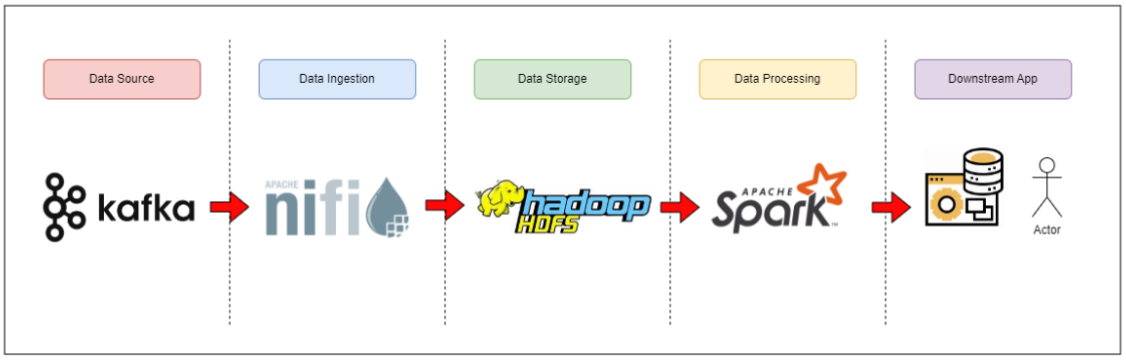
AssignmentDE

- Kiến trúc hệ thống



- Kafka gồm 2 broker và được quản lý bởi zookeeper
- Nifi bao gồm NiFi registry và Nifi được quản lý bởi zookeeper
- HDFS gồm namenode, 2 datanode cùng với nodemanager và resourcemanager phục vụ cho YARN
- Spark gồm 1 master node, 2 worker node và 1 spark-submit

▼ Cách thức triển khai, thực hiện các bước

- Cấu hình các container như trong docker-compose.yml
- Các dependency cần thiết có trong pom.xml

▼ Set up phần Nifi

Ban đầu chúng ta chưa mount thư mục conf vào Nifi

```
      - ./nifi/provenance_repository:/opt/nifi/nifi-current/provenance_repository
      - ./nifi/state:/opt/nifi/nifi-current/state
      - ./nifi/logs:/opt/nifi/nifi-current/logs
      # uncomment the next line after copying the /conf directory from the container to your local directory
      #- ./nifi/conf:/opt/nifi/nifi-current/conf
networks:
      - persistent_network
```

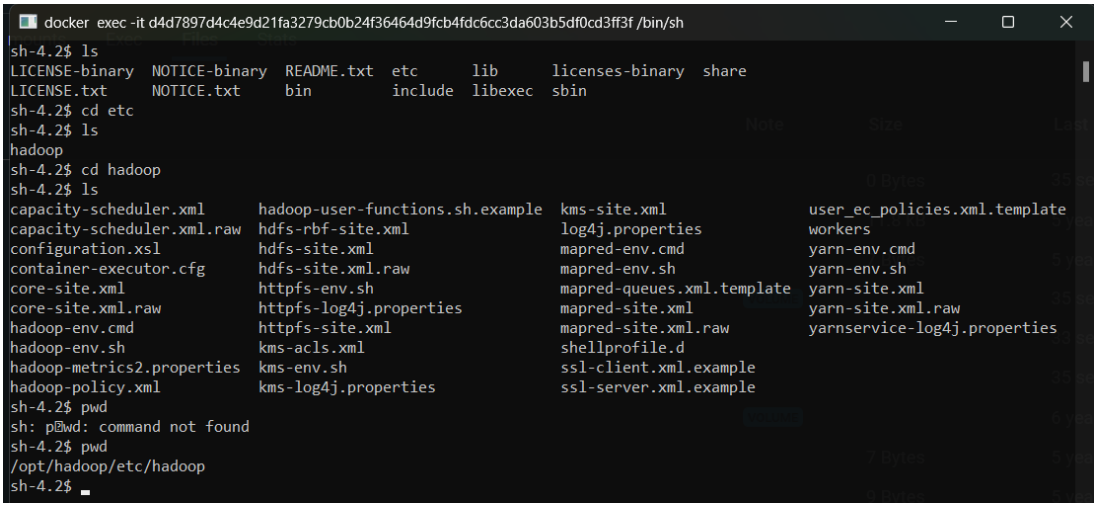
Sau khi docker compose up thì các thư mục chưa tồn tại sẽ được tạo ra trên local và chúng ta phải tìm ID của container Nifi để copy thư mục conf từ Nifi về local

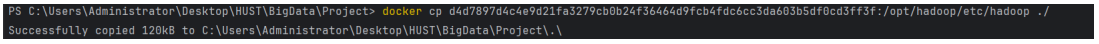
```
# to get the container ID of NiFi's docker container
> docker ps

# the result will look like this (shortened to fit this article)
CONTAINER ID   IMAGE                          COMMAND
7554d9c68c8f   apache/nifi:1.14.0             ...
8af04cd37e06   apache/nifi-registry:1.15.0    ...
a2dacb43ed23   bitnami/zookeeper:3.7.0        ...

# copy the directory from the docker container to the local machine
> docker cp 7554d9c68c8f:/opt/nifi/nifi-current/conf ./nifi/
```

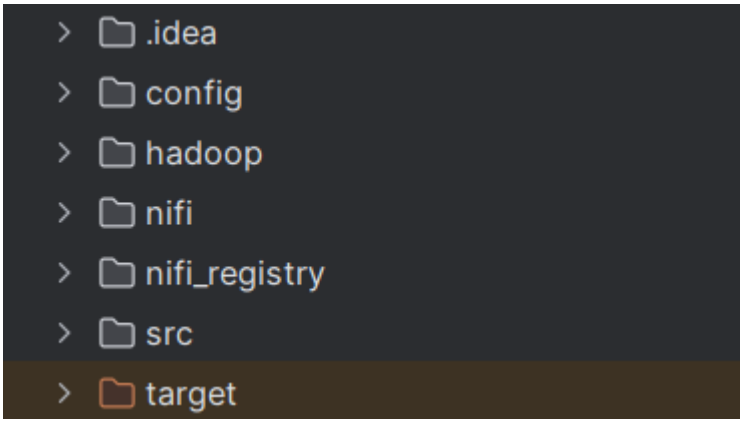
Copy thành công thì lần compose sau chúng ta sẽ bỏ comment câu lệnh

```
- ./nifi/conf:/opt/nifi/nifi-current/conf
```

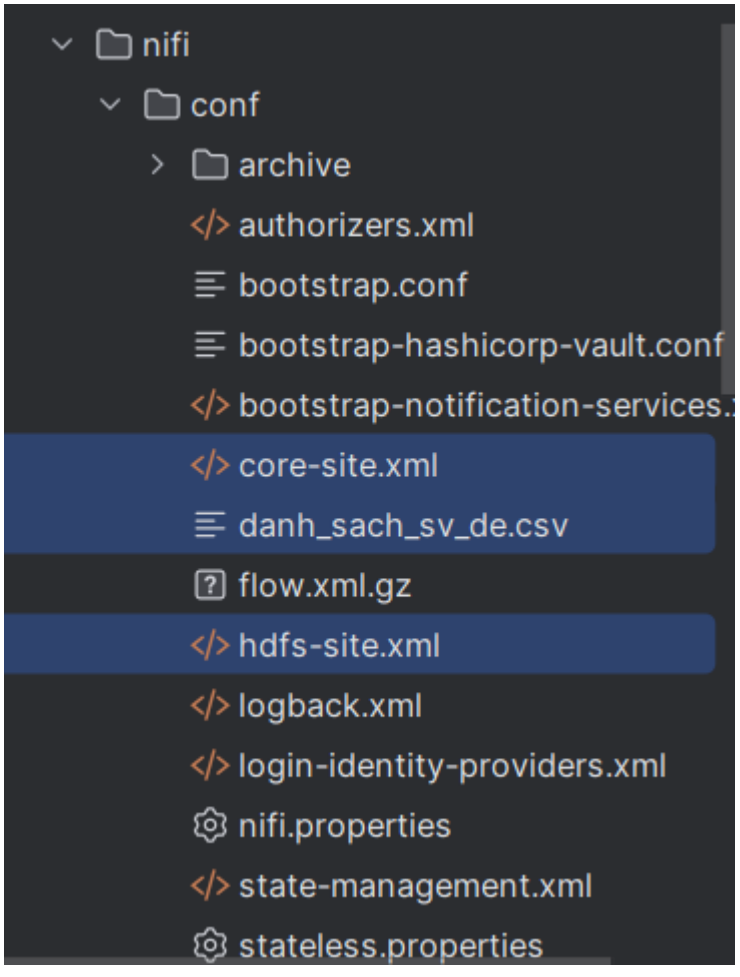
Chúng ta sẽ mount thư mục các file cấu hình của hdfs ra local







Copy 2 file core-site.xml và hdfs-site.xml tới ./nifi/conf để phục vụ cho việc truyền file từ Nifi vào HDFS. Copy file danh_sach_sv_de.csv tới ./nifi/conf để phục vụ truyền file sau này.



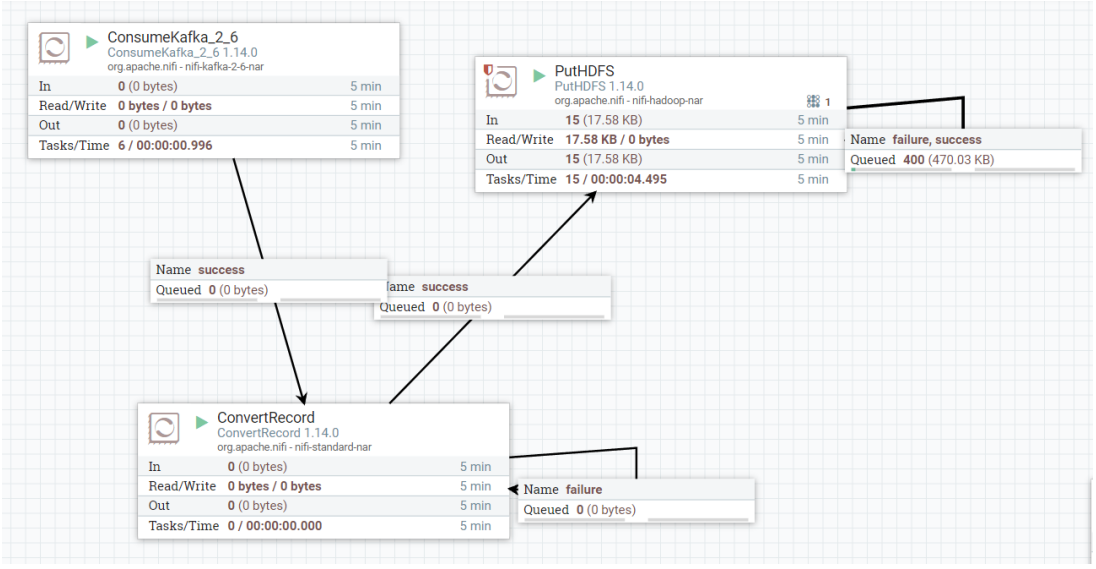
Đến đây việc chuẩn bị cho Nifi đã hoàn thành.

▼ Producer của Kafka

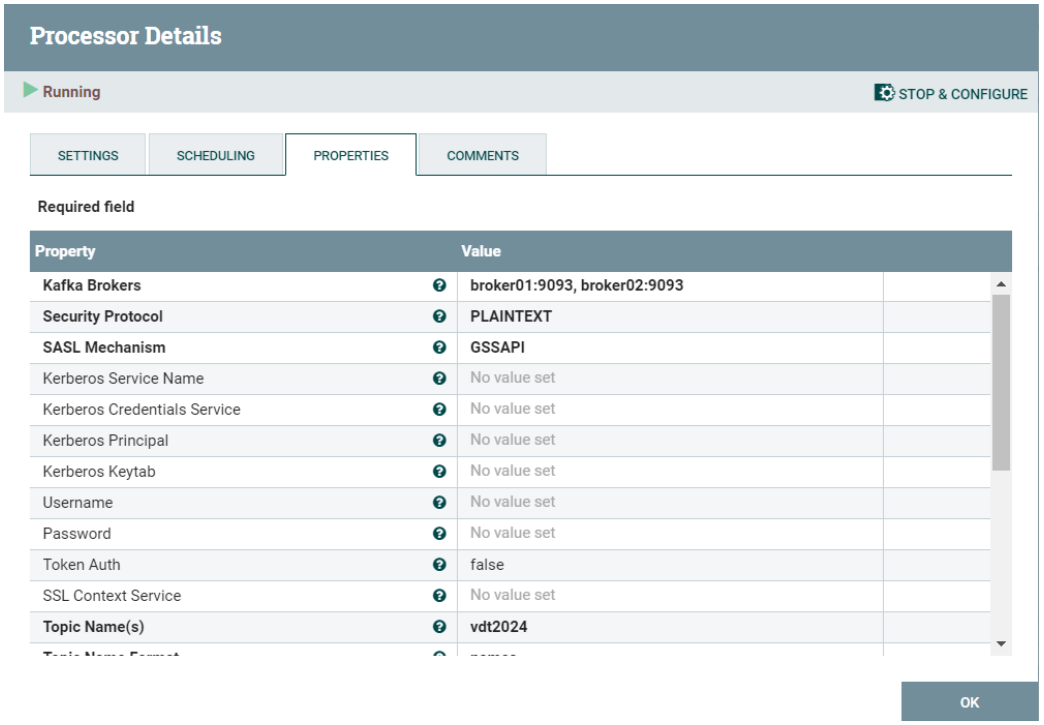
```java
public static void main(String[] args) {
    Properties props = new Properties();
    props.put(ProducerConfig.BOOTSTRAP_SERVERS_CONFIG, BOOTSTRAP_SERVERS);
    props.put(ProducerConfig.KEY_SERIALIZER_CLASS_CONFIG, "org.apache.kafka.common.serialization.StringSerializer");
    props.put(ProducerConfig.VALUE_SERIALIZER_CLASS_CONFIG, "org.apache.kafka.common.serialization.StringSerializer");

    ObjectMapper objectMapper = new ObjectMapper();

    try (KafkaProducer<String, String> producer = new KafkaProducer<>(props);
         BufferedReader reader = new BufferedReader(new FileReader(logPath))) {
        String line;
        int recordCount = 0;

        while ((line = reader.readLine()) != null) {
            // Process the CSV line and create a JSON string
            String[] fields = line.split( regex: ",");
            Map<String, Object> jsonMap = new HashMap<>();
            jsonMap.put( k: "student_code", Integer.parseInt(fields[0]));
            jsonMap.put( k: "activity", fields[1]);
            jsonMap.put( k: "numberOfFile", Integer.parseInt(fields[2]));
            jsonMap.put( k: "timestamp", fields[3]);

            String jsonString = objectMapper.writeValueAsString(jsonMap);
            String key = getKey(fields); // Control partitions of records
            ProducerRecord<String, String> record = new ProducerRecord<>(TOPIC_NAME, key, jsonString);
```

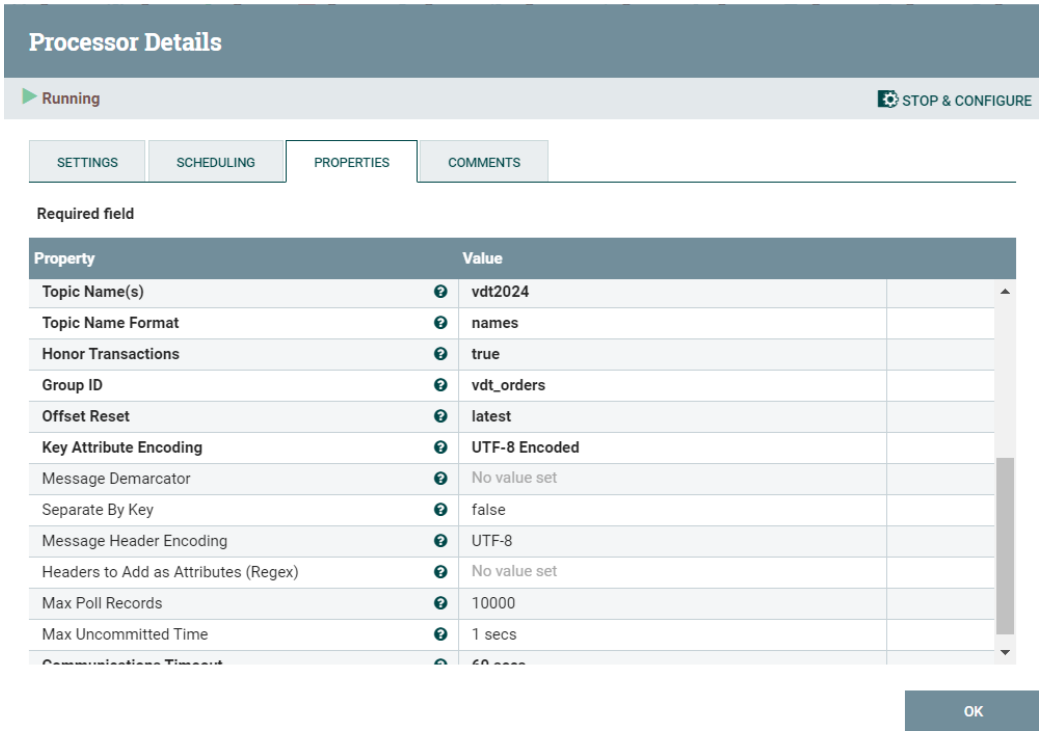
Sau khi broker Kafka hoạt động ổn định chạy Producer sẽ đẩy dữ liệu lên Kafka broker

▼ Triển khai phần Nifi để nhận dữ liệu từ kafka, convert record sang parquet và truyền file parquet tới HDFS để lưu trữ ở đường dẫn “/raw_zone/fact/activity”.

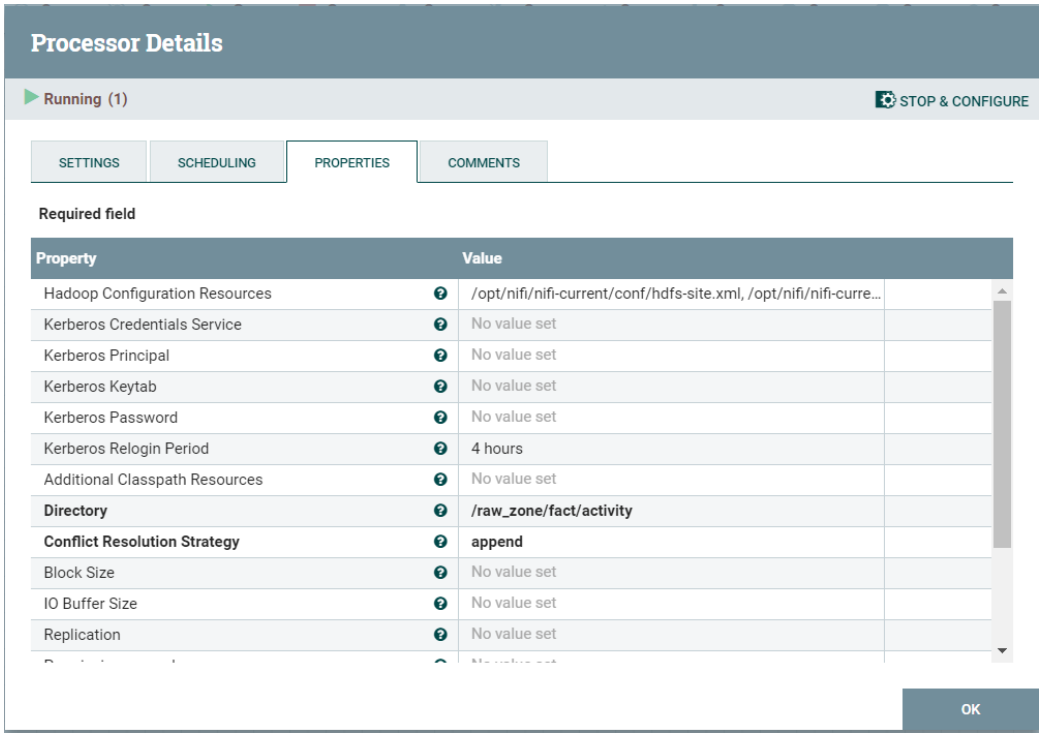


- Cấu hình ConsumeKafka

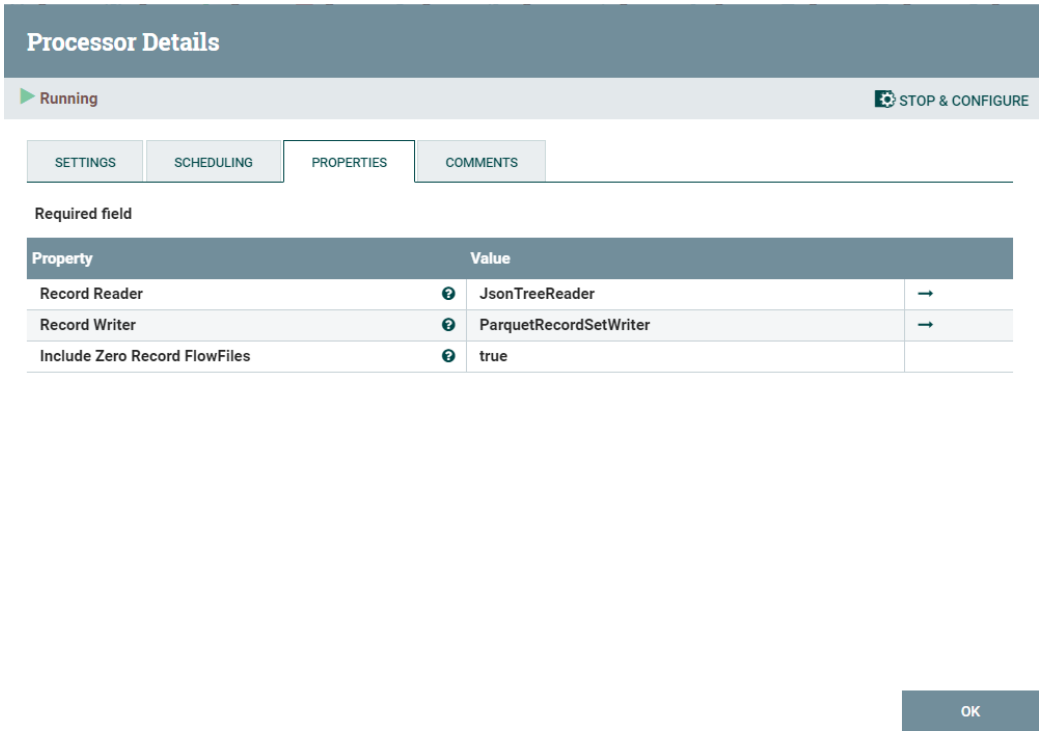




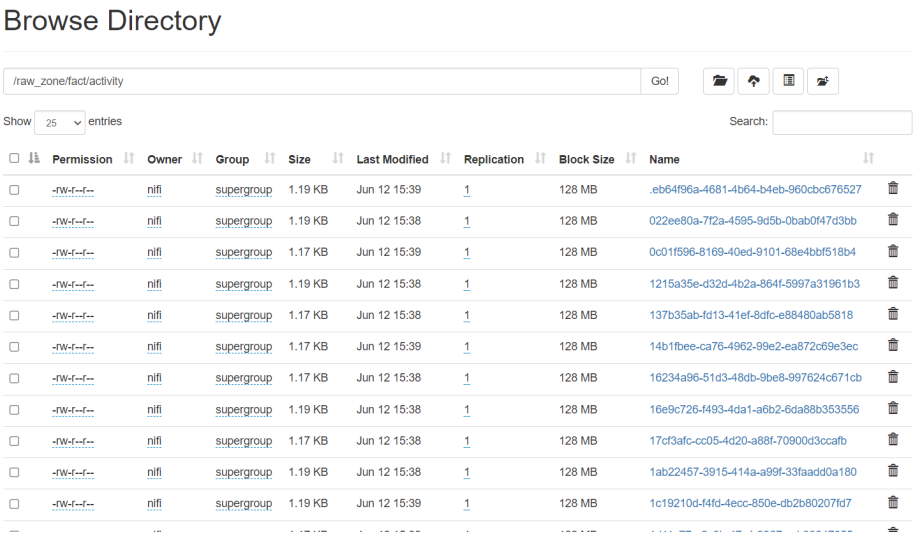
- Cấu hình PutHDFS



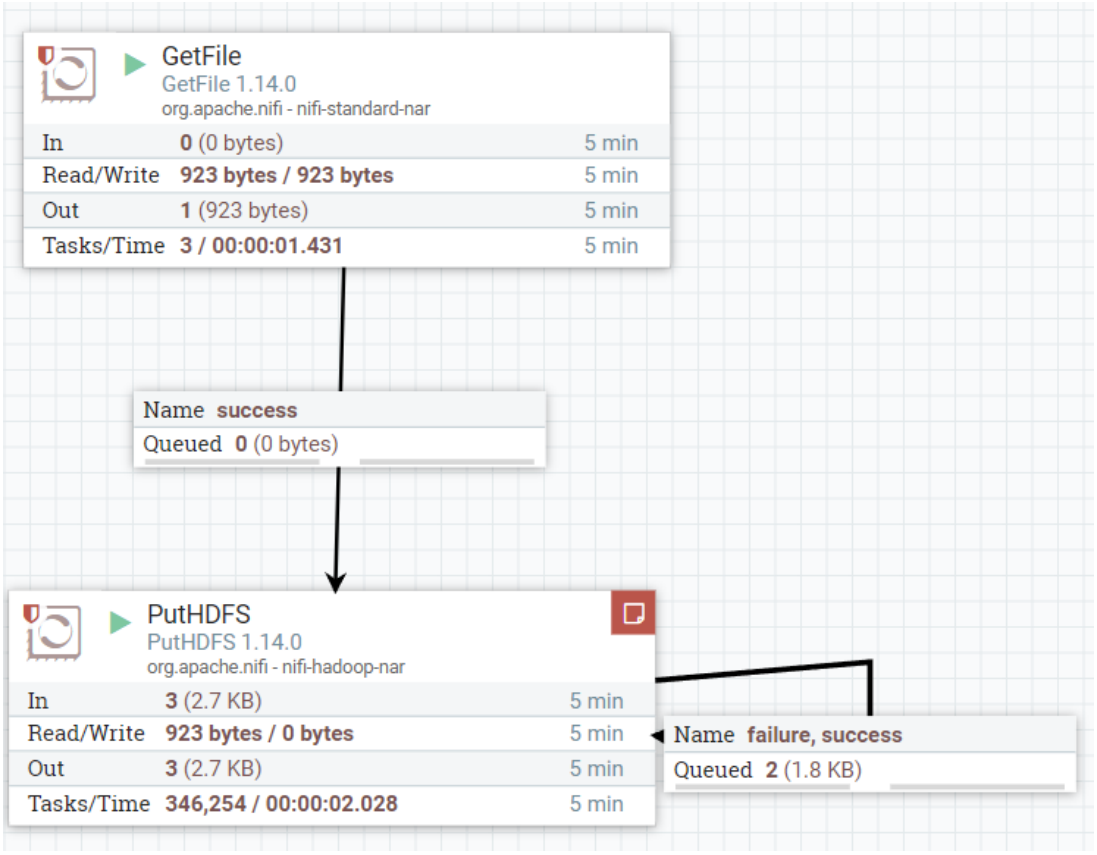
- Cấu hình ConvertRecord



Kết quả sau khi thực hiện

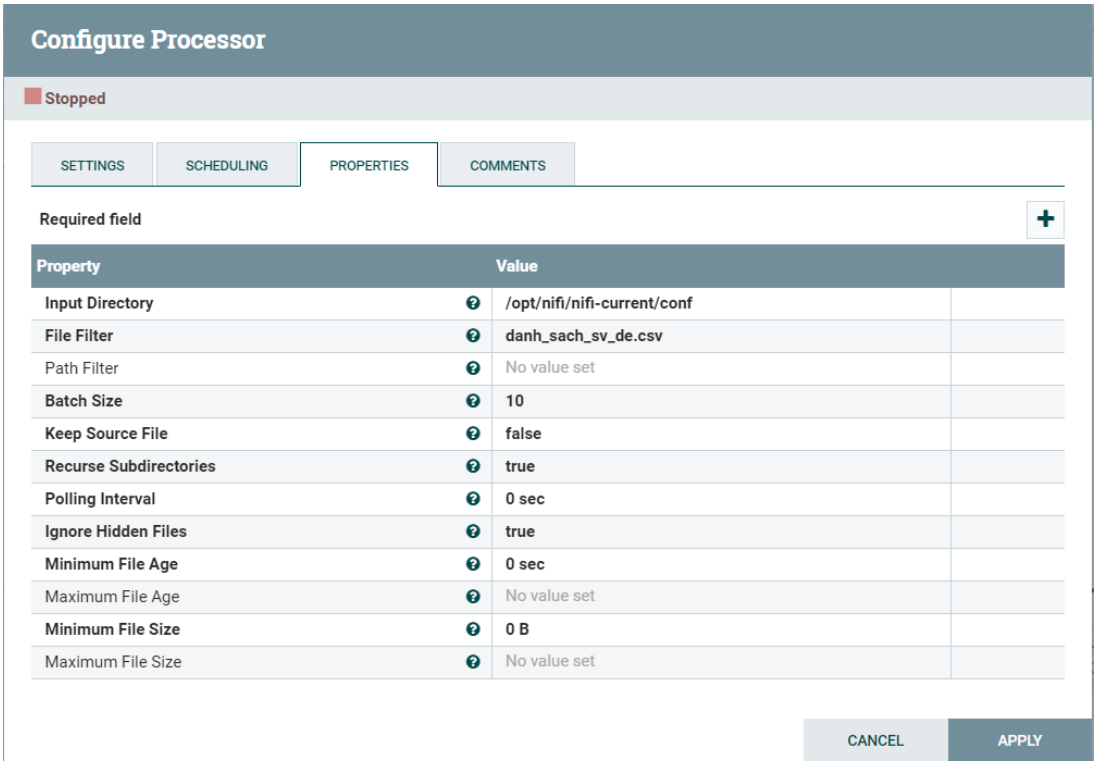


▼ Triển khai Nifi truyền file “danh_sach_sv_de.csv” đến HDFS

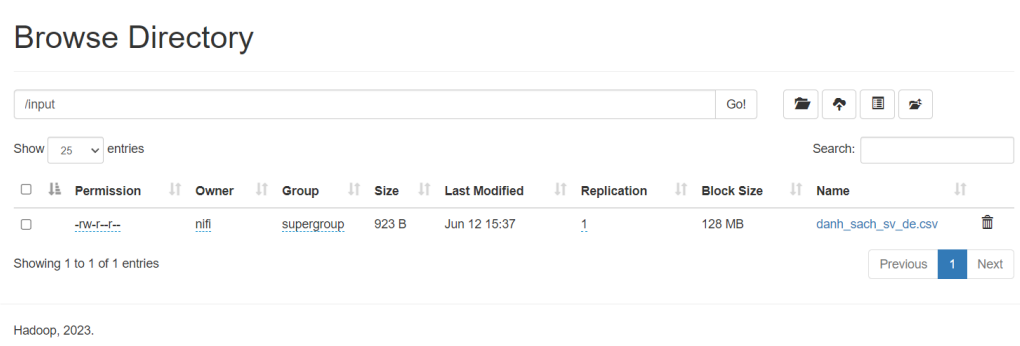


Cấu hình PutHDFS như trên.

Cấu hình GetFile



Kết quả sau khi thực hiện

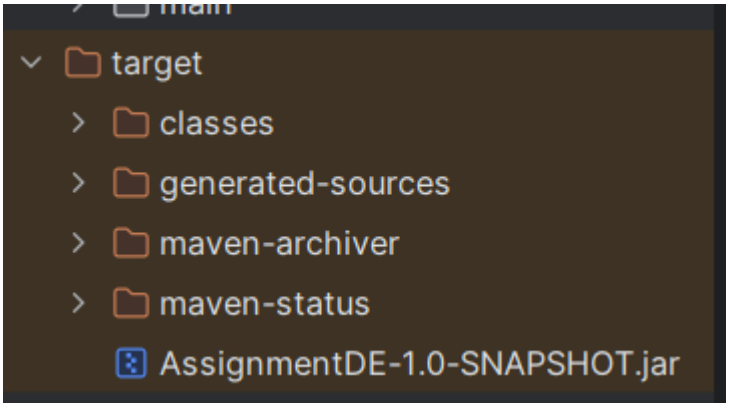


▼ Apache Spark

Chương trình để đọc danh_sach_sv_de.csv chuyển thành dataframe và đọc lần lượt các file parquet để tạo thành dataframe. Từ đó xử lí dữ liệu và xuất output vào HDFS.

```java
SparkSession spark = SparkSession.builder()
        .appName("Read Parquet from HDFS and Aggregate Activities")
        .config("spark.master", "local")
        .getOrCreate();

String studentFilePath = "hdfs://namenode:8020/input/danh_sach_sv_de.csv";

StructType schema = new StructType(new StructField[]{
        DataTypes.createStructField( name: "student_code", DataTypes.IntegerType,  nullable: false),
        DataTypes.createStructField( name: "student_name", DataTypes.StringType,  nullable: false)
});

Dataset<Row> studentDf = spark.read()
        .format( source: "csv")
        .schema(schema)
        .option("header", "false")
        .load(studentFilePath);

String hdfsPath = "hdfs://namenode:8020/raw_zone/fact/activity";
Dataset<Row> df = spark.read().parquet(hdfsPath);

Dataset<Row> aggregatedDF = df.groupBy( col1: "student_code",  ...cols: "activity", "timestamp")
        .agg(sum( columnName: "numberOfFile").as( alias: "totalFile"));

Dataset<Row> outputDF = aggregatedDF.join(studentDf, usingColumn: "student_code")
        .select( col: "timestamp",  ...cols: "student_code", "student_name", "activity", "totalFile")
        .orderBy( sortCol: "student_code",  ...sortCols: "activity");

String outputHdfsPath = "hdfs://namenode:8020/output/aggregated_activities";
outputDF.write()
        .mode( saveMode: "overwrite")
        .option("header", "false")
```

Sau đó chạy lệnh mvn package trong terminal để xuất ra file .jar

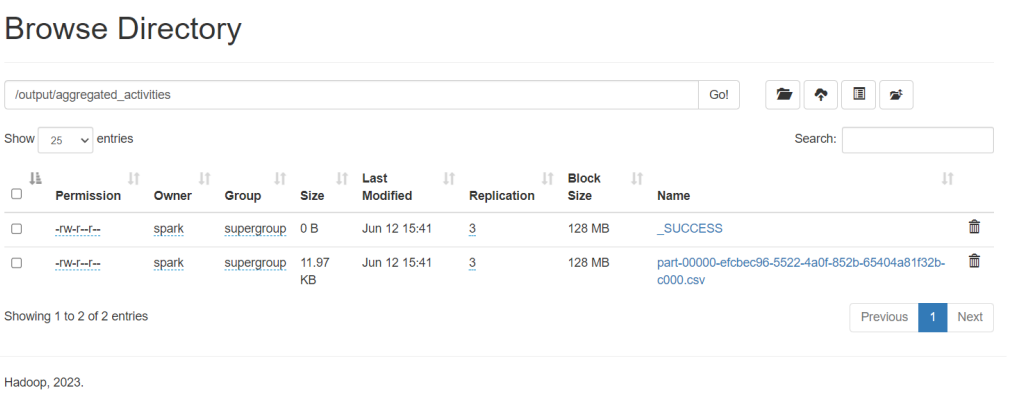


Cấu hình spark-submit để khi khởi động sẽ chạy chương trình thực thi quá trình chuyển đổi và đưa ra kết quả

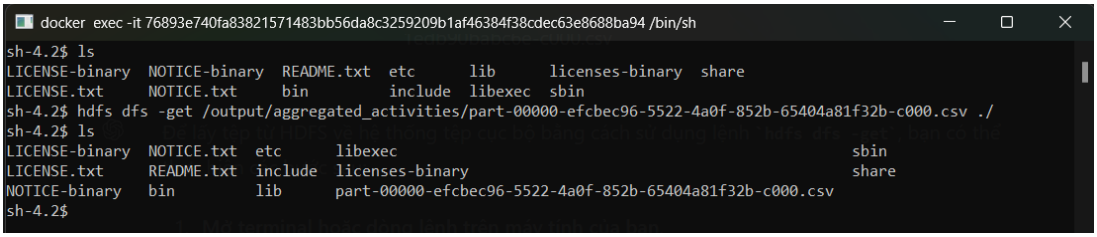
```yaml
spark-submit-job:
  image: bitnami/spark:latest
  container_name: spark-submit-job
  depends_on:
    - spark-master
    - spark-worker-1
    - spark-worker-2
  command: bash -c "sleep 30 && spark-submit --class org.spark.ParquetReader --master spark://spark-master:7077 /app/AssignmentDE-1.0-SNAPSHOT.jar"
  volumes:
    - ./target/AssignmentDE-1.0-SNAPSHOT.jar:/app/AssignmentDE-1.0-SNAPSHOT.jar
  networks:
    - my_persistent_network
```

Do không phải spark streaming nên phải đợi Nifi truyền hết dữ liệu vào HDFS thì mới khởi động container spark-submit để xử lí.

Kết quả sau khi xử lí được lưu vào HDFS

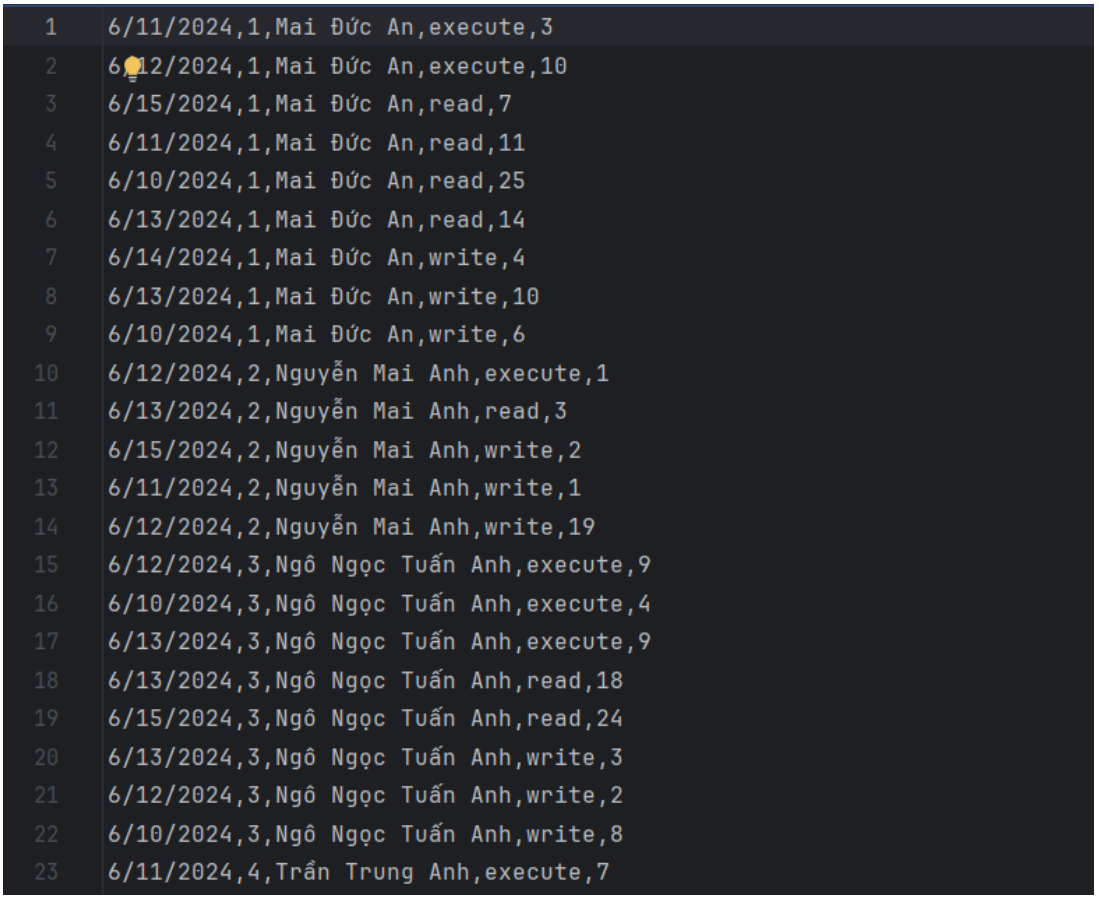


Copy file .csv về container



Copy file từ container về local

Một số giá trị kết quả



```
6/11/2024,1,Mai Đức An,execute,3
6/12/2024,1,Mai Đức An,execute,10
6/15/2024,1,Mai Đức An,read,7
6/11/2024,1,Mai Đức An,read,11
6/10/2024,1,Mai Đức An,read,25
6/13/2024,1,Mai Đức An,read,14
6/14/2024,1,Mai Đức An,write,4
6/13/2024,1,Mai Đức An,write,10
6/10/2024,1,Mai Đức An,write,6
6/12/2024,2,Nguyễn Mai Anh,execute,1
6/13/2024,2,Nguyễn Mai Anh,read,3
6/15/2024,2,Nguyễn Mai Anh,write,2
6/11/2024,2,Nguyễn Mai Anh,write,1
6/12/2024,2,Nguyễn Mai Anh,write,19
6/12/2024,3,Ngô Ngọc Tuấn Anh,execute,9
6/10/2024,3,Ngô Ngọc Tuấn Anh,execute,4
6/13/2024,3,Ngô Ngọc Tuấn Anh,execute,9
6/13/2024,3,Ngô Ngọc Tuấn Anh,read,18
6/15/2024,3,Ngô Ngọc Tuấn Anh,read,24
6/13/2024,3,Ngô Ngọc Tuấn Anh,write,3
6/12/2024,3,Ngô Ngọc Tuấn Anh,write,2
6/10/2024,3,Ngô Ngọc Tuấn Anh,write,8
6/11/2024,4,Trần Trung Anh,execute,7
```

Link source code github:

https://github.com/PJavis/AssignmentDE.git