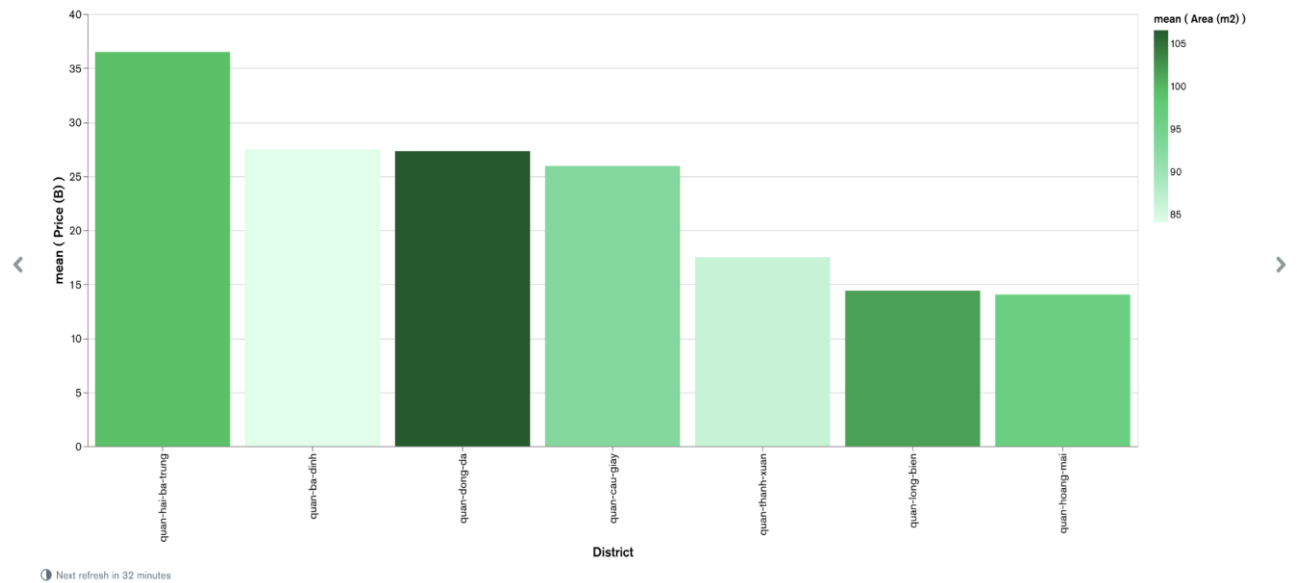
● Dữ liệu được đẩy vào MongoDB

# cleaned_data										
	_id	objectid	url String	Title String	Type String	Area (m2) Double	Price (₫) Double	m/m2 Double	Location String	
1	61d06c2afc7724b04c568d2f		"https://alonhadat.com.vn/bai	"Bán ""GẤP"" nhà Phố Vương Ti	"nha-mat-tien"	51	5.7	111.76476588235294	"Thanh Xuân Phố	  
2	61d06c2afc7724b04c568d39		"https://alonhadat.com.vn/bai	"Bán nhà mới Hồ Tùng Mậu 125"	"nha-mat-tien"	125	17.5	140	"Đường Mai Dịch,	  
3	61d06c2afc7724b04c568d31		"https://alonhadat.com.vn/bai	"Bán nhà phân lô 147 Tân Mai,	"nha-mat-tien"	259	5.8	23.2	"259 Phố Vọng, 2	  
4	61d06c2afc7724b04c568d32		"https://alonhadat.com.vn/hu	"HIỆM TÒA VP NP HOÀNG CẦU VỊ	"nha-mat-tien"	96	30	312.5	"Phố Hoàng Cầu,	  
5	61d06c2afc7724b04c568d33		"https://alonhadat.com.vn/ma	"NP VĨA HÉ RỘNG 10M TRUNG KÌ	"nha-mat-tien"	75	26	346.6666666666667	"Phố Trung Kinh,	  
6	61d06c2afc7724b04c568d34		"https://alonhadat.com.vn/au	"ẢNH THẬT Bán nhà 521 Trương	"nha-mat-tien"	259	5.8	23.2	"ngõ 24 Kim Đồng	  
7	61d06c2afc7724b04c568d35		"https://alonhadat.com.vn/bai	"Bán nhà phố Kim Đồng, Giáp l	"nha-mat-tien"	55	7.9	143.63636363636365	"Phố Kim Đồng, P	  
8	61d06c2afc7724b04c568d36		"https://alonhadat.com.vn/ik	"LK SHOPHOUSE MẶT PHỐ TÔN TH	"nha-mat-tien"	120	63	525	"Phố Trần Thái T	  
9	61d06c2afc7724b04c568d37		"https://alonhadat.com.vn/da	"ĐẮC ĐỊA HIỆM CỎ MẶT PHỐ P. P	"nha-mat-tien"	315	138	438.0952380952381	"Đường Phạm Đìn	  
10	61d06c2afc7724b04c568d38		"https://alonhadat.com.vn/bu	"BIỆT THỰ CẦU GIẤY, PHÂN LÔ,	"nha-mat-tien"	85	10.8	127.05862352941178	"Đường Hồ Tùng M	  
11	61d06c2afc7724b04c568d39		"https://alonhadat.com.vn/bai	"Bán nhà mặt phố Trần Đại Ng	"nha-mat-tien"	26	5.5	211.53846153846155	"Đường Sông Sét,	  
12	61d06c2afc7724b04c568d3a		"https://alonhadat.com.vn/bai	"Cần bán gấp nhà mặt phố kìn	"nha-mat-tien"	66	26	393.95939393939394	"Đường Trần Duy	  
13	61d06c2afc7724b04c568d3b		"https://alonhadat.com.vn/bai	"BÁN NHÀ NGÃ TƯ SỐ 50M2x5T M	"nha-mat-tien"	59	4.3	86	"Đường Nguyễn Tr	  
14	61d06c2afc7724b04c568d3c		"https://alonhadat.com.vn/su	"SIÊU GIÁ TRỊ MẶT PHỐ VÀNH Đ	"nha-mat-tien"	259	62	248	"Phố Tân Mai, Ph	  
15	61d06c2afc7724b04c568d3d		"https://alonhadat.com.vn/vi	"VỊ TRÍ VIP MẶT PHỐ TUỆ TỈNH	"nha-mat-tien"	145	69	475.8620696551727	"Phố Tuệ Tĩnh, P	  
16	61d06c2afc7724b04c568d3e		"https://alonhadat.com.vn/vi	"ĐẦU TƯ VIP MẶT PHỐ MINH KH	"nha-mat-tien"	2.015	250	124069.47898818857	"Đường Minh Khai	  
17	61d06c2afc7724b04c568d3f		"https://alonhadat.com.vn/bu	"HIỆM BÁN- NHÀ PHÂN LÔ GỖ - l	"nha-mat-tien"	105	14	133.33333333333334	"Phố Hoàng Văn T	  
18	61d06c2afc7724b04c568d40		"https://alonhadat.com.vn/bai	"Bán Nhà Phố Hoàng Quốc Việ	"nha-mat-tien"	70	15	214.28571428571428	"Đường Hoàng Qu	  
19	61d06c2afc7724b04c568d41		"https://alonhadat.com.vn/bai	"Bán nhà 48m2x5T Tân Mai, H	"nha-mat-tien"	259	5.8	23.2	"259 Phố Vọng, 2	  
20	61d06c2afc7724b04c568d42		"https://alonhadat.com.vn/nh	"NHÀ MẶT PHỐ ÔTÔ ĐỒ CỬA-KINH	"nha-mat-tien"	67	12	179.1044776119403	"2.9 . Phố Vọng,	  

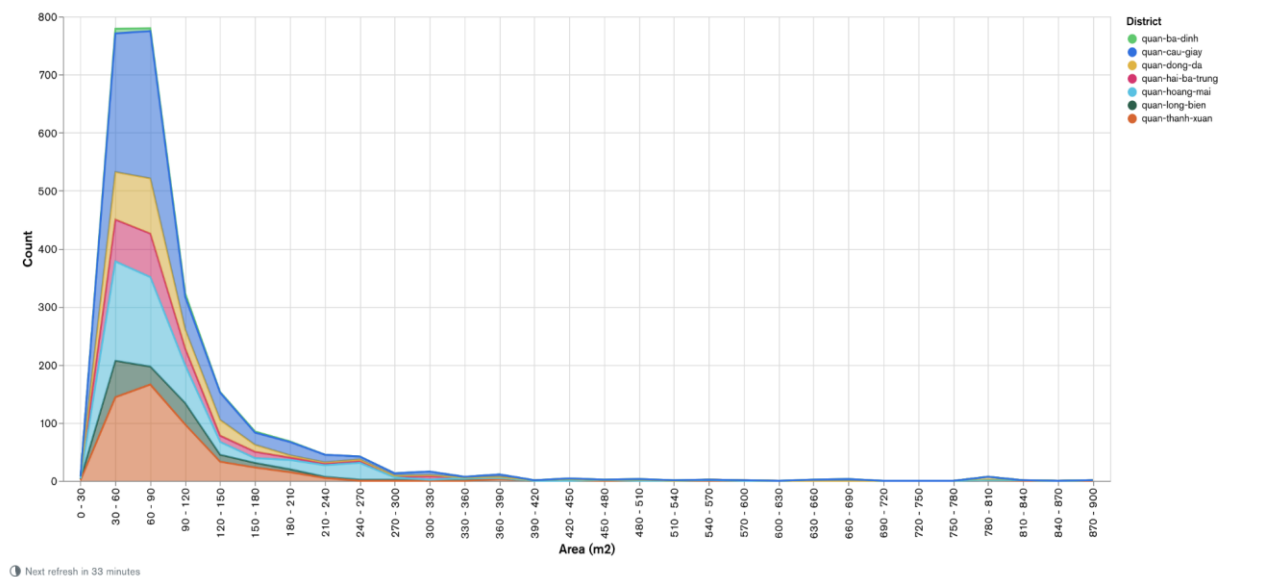
5. Visulize dữ liệu và phân tích



Giá nhà đất và diện tích trung bình khu vực



Bảng phân bố diện tích theo khu vực



- Nhìn vào biểu đồ trên ta có thể dễ dàng thấy số lượng các tin bán nhà mặt phố tại Quận Cầu Giấy vượt trội với khoảng gần 700 tin, tiếp theo sau là quận Thanh Xuân, Hoàng Mai và ít tin nhất với khoảng 20 tin là quận Ba Đình
- Quận Ba Đình là quận đắt đỏ nhất khi dựa vào biểu đồ 2 có thể dễ dàng thấy diện tích tuy nhỏ nhưng giá lại khá cao, ngang ngửa Quận Hai Bà Trưng nhưng lại bé hơn nhiều
- Các Bất động sản lớn với giá cao hiện đang tập trung tại Quận Hai Bà Trưng.
- Nhìn vào biểu đồ cuối, có thể thấy diện tích thường cho một nhà phổ biến vào khoản 30-150 m2.

6. Tài liệu tham khảo

[1]. Slide môn học Lưu trữ và xử lý dữ liệu lớn TS.Đào Thành Chung, Đại Học Bách Khoa Hà Nội

<https://scholar.google.com.vn/citations?user=bGBBNocAAAAJ&hl=en>

[2]. Slide môn học “Lưu trữ và xử lý dữ liệu lớn” TS.Trần Việt Trung, Đại Học Bách Khoa Hà Nội

<https://scholar.google.com/citations?user=wYWRXQ0AAAAJ&hl=en>

[3]. <https://viblo.asia/p/kafka-la-gi-gDVK2Q7A5Lj>

[4]. <https://wiki.tino.org/kafka-la-gi/>

[5]. <https://hocspringboot.net/2021/08/29/apache-kafka-la-gi-cac-khai-niem-co-ban/>

[6]. <https://kafka.apache.org/>

[7]. <https://spark.apache.org/>

[8]. <https://viblo.asia/p/tim-hieu-ve-apache-spark-ByEZkQQW5Q0>

[9]. <https://bizfly.vn/techblog/apache-spark-la-gi.html>

[10]. <https://docs.mongodb.com/>