Shape

Description automatically generated

**TRƯỜNG ĐẠI HỌC PHENIKAA**

**KHOA CÔNG NGHỆ THÔNG TIN**

**------------oOo-----------**

Logo, company name

Description automatically generated

**BÀI THI KẾT THÚC HỌC PHẦN MÔN**

**LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG**

**Đề tài : Xây dựng một ứng dụng phân tích dữ liệu sử dụng Apache Hadoop**

**Lớp: Lập trình hướng đối tượng(N03)**

**Nhóm : 14**

Giảng viên hướng dẫn : TS.Trần Đăng Hoan

Khoa chuyên môn : Khoa Công Nghệ Thông Tin

Thành viên :

Nguyễn Minh Trường Giang - 21011597

Hoàng Anh Tuấn - 21012902

Đỗ Đình Quý - 21011627

HÀ NỘI, 4/2023

**Bảng phân công công việc**

|  |  |  |
| --- | --- | --- |
| Họ Tên | Mã Sinh Viên | Chi tiết công việc |
| Nguyễn Minh Trường Giang | 21011597 | Cài đặt Hadoop và Maven, Thiết kế ứng dụng, triển khai trên local |
| Hoàng Anh Tuấn | 21012902 | Tìm hiểu Big Data, Apache Hadoop |
| Đỗ Đình Quý | 21011627 | Thiết kế ứng dụng, triển khai trên local |

MỤC LỤC

**Contents**

[MỤC LỤC 1](#_Toc133323103)

[LỜI MỞ ĐẦU 1](#_Toc133323104)

[NỘI DUNG 2](#_Toc133323105)

[**1. GIỚI THIỆU** 2](#_Toc133323106)

[1.1 Giới thiệu chung về Big Data 2](#_Toc133323107)

[1.2 Giới thiệu về Apache Hadoop 2](#_Toc133323108)

[1.3 Mục tiêu đề tài 6](#_Toc133323109)

[1.4 Hướng giải quyết 6](#_Toc133323110)

[1.5 Khó khăn và thách thức 6](#_Toc133323111)

[1.6 Hướng giải quyết 7](#_Toc133323112)

[**2. NỀN TẢNG CÔNG NGHỆ** 7](#_Toc133323113)

[2.1 Các thành phần trong Hadoop 7](#_Toc133323114)

[2.2 Các công nghệ quan trọng 8](#_Toc133323115)

[2.3 Giải thuật sử dụng 9](#_Toc133323116)

[**3. THIẾT KẾ ỨNG DỤNG** 11](#_Toc133323117)

[3.1 Mô tả yêu cầu 11](#_Toc133323118)

[3.2 Tính năng chính 11](#_Toc133323119)

[3.3 Thiết kế kiến trúc của ứng dụng 12](#_Toc133323120)

[3.4 Mô tả luồng xử lý dữ liệu 12](#_Toc133323121)

[3.5 Xây dựng một số hàm phân tích dữ liệu trên Hadoop 13](#_Toc133323122)

[**4. TRIỂN KHAI ỨNG DỤNG** 14](#_Toc133323123)

[4.1 Cài đặt Hadoop 14](#_Toc133323124)

[4.2 Xây dựng môi trường phát triển 16](#_Toc133323125)

[4.3 Triển khai ứng dụng trên Hadoop 20](#_Toc133323126)

[4.4 Thiết kế kiểm thử 25](#_Toc133323127)

[4.4.1 Các ca kiểm thử (Test Cases) 25](#_Toc133323128)

[4.4.2 Tích hợp kiểm thử. 26](#_Toc133323129)

[KẾT LUẬN 28](#_Toc133323130)

[TÀI LIỆU THAM KHẢO 29](#_Toc133323131)

# LỜI MỞ ĐẦU

Trong thời đại kỹ thuật số, dữ liệu đã trở thành tài nguyên quan trọng trong việc ra quyết định và phát triển kinh doanh. Tuy nhiên, khi số lượng dữ liệu lớn và phức tạp, việc phân tích và xử lý trở nên khó khăn và tốn nhiều thời gian. Apache Hadoop là một công cụ mã nguồn mở được sử dụng rộng rãi trong phân tích dữ liệu lớn, cho phép xử lý và lưu trữ dữ liệu quy mô lớn với hiệu suất cao.

Trong đề tài này, chúng em sẽ trình bày quá trình xây dựng một ứng dụng phân tích dữ liệu sử dụng Apache Hadoop, từ việc chuẩn bị dữ liệu đến triển khai và tối ưu hoá hiệu suất của ứng dụng trên một cụm Hadoop. Bên cạnh đó, chúng tôi cũng sẽ giới thiệu các lợi ích của việc sử dụng Apache Hadoop trong phân tích dữ liệu lớn và cách sử dụng các kỹ thuật như chia nhỏ dữ liệu, tối ưu hoá MapReduce và cấu hình hệ thống để tăng hiệu suất và độ tin cậy của ứng dụng.

# NỘI DUNG

## **1. GIỚI THIỆU**

### 1.1 Giới thiệu chung về Big Data

Big data là thuật ngữ được sử dụng để mô tả các dữ liệu khổng lồ - với lượng dữ liệu cực kỳ lớn, phức tạp và đa dạng, mà không thể xử lý bằng các công cụ và phương pháp truyền thống. Điều này đòi hỏi các công nghệ và phương pháp xử lý dữ liệu đặc biệt, như Apache Hadoop. Apache Hadoop là một hệ thống phân tán và nguồn mở, được dùng để lưu trữ và xử lý dữ liệu lớn. Với khả năng mở rộng linh hoạt, Hadoop cho phép xử lý các tập dữ liệu khổng lồ trên các nút máy tính trong một cụm. Ứng dụng phân tích dữ liệu xây dựng trên Apache Hadoop có khả năng xử lý các tập dữ liệu lớn và phát hiện ra những thông tin và xu hướng quan trọng - giúp cho doanh nghiệp đưa ra các quyết định thông minh và kịp thời trong thị trường cạnh tranh ngày càng khốc liệt hơn.

### 1.2 Giới thiệu về Apache Hadoop

Apache Hadoop là một framework mã nguồn mở được sử dụng để xử lý, lưu trữ và phân tích các tập dữ liệu lớn (Big Data). Hướng đến mục tiêu chung là giảm thiểu thời gian và chi phí khi phân tích dữ liệu lớn.

Hadoop cung cấp một hệ thống phân tán cho phép phân tích dữ liệu trên hàng nghìn máy chủ. Nó bao gồm hai thành phần chính: Hadoop Distributed File System (HDFS) và MapReduce. HDFS là hệ thống lưu trữ phân tán, trong khi MapReduce được sử dụng để xử lý các tác vụ phân tán.

Hadoop đã được sử dụng rộng rãi trong các ứng dụng Big Data như phân tích dữ liệu xã hội, phân tích ngôn ngữ tự nhiên và dự báo hệ thống tài chính. Nó cũng được sử dụng trong các lĩnh vực liên quan đến y tế và khoa học.

Hadoop viết bằng Java. Tuy nhiên, nhờ cơ chế streaming, Hadoop cho phép phát triển các ứng dụng phân tán bằng cả java lẫn một số ngôn ngữ lập trình khác như C++,

Python, Pearl.

**Kiến trúc Hadoop:** Apache Hadoop dùng để quản lý và truy cập dữ liệu, và chỉ bao gồm 2 thành phần là: MapReduce và Hadoop Distributed File System (HDFS). Theo thời gian, nền tảng Hadoop mở rộng kết hợp với một loạt các dự án khác để thành một nền tảng hoàn chỉnh. Nền tảng này chia thành 5 loại sau: data access, data management, security, operations và governance.

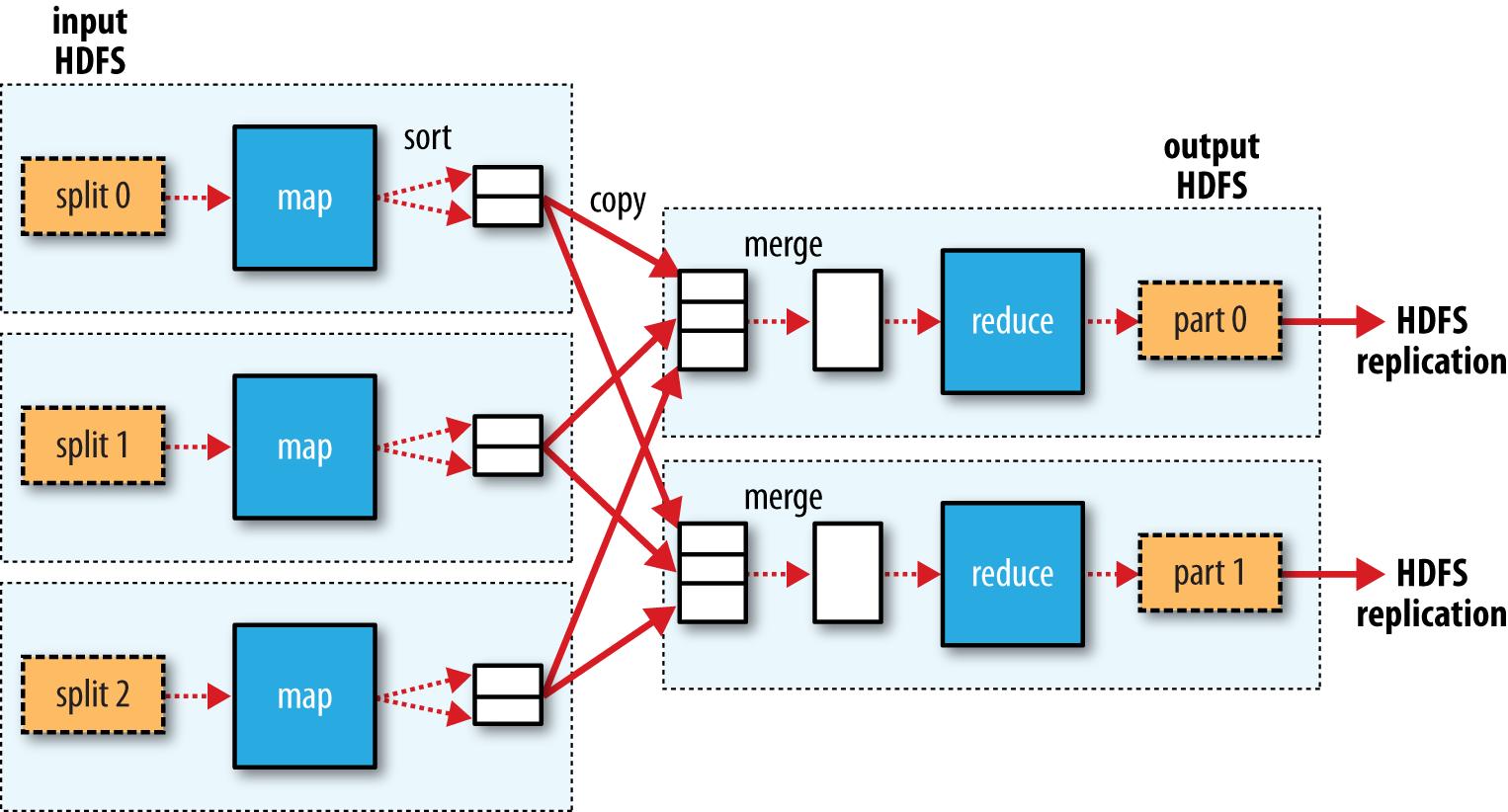


*Hình 1.1 Kiến trúc Hadoop*

**Hadoop MapReduce**: Map-Reduce là một framework dùng để viết các ứng dụng xử lý song song một lượng lớn dữ liệu có khả năng chịu lỗi cao xuyên suốt hàng ngàn cluster(cụm) máy tính. Map-Reduce thực hiện 2 chức năng chính đó là **Map** và **Reduce.**

- Map: Sẽ thực hiện đầu tiên, có chức năng tải, phân tích dữ liệu đầu vào và được chuyển đổi thành tập dữ liệu theo cặp key/value

- Reduce: Sẽ nhận kết quả đầu ra từ tác vụ Map, kết hợp dữ liệu lại với nhau thành tập dữ liệu nhỏ hơn.



*Hình 1.2. Mô hình MapReduce*

Các tính năng của MapReduce:

- Xử lý cục bộ.

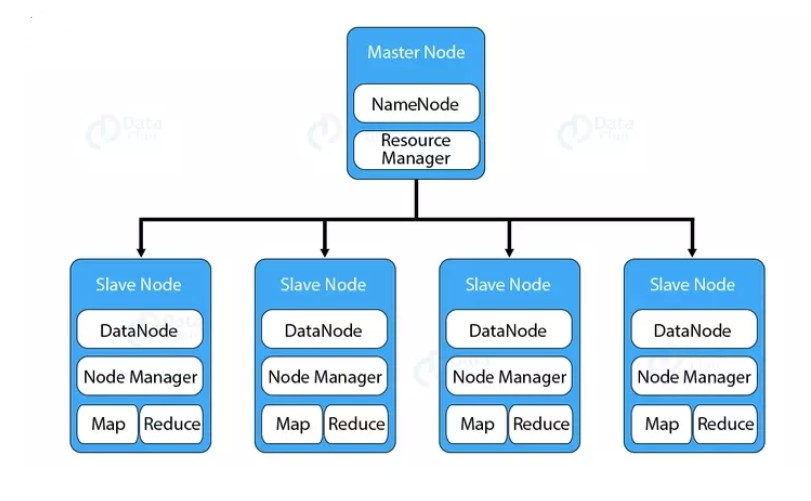
- Xây dựng sẵn.

- Ngôn ngữ độc lập.

- MapReduce execution framework..

- Truyền thông liên tiến trình.

**Hadoop Distributed File System (HDFS)**: lưu trữ dữ liệu trên nhiều máy tính khác nhau trong cụm. Hadoop MapReduce là một khung công cụ cho phép lập trình viên xử lý dữ liệu trên Hadoop. MapReduce phân tách các tác vụ phân tích dữ liệu thành các phần nhỏ và phân phối chúng trên toàn bộ cụm để xử lý đồng thời. Kết quả được thu thập và đồng bộ hóa lại để tạo ra kết quả cuối cùng.



*Hình 1.3. Mô hình Hadoop Distributed File System*

Cụ thể cơ chế master – slave được thể hiện trong HDFS bằng việc trong một cụm máy, sẽ chỉ bao gồm một máy duy nhất được gọi là là Namenode (Master) và các máy còn lại gọi là Datanode (Slave).

* Datanode sẽ là nơi lưu trữ các file dữ liệu mà bạn đưa vào.
* Namenode là nơi lưu địa chỉ của file đó được chia và lưu trên các datanode nào.
* Namenode

Qua cơ chế master – slave, vai trò của namenode cũng đã được thể hiện phần nào. Nhưng mình sẽ trình bày chi tiết hơn về vai trò của namenode trong phần này. Cụ thể:

* Namenode là đầu mối truy cập và thực hiện các thao tác với file từ phía client.
* Namenode có nhiệm vụ duy trì và quản lý các datanode.
* Namenode là nơi lưu trữ và cập nhật các metadata, ví dụ như địa chỉ của các block trên datanode, quyền truy cập của client,.. Có 2 loại file metadata:
* FsImage: Là nơi lưu trữ trạng thái của các file system namespace (từ này hơi khó để dịch) kể từ khi namenode được khởi động.
* EditLogs: Là nơi lưu lại quá sửa đổi của các file system.
* Nó luôn lắng nghe và theo dõi các datanode để đảm bảo là chúng vẫn “còn sống”.
* Namenode còn là nơi điều khiển sự cân bằng về dung lượng lưu trữ của các datanode, quản lý lượng truy cập tới các datanode.
* Datanode

Datanode chính là các máy slave trong kiến trúc của HDFS, nó có các vai trò và chức năng sau:

* Là nơi lưu trữ dữ liệu, khi dữ liệu được đưa vào HDFS.
* Là nơi chạy các tiến trình xử lý dữ liệu (các job).
* Gửi dữ liệu định kỳ về tình trạng “sức khỏe” tới datanode, mặc định là cứ 3s nó sẽ gửi một lần.

Không giống như namenode, datanode không yêu cầu tính sẵn sàng cao, bởi trong một hadoop cluster thì thường có nhiều datanode. Vì vậy mà các máy datanode chỉ cần chạy trên các phần cứng thông thường, và dễ dàng bổ sung thêm datanode với chi phí rẻ.

Các tính năng của HDFS:

- Chịu lỗi tốt.

- Độ tin cậy cao.

- Ứng dụng cao.

- Khả năng mở rộng.

- Lưu trữ phân tán.

Bên cạnh đó ngoài 2 thành phần cốt lõi trên cũng còn có một số modules :

- Hadoop YARN: là một Phần mềm nền tảng quản lý tài nguyên cho các ứng dụng phân tán. Nó được thiết kế để cung cấp một kiến trúc linh hoạt và hỗ trợ cho các ứng dụng phân tích dữ liệu chạy trên Hadoop. YARN cung cấp daemons và APIs cần thiết cho việc phát triển ứng dụng phân tán, ồng thời xử lý, lập lịch sử dụng tài nguyên tính toán (CPU hay Memory) cũng như giám sát quá trình thực thi các ứng dụng.

- Hadoop Common: Đây là một khung công cụ phân tán mã nguồn mở được thiết kế để xử lý, lưu trữ và phân tích dữ liệu lớn trên các cụm máy tính. Hadoop cho phép các nhà phát triển xây dựng các ứng dụng phân tích dữ liệu với khả năng xử lý lượng dữ liệu lớn một cách nhanh chóng và hiệu quả.

### 1.3 Mục tiêu đề tài

Xây dựng một ứng dụng phân tích dữ liệu sử dụng Apache Hadoop.

### 1.4 Hướng giải quyết

Xử lý và làm việc khối lượng dữ liệu khổng lồ tính bằng Petabyte.

Xử lý trong môi trường phân tán, dữ liệu lưu trữ ở nhiều phần cứng khác nhau, yêu cầu xử lý ồng bộ

Xử lý các lỗi xuất hiện thường xuyên.

### 1.5 Khó khăn và thách thức

Số lượng và đa dạng của dữ liệu: Apache Hadoop được xây dựng để xử lý dữ liệu lớn, tuy nhiên với lượng dữ liệu ngày càng tăng và đa dạng hơn, việc phân tích và quản lý dữ liệu trở nên phức tạp hơn.

Hiệu suất và tối ưu hóa: Việc tối ưu hiệu suất của Hadoop là một thách thức lớn nhất đối với các ứng dụng phân tích dữ liệu. Đảm bảo rằng ứng dụng của bạn hoạt động hiệu quả với các quy trình tối ưu hóa mạnh mẽ đối với cả phần cứng và dữ liệu.

Kỹ thuật và công nghệ mới: Khi xây dựng một ứng dụng phân tích dữ liệu sử dụng Apache Hadoop, bạn cần nắm rõ các công nghệ mới và kỹ thuật đang phát triển. Sẽ rất khó khăn để tạo ra một ứng dụng hoàn chỉnh và hiệu quả nếu bạn không có hiểu biết vững chắc về kỹ thuật và công nghệ này.

### 1.6 Hướng giải quyết

Tối ưu hóa dữ liệu: Cải thiện việc tối ưu hóa dữ liệu là một trong những cách giải quyết tốt nhất để đối phó với khối lượng lớn và đa dạng của dữ liệu. Sử dụng các công cụ như Hive hoặc Pig để phân tích dữ liệu sẽ giúp cho việc quản lý dữ liệu trở nên dễ dàng hơn.

Nâng cao hiệu suất Hadoop: Để tận dụng được tối đa khả năng của Apache Hadoop, bạn có thể sử dụng Spark hoặc Storm để tối ưu hóa hiệu suất và khả năng xử lý của nó.

Sử dụng các giải pháp công nghệ mới: Tận dụng các công nghệ và kỹ thuật mới để phát triển ứng dụng phân tích dữ liệu. Ví dụ như sử dụng công nghệ Artificial Intelligence (AI) để phân tích dữ liệu và rút ra các thông tin quan trọng, điều này sẽ giúp tăng tốc độ phân tích dữ liệu và cải thiện chính xác của kết quả.

## **2. NỀN TẢNG CÔNG NGHỆ**

### 2.1 Các thành phần trong Hadoop

Hadoop Distributed File System (HDFS): HDFS là một hệ thống tập tin phân tán cho phép lưu trữ các tập tin lớn trên nhiều thiết bị lưu trữ khác nhau. Nó tạo ra một lớp giữa ứng dụng và các thiết bị lưu trữ, cho phép các tệp được phân tán trên các nút khác nhau trong một cụm Hadoop.

MapReduce: MapReduce là một khung công việc phân tán cho phép xử lý dữ liệu trên nhiều nút trong một cụm Hadoop.

Hadoop YARN: Hadoop YARN (Yet Another Resource Negotiator) là một phần mở rộng của Hadoop, cho phép quản lý tài nguyên cho các ứng dụng có thể chạy trên một cụm Hadoop.

Apache Spark: Apache Spark là một framework mới cho xử lý dữ liệu phân tán, được thiết kế để xử lý dữ liệu nhanh hơn và hiệu quả hơn so với MapReduce.

Hadoop Ecosystem: Hadoop cung cấp nhiều công cụ và framework để hỗ trợ phân tích dữ liệu, bao gồm Hive, HBase, Pig, và Sqoop.

Các ngôn ngữ lập trình: Ứng dụng phân tích dữ liệu được phát triển bằng các ngôn ngữ lập trình Java.

Hệ thống máy chủ: Các nút trong cụm Hadoop phải được cấu hình và cài đặt trên các máy chủ có phần cứng và mạng phù hợp để đáp ứng nhu cầu sử dụng.

Cách triển khai: Cần phải có kế hoạch triển khai chi tiết để xây dựng, triển khai và vận hành ứng dụng phân tích dữ liệu trên cụm Hadoop.

### 2.2 Các công nghệ quan trọng

Apache Hadoop Distributed File System (HDFS): Là một hệ thống lưu trữ phân tán, được thiết kế để xử lý những tệp dữ liệu lớn trong môi trường phân tán. HDFS được sử dụng để lưu trữ dữ liệu và đảm bảo tính toàn vẹn dữ liệu trong cụm Apache Hadoop.

Apache Hive: Là một công cụ giúp thực hiện truy vấn và phân tích dữ liệu lớn trên các hệ thống Hadoop. Hive cho phép sử dụng ngôn ngữ SQL để thao tác với dữ liệu lưu trữ trong Hadoop, giúp đơn giản hóa việc phân tích dữ liệu.

Apache Pig: Cũng là một công cụ giúp thực hiện phân tích dữ liệu trên Hadoop, nhưng sử dụng một ngôn ngữ lập trình riêng để biểu diễn các tác vụ phân tích. Pig cho phép xử lý dữ liệu trong các chuỗi các tác vụ phức tạp và hoàn toàn có thể tùy chỉnh.

Apache Spark: Là một hệ thống xử lý dữ liệu phân tán mã nguồn mở, được tích hợp với Hadoop và cung cấp khả năng xử lý số liệu lớn với tốc độ cao và hiệu quả hơn. Spark hỗ trợ các cách tiếp cận phân tích dữ liệu khác nhau, bao gồm xử lý theo thời gian thực, học máy và tính toán phân tán.

Hadoop YARN: Là một khung công việc quản lý tài nguyên cho phép chạy các ứng dụng trên Hadoop, bao gồm cả Spark, Hive và Pig. YARN cho phép tập trung quản lý các tài nguyên phân phối, nhằm giảm bớt áp lực cho các quản trị viên hệ thống và tăng hiệu suất.

Hadoop MapReduce: Là một khung công việc phân tán, được sử dụng để xử lý và phân tích dữ liệu lớn. MapReduce cho phép xử lý dữ liệu trên cụm Hadoop bằng cách chia nhỏ tác vụ thành các tác vụ con, sau đó gộp kết quả từ các tác vụ con lại với nhau để tạo ra kết quả cuối cùng.

### 2.3 Giải thuật sử dụng

Thuật toán MapReduce:

**Mô tả thuật toán**:

- MapReduce có hai giai đoạn chính: Map và Reduce.

- Trong giai đoạn Map, dữ liệu được chia nhỏ thành các đoạn nhỏ hơn và được gửi đến các máy tính xử lý phân tán. Mỗi máy tính sẽ nhận một phần dữ liệu và thực hiện các phép tính trên đó theo một hàm ánh xạ được định nghĩa sẵn. Hàm ánh xạ này sẽ lấy đầu vào là một cặp key-value và đưa ra một tập hợp các cặp key-value mới.

- Trong giai đoạn Reduce, các kết quả từ giai đoạn Map được tổng hợp lại. Các máy tính xử lý phân tán sẽ được phân công các cặp key-value theo key và thực hiện các phép tính trên chúng. Kết quả sau cùng sẽ được trả về thành một tập hợp các cặp key-value mới.

- Nhờ đó, giải thuật MapReduce đã giúp chúng ta có thể xử lý được rất nhiều dữ liệu lớn một cách hiệu quả. Đặc biệt là trong các ứng dụng phân tích dữ liệu, MapReduce giúp phân tách dữ liệu và tính toán chúng một cách song song và nhanh chóng.

**Ứng dụng:**

- Xử lý dữ liệu lớn: MapReduce cho phép phân chia dữ liệu thành các phần nhỏ và xử lý chúng một cách đơn giản và hiệu quả. Điều này làm giảm thời gian xử lý của các tập dữ liệu lớn đến mức chúng ta có thể làm việc trên hàng terabyte dữ liệu.

- Phân tích bất cứ dữ liệu nào: MapReduce được sử dụng để phân tích bất cứ loại dữ liệu nào, từ log file, dữ liệu mạng xã hội, và các dạng dữ liệu khác.

- Tăng tốc độ xử lý dữ liệu: MapReduce giúp tăng tốc độ xử lý dữ liệu lên đến hàng trăm lần so với phương pháp truyền thống.

- Phân tích dữ liệu thời gian thực: MapReduce có thể được sử dụng để phân tích dữ liệu thời gian thực và trả về kết quả trong thời gian gần như ngay lập tức.

- Tăng tính khả dụng và tin cậy của hệ thống: Hadoop cung cấp khả năng chịu lỗi, giảm thiểu các lỗi phát sinh trong quá trình xử lý dữ liệu và giúp tăng độ tin cậy của hệ thống.

**Ví dụ minh họa**

- Một công ty bán lẻ có hệ thống lưu trữ tất cả các đơn hàng của khách hàng trong database. Họ muốn phân tích các dữ liệu đó để hiểu được xu hướng mua sắm của khách hàng trong từng khoảng thời gian cụ thể. Cho dữ liệu có kích thước lớn, công ty này sử dụng MapReduce để phân tán và xử lý các tập dữ liệu này. Sau khi xử lý, công ty có thể thu được kết quả phân tích một cách nhanh chóng và hiệu quả hơn.

## **3. THIẾT KẾ ỨNG DỤNG**

### 3.1 Mô tả yêu cầu

Khả năng xử lý dữ liệu lớn: Ứng dụng cần có khả năng xử lý cả dữ liệu có kích thước lớn, chẳng hạn như hàng tỷ hoặc hàng trăm tỷ bản ghi, và đảm bảo hiệu suất và thời gian phản hồi nhanh chóng.

Tính linh hoạt: Ứng dụng phân tích dữ liệu cần có khả năng hoạt động với nhiều nguồn dữ liệu khác nhau và có thể thay đổi cấu trúc dữ liệu mà không làm ảnh hưởng đến hệ thống.

Khả năng truy vấn dữ liệu: Ứng dụng phân tích dữ liệu cần có khả năng truy vấn dữ liệu một cách hiệu quả và chính xác, giúp người dùng tìm thấy thông tin cần thiết để đưa ra quyết định.

Khả năng định dạng dữ liệu: Ứng dụng phân tích dữ liệu cần có khả năng định dạng dữ liệu một cách chuẩn để đảm bảo tính thống nhất của dữ liệu trong quá trình phân tích.

Độ tin cậy cao: Ứng dụng phân tích dữ liệu cần đảm bảo tính tin cậy và bảo mật của dữ liệu, đặc biệt là khi xử lý các dữ liệu nhạy cảm về bảo mật.

Khả năng tương thích cao: Ứng dụng phân tích dữ liệu cần có khả năng tương thích với các nền tảng khác nhau, giúp dễ dàng tích hợp và sử dụng với các ứng dụng và hệ thống khác.

Tiết kiệm tài nguyên: Ứng dụng phân tích dữ liệu cần có khả năng tối ưu hóa tài nguyên sử dụng như bộ nhớ và băng thông để đảm bảo hiệu suất và chi phí tối thiểu.

### 3.2 Tính năng chính

Hỗ trợ các công cụ phân tích dữ liệu: Ứng dụng phải khả năng hỗ trợ các công cụ phân tích dữ liệu như MapReduce, Hive, Pig, Spark để có thể giải quyết các vấn đề phức tạp như xử lý dữ liệu theo thời gian thực, phân tích dữ liệu đa chiều, trích xuất thông tin từ dữ liệu vô cấu trúc.

Tích hợp với Hadoop cluster: Ứng dụng phải tích hợp tốt với Hadoop cluster để có thể trích xuất và xử lý dữ liệu lớn theo thời gian thực, thực hiện các tác vụ liên quan đến xử lý dữ liệu và tính toán và đồng bộ hóa dữ liệu trên nhiều node.

Cải thiện tối đa thời gian xử lý: Ứng dụng cần cải thiện tối đa thời gian xử lý dữ liệu bằng cách tối ưu hóa quy trình xử lý dữ liệu, tối ưu lưu trữ dữ liệu trên ổ cứng và truyền tải dữ liệu giữa các node trong cluster bằng cách sử dụng các công nghệ tiên tiến nhất.

Bảo mật dữ liệu: Ứng dụng phải cung cấp các tính năng bảo mật dữ liệu vượt trội như mã hóa dữ liệu, chứng thực xác thực người dùng, giám sát các hoạt động truy cập dữ liệu để đảm bảo an toàn thông tin quý giá của người dùng.

### 3.3 Thiết kế kiến trúc của ứng dụng

### 3.4 Mô tả luồng xử lý dữ liệu

Thu thập dữ liệu: Dữ liệu cần được thu thập từ nhiều nguồn khác nhau, ví dụ như các file log của server, cơ sở dữ liệu, hoặc các thiết bị IoT. Sau khi thu thập, dữ liệu được gộp lại và chuẩn hóa để chuẩn bị cho bước xử lý tiếp theo.

Lưu trữ dữ liệu: Dữ liệu sau khi được chuẩn hóa được lưu trữ trên Hadoop HDFS, hệ thống lưu trữ phân tán của Apache Hadoop. HDFS cho phép lưu trữ và quản lý các file và thư mục có kích thước rất lớn (ví dụ như GB hoặc TB).

Xử lý dữ liệu: Dữ liệu trên HDFS được xử lý bằng Apache Hadoop MapReduce framework, một cơ chế tính toán phân tán cho phép xử lý dữ liệu khối lượng lớn. Quá trình xử lý này bao gồm:

- Mapper: Hadoop MapReduce sử dụng Mapper để chuyển đổi dữ liệu đầu vào thành các cặp key-value. Mapper được thiết kế để xử lý dữ liệu đầu vào phân tán, ngang hàng trên các node của Hadoop cluster.

- Reducer: Các cặp key-value được nhóm lại và được đưa vào Reducer để thực hiện các tính toán phức tạp. Reducer sẽ đưa ra các kết quả của quá trình xử lý dữ liệu.

Lưu trữ kết quả: Kết quả của quá trình xử lý được lưu trữ trên Hadoop HDFS. Điều này giúp cho việc truy cập và truy vấn kết quả trở nên dễ dàng hơn.

Truy vấn dữ liệu: Kết quả của quá trình xử lý có thể được truy vấn bằng Apache Hive hay Apache HBase. Apache Hive là một công cụ truy vấn dữ liệu biến đổi dựa trên SQL, còn Apache HBase là một hệ thống lưu trữ dữ liệu NoSQL.

### 3.5 Xây dựng một số hàm phân tích dữ liệu trên Hadoop

Xây dựng dữ liệu trên Hadoop rất khó khăn nên với việc chúng ta sử dụng hàm để phân tích dữ liệu trở nên dễ dàng và hiệu quả hơn, qua đó giúp cho ứng dụng phân tích dữ liệu trở nên chính xác và đáng tin cậy hơn.

Một số hàm được sử dụng như sau:

- Hàm MapReduce: Là hàm cơ bản trong Hadoop, nó giúp xử lý các tập dữ liệu lớn bằng phương pháp phân tán. Hàm MapReduce sẽ được sử dụng để đọc dữ liệu, chia nhỏ dữ liệu thành các khối và phân tán chúng cho các máy tính con xử lý.

- Hàm GroupBy: Sử dụng để nhóm dữ liệu thành các nhóm nhỏ hơn dựa trên các thuộc tính của chúng. Hàm GroupBy giúp cho việc phân tích dữ liệu trở nên dễ dàng hơn bằng cách tổng hợp các thông tin tương tự thành một nhóm duy nhất.

- Hàm Join: Sử dụng để kết hợp nhiều bảng dữ liệu khác nhau với nhau, giúp phân tích dữ liệu từ nhiều nguồn trở nên thuận tiện hơn. Hàm Join có thể được sử dụng để kết hợp các bảng dữ liệu đến từ hàng triệu bản ghi khác nhau.

- Hàm Filter: Sử dụng để lọc ra các bản ghi trong tập dữ liệu dựa trên các điều kiện nhất định. Hàm Filter giúp cho việc phân tích dữ liệu trở nên dễ dàng hơn bằng cách loại bỏ các bản ghi không cần thiết, làm giảm kích thước của tập dữ liệu và tăng tốc độ xử lý.

- Hàm Aggregate: Sử dụng để tính toán các giá trị thống kê dựa trên các thuộc tính của tập dữ liệu. Hàm Aggregate có thể được sử dụng để tính toán trung bình, đếm, tổng, tối đa và tối thiểu các giá trị của một tập dữ liệu.

- Hàm Sort: Sử dụng để sắp xếp các bản ghi trong tập dữ liệu theo một điều kiện nhất định. Hàm Sort giúp cho việc phân tích dữ liệu trở nên dễ dàng hơn bằng cách sắp xếp các bản ghi theo thứ tự từ thấp đến cao hoặc ngược lại.

## **4. TRIỂN KHAI ỨNG DỤNG**

### 4.1 Cài đặt Hadoop

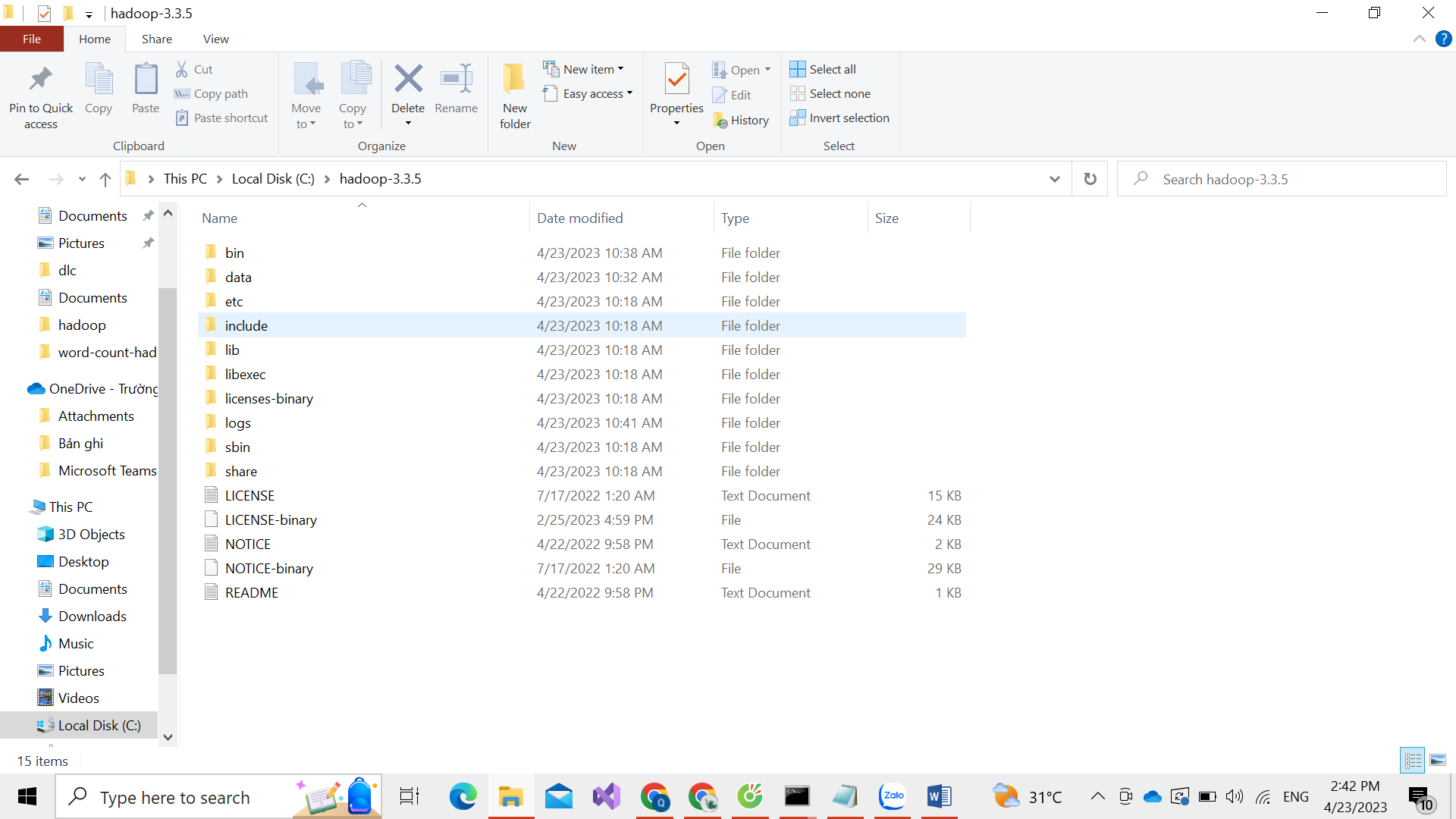
**Lưu ý:** Bắt buộc phải tải phiên bản java 8:

Link tải:<https://www.oracle.com/java/technologies/javase/javase8-archive-downloads.html>

Link tải hadoop:

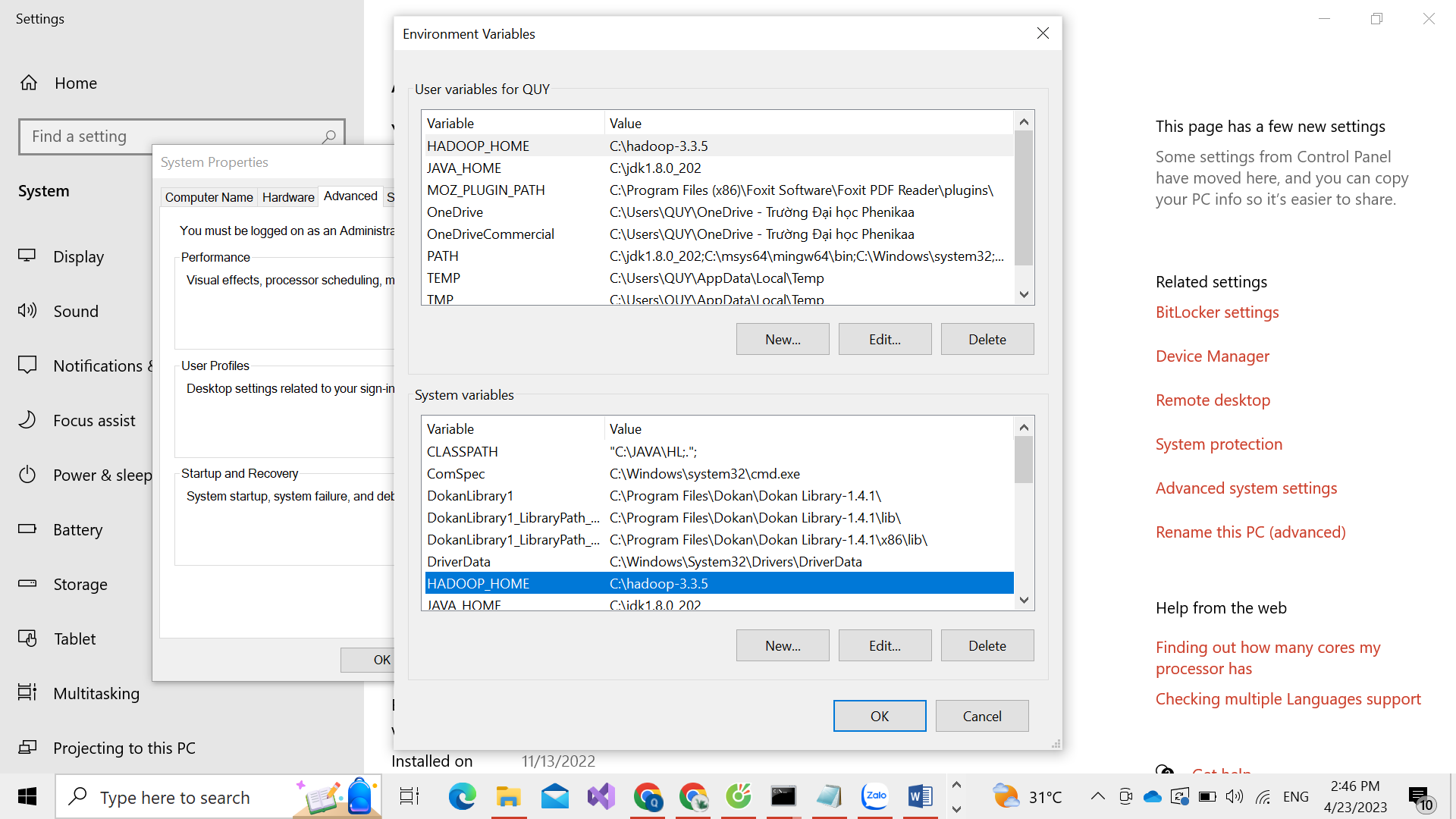
<https://www.apache.org/dyn/closer.cgi/hadoop/common/hadoop-3.3.5/hadoop-3.3.5.tar.gz>

Bước 1 :



*Hình 4.1.1 Thư mục hadoop 3.3.5*

Bước 2 : Thiết lập biến môi trường



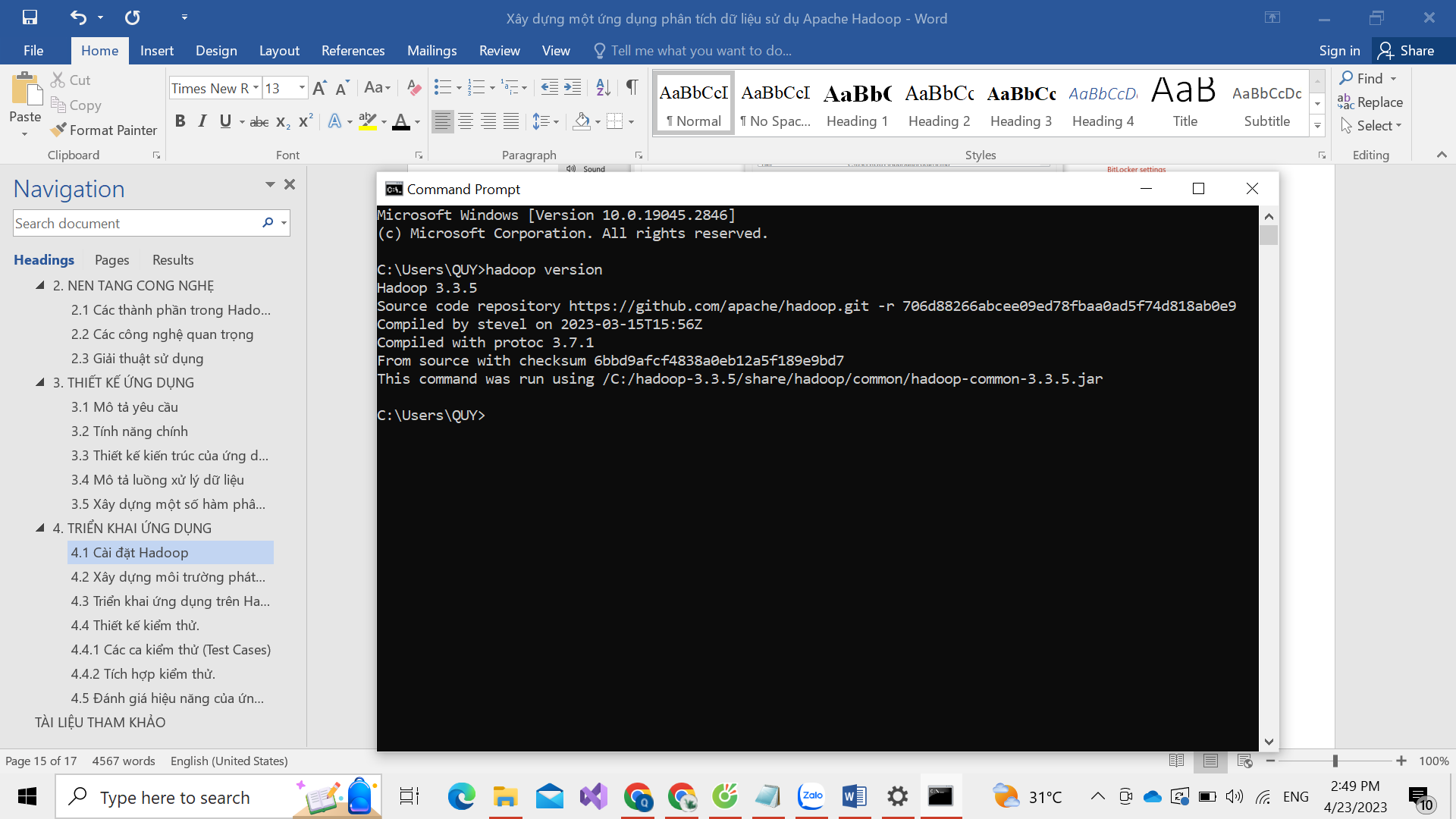
*Hình 4.1.2 Thiết lập biến môi trường*

Sau đó chỉnh sửa biến path cho cả user và system variable. Bổ sung thêm:

%HADOOP\_HOME%\bin

%HADOOP\_HOME%\sbin

Bước 3 : Mở cmd để test lại



Hình 4.1.3 : Chạy hadoop trên cmd

### 4.2 Xây dựng môi trường phát triển

Trong thư mục **C:/Hadoop-3.3.0/etc/hadoop** lần lượt chỉnh sửa các file :

* core-site.xml
* mapred-site.xml
* hdfs-site.xml
* yarn-site.xml
* hadoop-env.cmd

Bước 1 : Sửa cấu hình core-site.xml

<configuration>

  <property>

    <name>fs.defaultFS</name>

    <value>hdfs://localhost:9000</value>

  </property>

</configuration>

Bước 2 : Sửa cấu hình mapred-site.xml

<configuration>

  <property>

    <name>mapreduce.framework.name</name>

    <value>yarn</value>

  </property>

<property>

    <name>yarn.app.mapreduce.am.env</name>

    <value>HADOOP\_MAPRED\_HOME=$HADOOP\_HOME</value>

</property>

<property>

   <name>mapreduce.map.env</name>

   <value>HADOOP\_MAPRED\_HOME=$HADOOP\_HOME</value>

</property>

<property>

   <name>mapreduce.reduce.env</name>

   <value>HADOOP\_MAPRED\_HOME=$HADOOP\_HOME</value>

</property>

</configuration>

Bước 3 : Sửa cấu hình hdfs-site.xml

<configuration>

  <property>

    <name>dfs.replication</name>

    <value>2</value>

  </property>

  <property>

    <name>dfs.namenode.name.dir</name>

    <value>C:\hadoop\data\namenode</value>

  </property>

    <property>

    <name>dfs.datanode.data.dir</name>

    <value>C:\hadoop\data\datanode</value>

  </property>

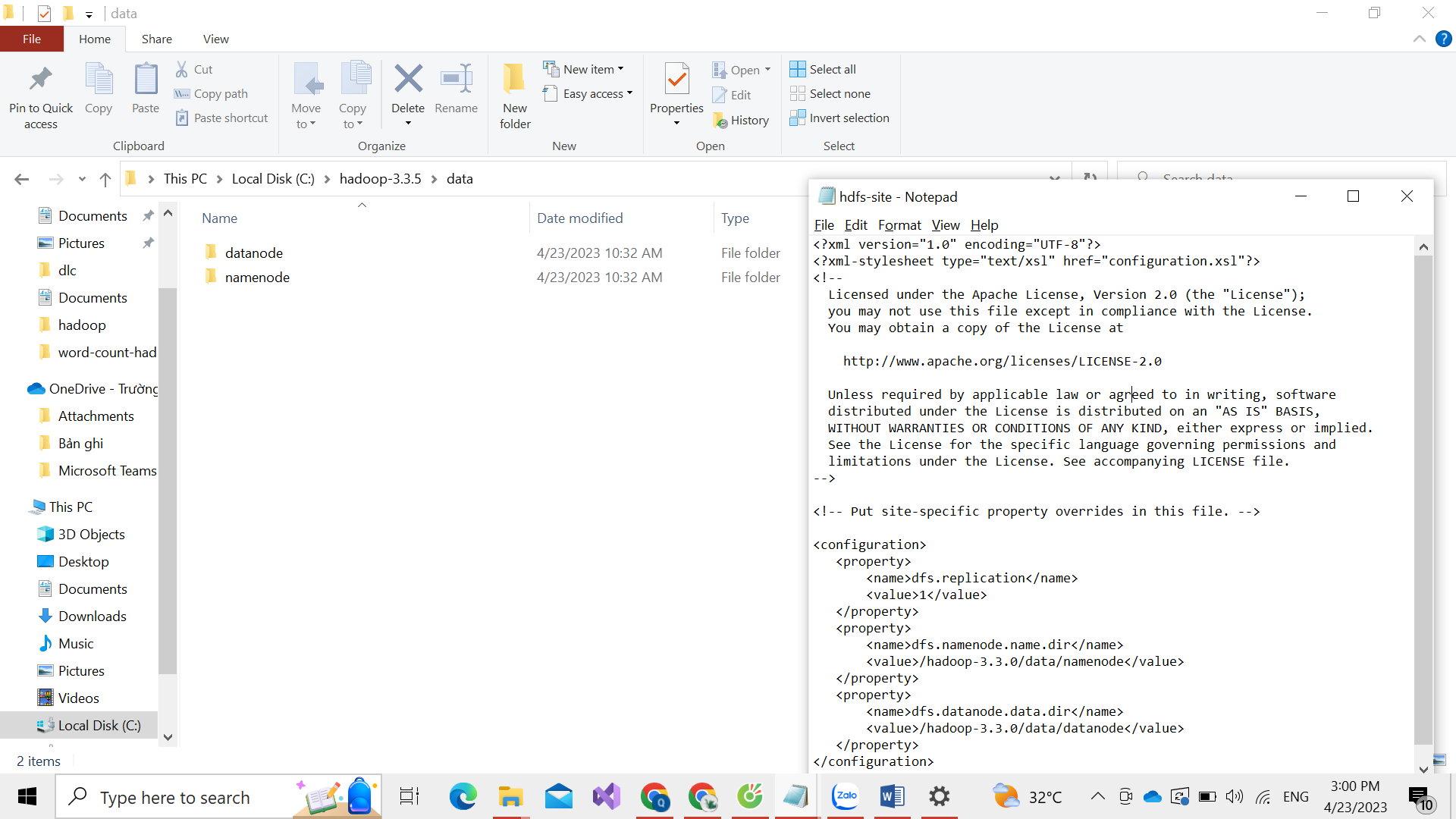
  <property>

  <name>dfs.permissions</name>

  <value>false</value>

</property>

</configuration>



*Hình 4.2.1 Cấu hình hdfs-site.xml*

Bước 4: Sửa cấu hình yarn-site.xml

<configuration>

  <property>

    <name>yarn.nodemanager.aux-services</name>

    <value>mapreduce\_shuffle</value>

  </property>

  <property>

    <name>yarn.nodemanager.auxservices.mapreduces.shuffle.class</name>

    <value>org.apache.hadoop.mapred.ShuffleHandler</value>

  </property>

   <property>

       <name>yarn.application.classpath</name>

       <value>

            %HADOOP\_HOME%\etc\hadoop,

            %HADOOP\_HOME%\share\hadoop\common\\*,

            %HADOOP\_HOME%\share\hadoop\common\lib\\*,

            %HADOOP\_HOME%\share\hadoop\hdfs\\*,

            %HADOOP\_HOME%\share\hadoop\hdfs\lib\\*,

            %HADOOP\_HOME%\share\hadoop\mapreduce\\*,

            %HADOOP\_HOME%\share\hadoop\mapreduce\lib\\*,

            %HADOOP\_HOME%\share\hadoop\yarn\\*,

            %HADOOP\_HOME%\share\hadoop\yarn\lib\\*

       </value>

    </property>

</configuration>

Bước 5: Sửa cấu hình hadoop-env.cmd

Mở file này lên và tìm tới lệnh:

* set JAVA\_HOME=%JAVA\_HOME%

sửa %JAVA\_HOME% thành đường dẫn cài JDK trong ổ C:

* set JAVA\_HOME= C:/jdk1.8.0\_201

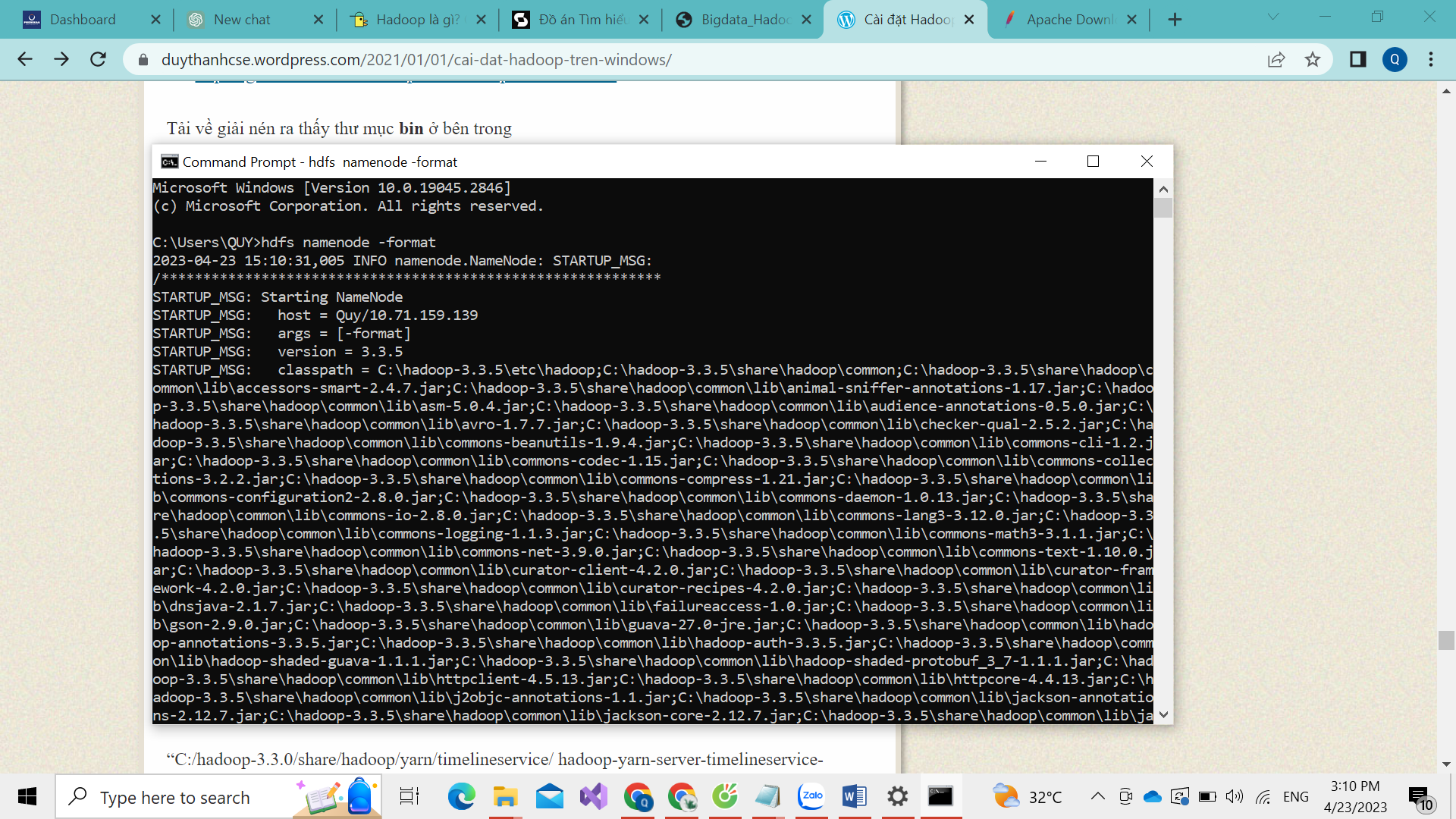
Bước 6: Tải về giải nén thư mục :

<https://github.com/s911415/apache-hadoop-3.1.0-winutils>

Chép đè thư mục bin này trong thư mục bin của C:\hadoop-3.3.0\bin

Sau đó format lại namenode và datanode: mở command line lên, gõ lệnh sau:

* hdfs namenode –format
* hdfs datanode -format

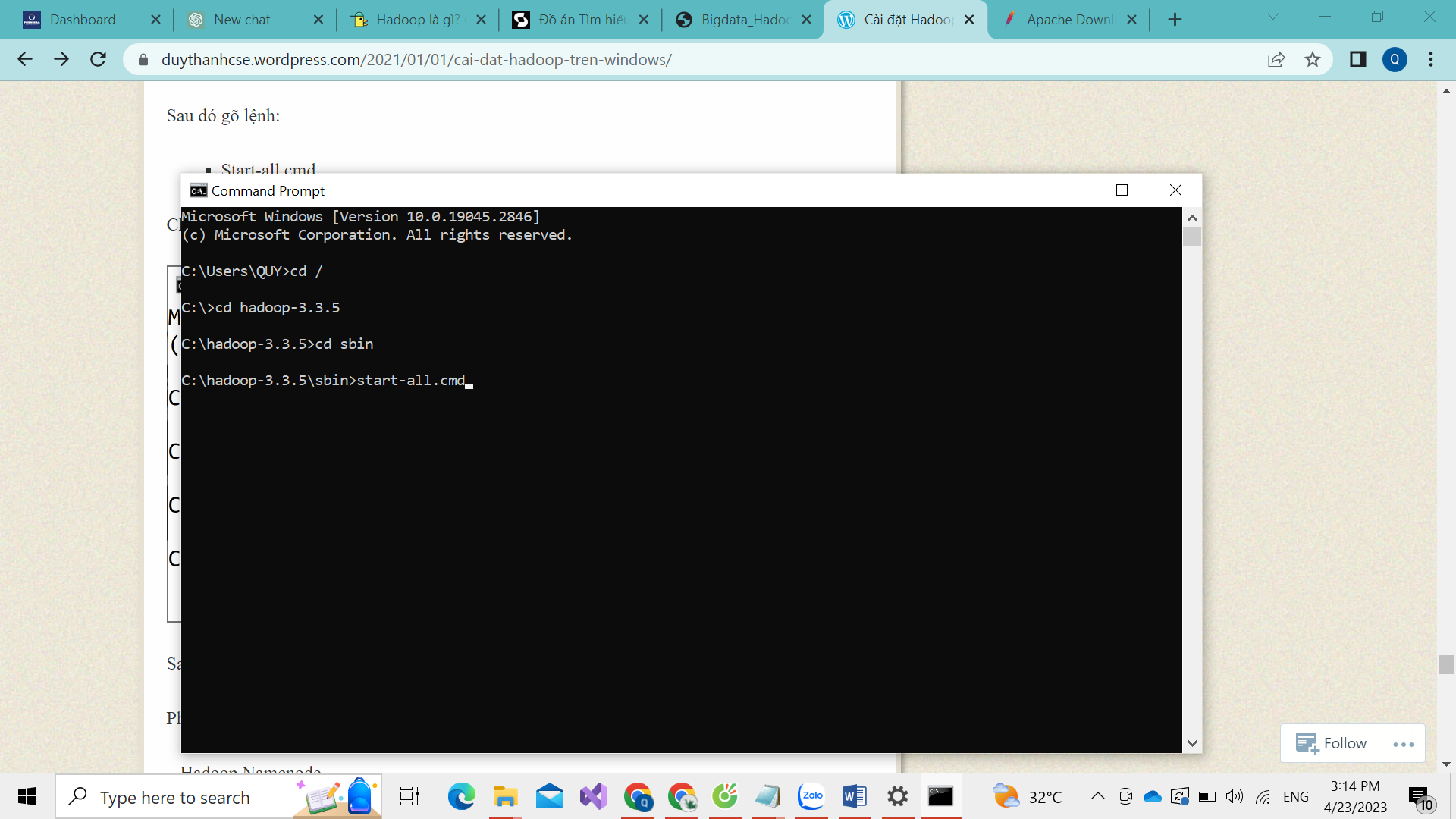


*Hình 4.2.2 Chạy hdfs namenode –format*

Bước 5 : Tiếp theo sao chép file:

“C:/hadoop-3.3.0/share/hadoop/yarn/timelineservice/ hadoop-yarn-server-timelineservice-3.3.0.jar” vào “C:/hadoop-3.3.0/share/hadoop/yarn/hadoop-yarn-server-timelineservice-3.3.0.jar”

Bước 6 : Hoàn thành cài đặt và test thử nghiệm start-all.cmd



*Hình 4.2.3 Chi tiết các lệnh*

Sau khi gõ lệnh trên hệ thống sẽ chạy Hadoop.

### 4.3 Triển khai ứng dụng trên Hadoop

Trong lần này, chúng em sẽ sử dụng bài toán chạy WordCount – Đếm từ nhằm minh họa cho quá trình MapReduce của mã nguồn.

Bài toán có những đặc điểm :

* File cần đếm rất lớn (quá lớn để có thể được tải lên bộ nhớ chính của 1 máy)
* Mỗi cặp quá lớn cho bộ nhớ

Mapreduce chia làm 3 thao tác :

* Map : Quét file đầu vào và ghi lại từng bản ghi
* Group by Key : Sắp xếp và trộn dữ liệu cho mỗi bản ghi sinh ra từ Map
* Reduce : Tổng hợp, thay đổi hay lọc dữ liệu từ thao tác trước và ghi kết quả ra File.

Về mặt định nghĩa thuật toán, ta có thể mô tả MR như sau :

* Input : Dữ liệu dưới dạng Key -> Value
* Lập trình viên viết 2 thủ tục :

Map(k,v) -> <k’,v’>

Reduce(k’,v’’>\*

* Với :

Map biến mỗi key k thu được bằng thành cặp <k’,v’>

Reduce nhận đầu vào là khóa k’ và danh sách cặp giá trị v’ và trả về kết quả là cặp <k’,v’’>.

Map trả về danh sách: <A, 1>, <A, 1> còn Reduce nhận kết quả trên và trả về <A, 2>.

Đầu tiên, chạy các ứng dụng trên cmd đảm bảo các ứng dụng được chạy

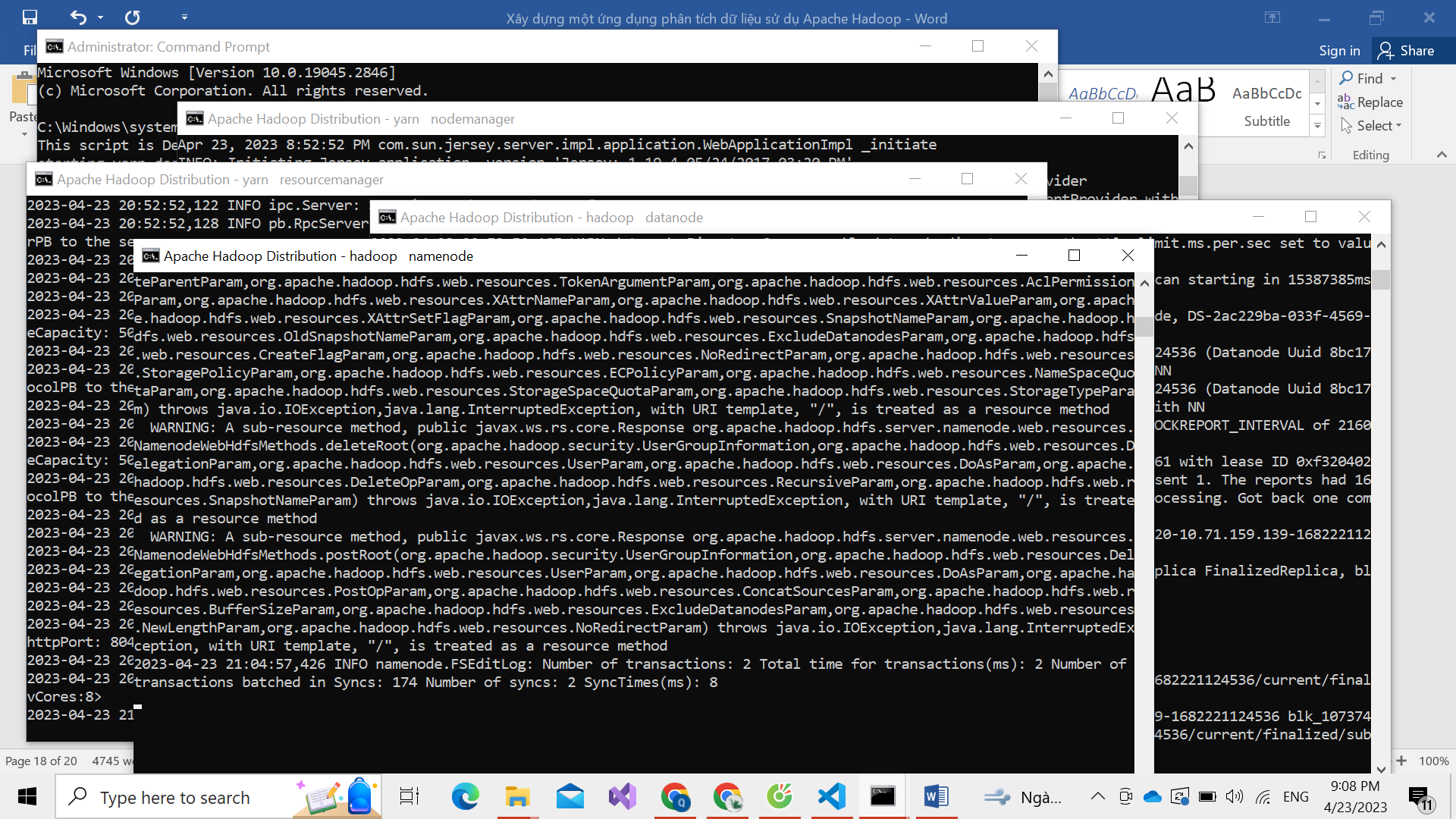
– Hadoop Namenode

– Hadoop datanode

– YARN Resource Manager

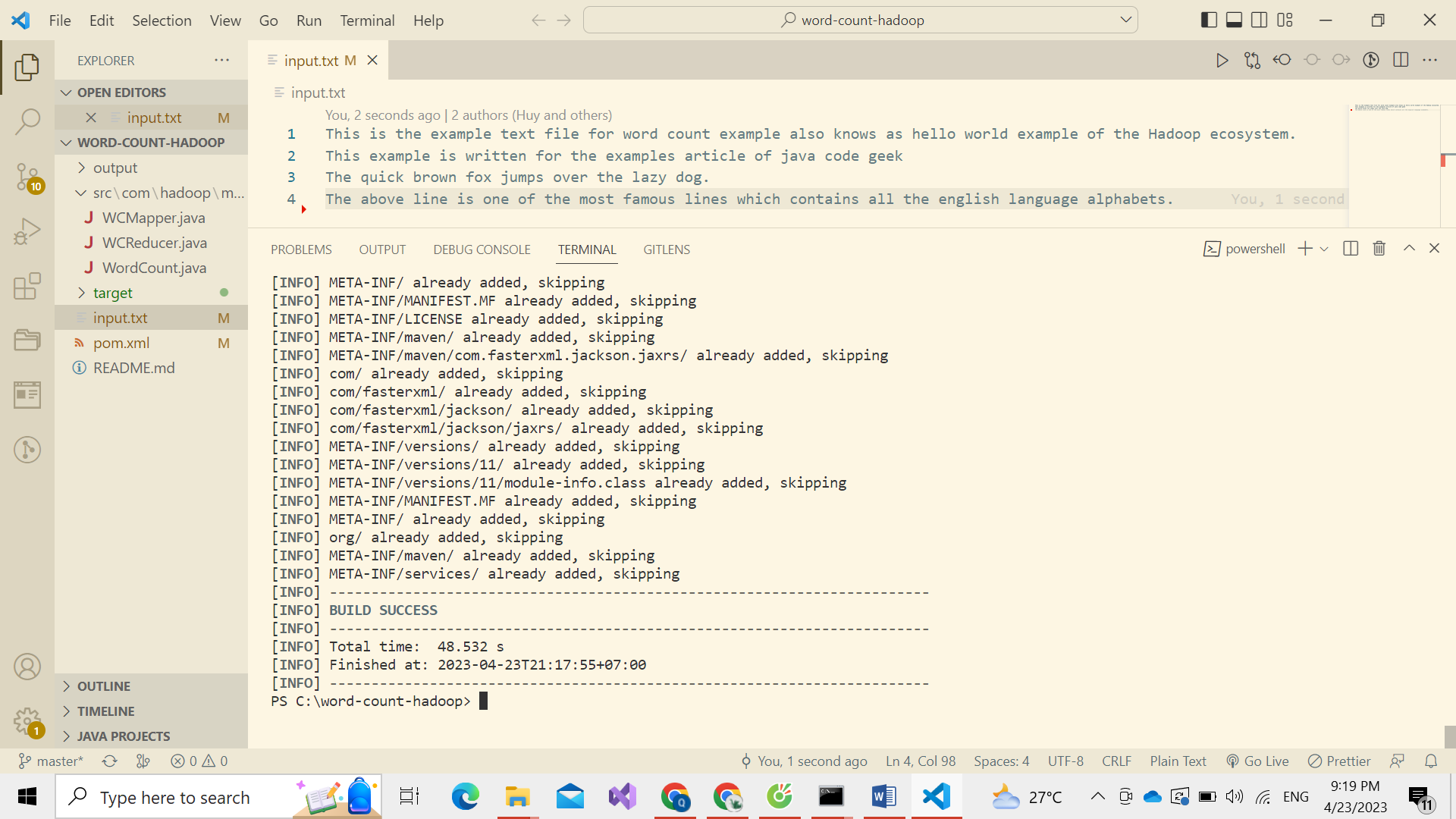
– YARN Node Manager

Bước 1 : Chạy lệnh “start-all.cmd” với quyền admin sẽ hiển thị ra như trên :



*Hình 4.3.1 Chạy các ứng dụng*

Bước 2 : Mở visual studio lên và đi tới thư mục gốc của word-count-hadoop và chạy lệnh : ‘mvn clean package’

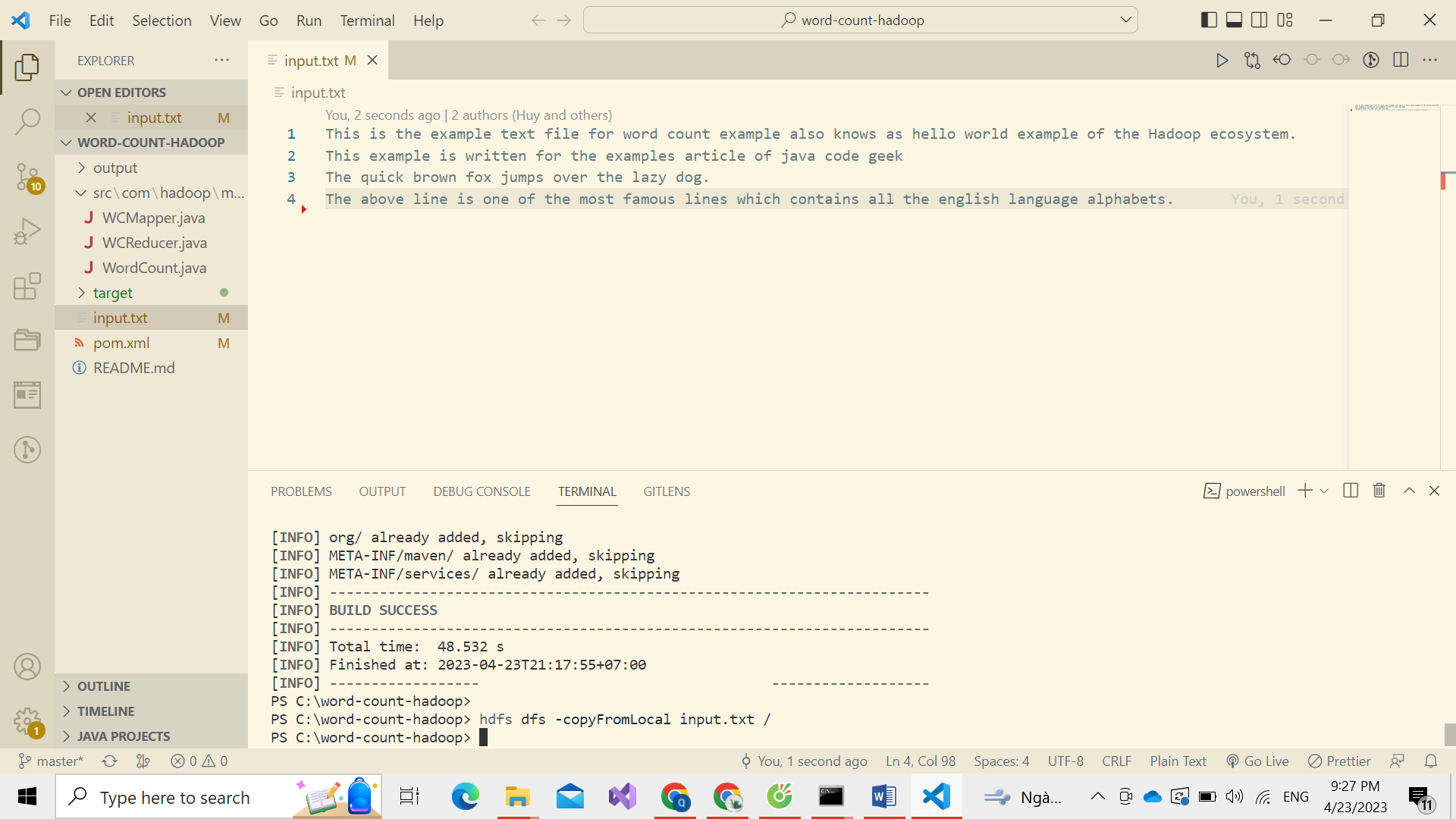


*Hình 4.3.2 Build file jar với maven*

Hiển thị BUILD SUCCESS là thành công

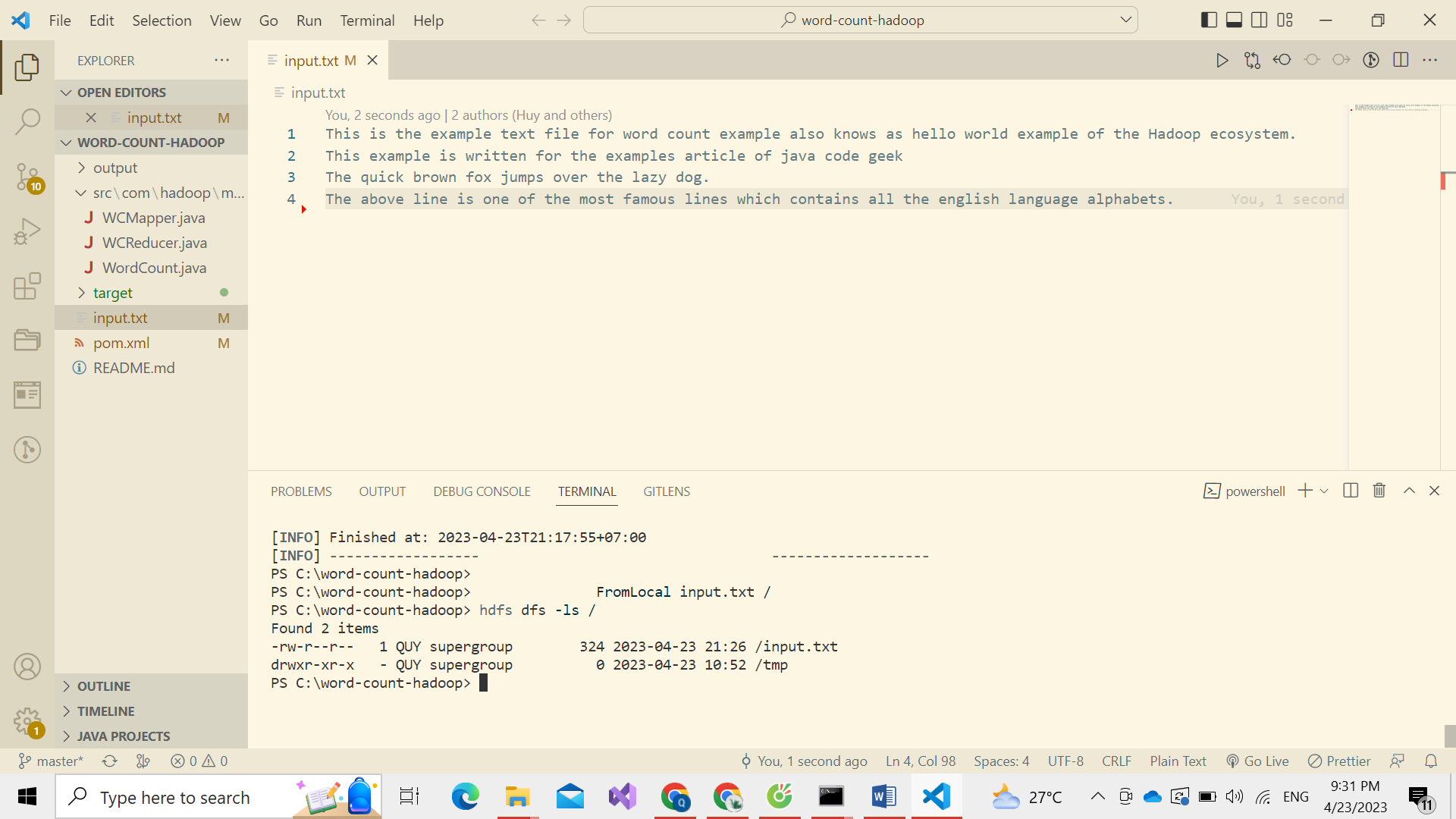
Bước 3 : Chạy file jar WordCount trên Hadoop

Trước khi chạy thì ta phải chuẩn bị dữ liệu đầu vào trước, trong Project mình có để một file input.txt chúng ta sài file này làm file đầu vào, bây giờ phải đẩy nó lên HDFS bằng câu lệnh sau : ‘hdfs dfs -copyFromLocal input.txt /’



*Hình 4.3.3 Đẩy lên HDFS*

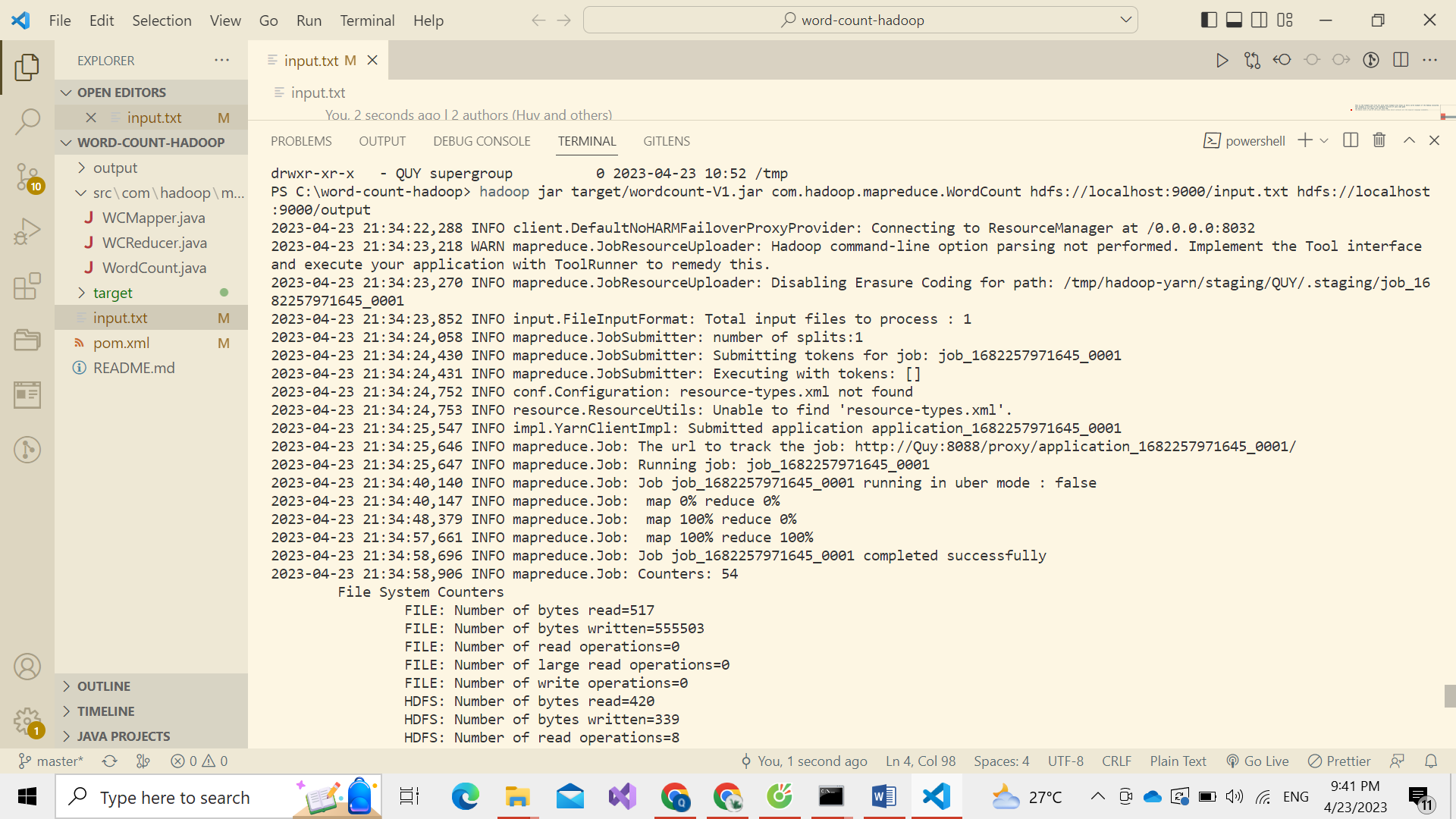
Bước 4 : Kiểm tra lại xem đã thành công chưa bằng cách kiểm tra các file và thư mục trong HDFS bằng lệnh : ‘hdfs dfs -ls /’



*Hình 4.3.4 Hiển thị file và thư mục*

Chúng ta cũng có thể xem trực tiếp công cụ UI tại đường dẫn : ‘localhost:9870’

Bước 5 : Chạy file jar WordCount



*Hình 4.3.5 Chạy chương trình*

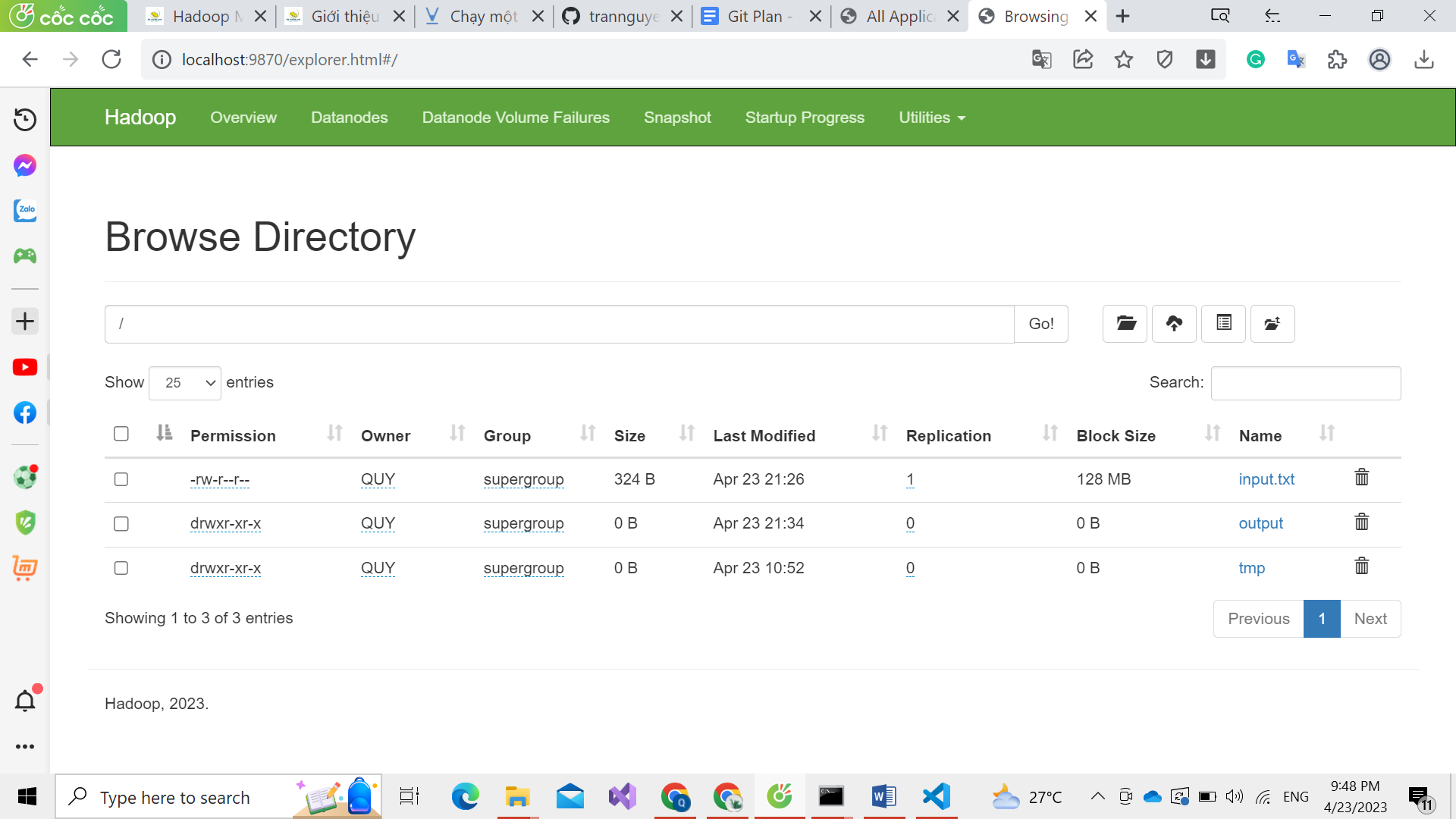
hadoop jar target/wordcount-V1.jar com.hadoop.mapreduce.WordCount hdfs://localhost:9000/input.txt hdfs://localhost:9000/output

Trong đó :

* ‘target/wordcount-V1.jar’ là đường dẫn tới file jar WordCount
* ‘com.hadoop.mapreduce.WordCount’ là đường dẫn tới vị trí ‘Class’ chứa hàm main
* ‘hdfs://localhost:9000/input.txt’ là đường dẫn file đầu vào trong HDFS[1]
* ‘hdfs://localhost:9000/output’ là đường dẫn của thư mục đầu ra[2]

[1] và [2] là 2 tham số mà mình đã chỉ định trong Project, trong các Project có thể set các tham số tùy thích, có thể fix cứng đầu ra và chỉ nhận 1 tham số đầu vào là input

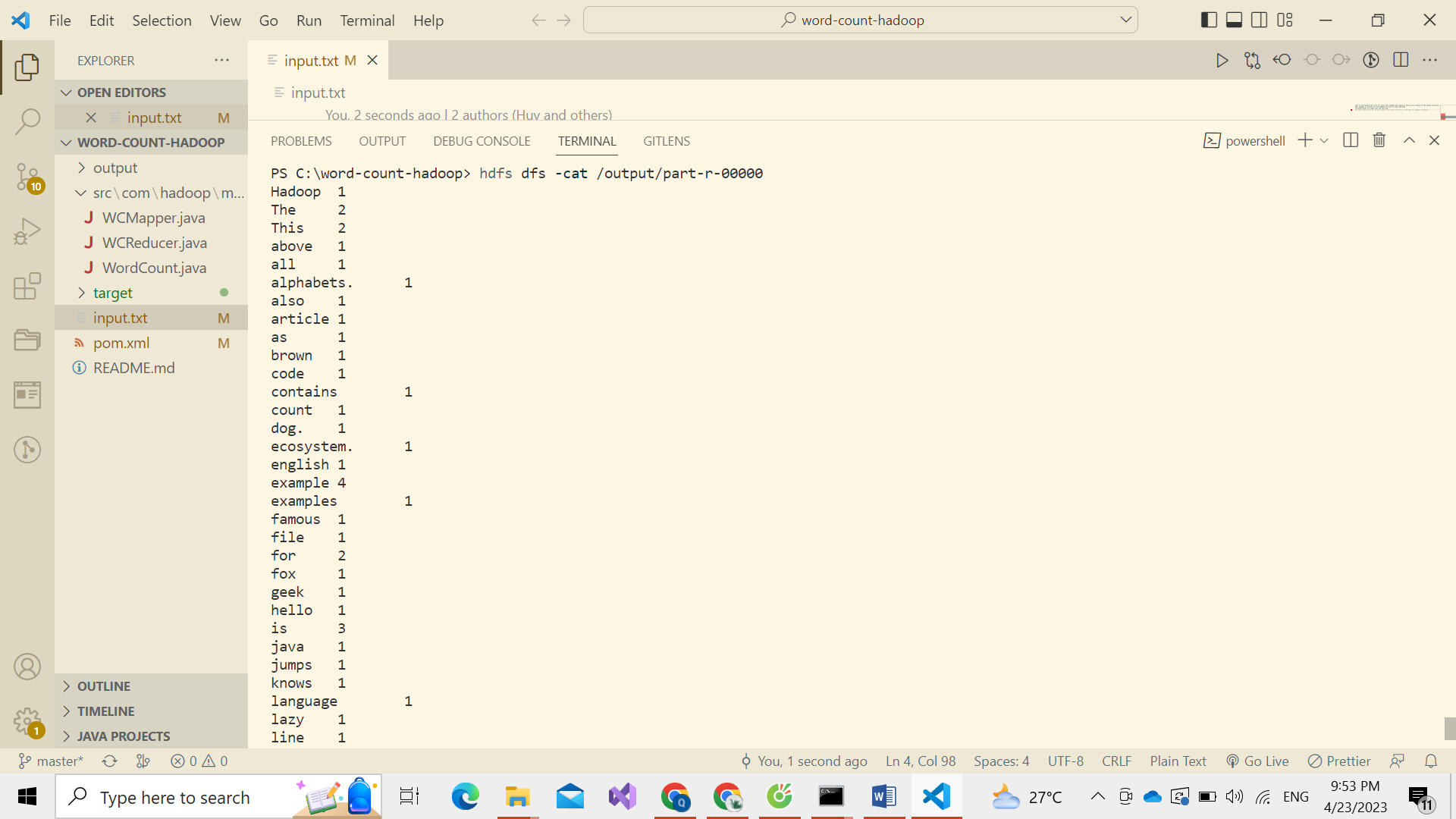
Sau khi chạy xong một thư mục output sẽ được sinh ra trên DHFS, truy cập vào ‘localhost:9870’ đây là kết quả :



*Hình 4.3.6 Hiển thị thư mục trên local*

Bước 6 : Kiểm tra kết quả

Chạy lệnh ‘hdfs dfs -cat /output/part-r-00000’ và hiển thị :



*Hình 4.3.7 Kết quả*

### 4.4 Thiết kế kiểm thử

### 4.4.1 Các ca kiểm thử (Test Cases)

Các ca kiểm thử (Test Cases) bao gồm các phần như sau:

Kiểm thử tính năng: để đảm bảo rằng ứng dụng của bạn hoạt động đúng như đã thiết kế và đáp ứng đầy đủ yêu cầu của người dùng. Các ca kiểm thử này bao gồm:

-Kiểm thử tính năng chính của ứng dụng, bao gồm các ca kiểm thử cơ bản và nâng cao để đảm bảo rằng ứng dụng hoạt động như mong muốn.

-Kiểm tra tính thích ứng của ứng dụng bằng cách thực hiện các ca kiểm thử tương tác như thay đổi kích thước hoặc cấu trúc của dữ liệu đầu vào.

-Kiểm thử tính bảo mật của ứng dụng để đảm bảo rằng nó không có lỗ hổng bảo mật phát sinh.

Kiểm tra khả năng xử lý đồng thời: Để đảm bảo rằng hệ thống Apache Hadoop có khả năng xử lý nhiều tác vụ đồng thời, kiểm thử nên tập trung vào một số tính năng như MapReduce và YARN.

Kiểm tra khả năng mở rộng: Hệ thống Apache Hadoop được tạo ra để có khả năng mở rộng tốt. Tuy nhiên, để đảm bảo tính đúng đắn của ứng dụng, các ca kiểm thử cần đảm bảo rằng hệ thống có thể mở rộng một cách hiệu quả.

Kiểm tra tính đối tượng: Hadoop là một hệ thống lưu trữ và xử lý dữ liệu đối tượng. Việc kiểm thử tính đối tượng cho phép đảm bảo hệ thống lưu trữ và xử lý dữ liệu chuẩn xác.

Kiểm tra tính bảo mật: Hadoop là một hệ thống rất mạnh mẽ và có thể lưu trữ và xử lý dữ liệu quan trọng. Tuy nhiên, để đảm bảo tính bảo mật, các ca kiểm thử nên kiểm tra tính an toàn của hệ thống bằng cách sử dụng các phương pháp mã hóa dữ liệu.

Kiểm tra khả năng phục hồi: Hadoop cung cấp khả năng phục hồi nhanh chóng nếu có sự cố xảy ra. Các ca kiểm thử nên kiểm tra khả năng phục hồi của hệ thống bằng cách thực hiện các ca kiểm thử về tình huống thất bại của phần cứng hoặc mạng.

### 4.4.2 Tích hợp kiểm thử.

Giảm đáng kể chi phí và thời gian: Tích hợp kiểm thử vào quy trình phát triển ứng dụng từ đầu giúp việc phát hiện và sửa các lỗi trở nên nhanh chóng và dễ dàng hơn. Điều này giúp giảm thời gian và chi phí trong việc phát triển và triển khai ứng dụng.

Đảm bảo chất lượng và hiệu suất của ứng dụng: Tích hợp kiểm thử sớm vào quy trình phát triển giúp đảm bảo rằng ứng dụng của bạn sẽ hoạt động đúng như mong đợi và đảm bảo hiệu suất của ứng dụng trên Hadoop. Điều này giúp giảm thiểu các sự cố và vấn đề phát sinh trong quá trình triển khai ứng dụng.

Độ tin cậy cao hơn: Tích hợp kiểm thử vào quy trình phát triển giúp đảm bảo rằng ứng dụng của bạn sẽ đáp ứng được yêu cầu của người dùng và hoạt động ổn định trên Hadoop. Đồng thời, tích hợp kiểm thử còn giúp kiểm tra và giải quyết các vấn đề tiềm ẩn sớm, giúp đảm bảo độ tin cậy cao hơn cho ứng dụng.

Tăng tốc độ phát triển ứng dụng: Tích hợp kiểm thử vào quy trình phát triển cũng giúp tăng tốc độ phát triển ứng dụng bằng cách giảm đáng kể thời gian kiểm thử ứng dụng trước khi triển khai. Điều này giúp tiết kiệm thời gian và giúp bạn tập trung vào nâng cao tính năng và hiệu suất của ứng dụng.

# KẾT LUẬN

Xây dựng một ứng dụng phân tích dữ liệu sử dụng Apache Hadoop là một đề tài rất thú vị và hấp dẫn. Với sự phát triển của công nghệ và xu hướng dữ liệu lớn ngày càng tăng, việc tìm hiểu và áp dụng Apache Hadoop vào phân tích dữ liệu là rất cần thiết. Trong bài thuyết trình này, chúng ta đã cùng nhau tìm hiểu về Apache Hadoop, các công cụ và kỹ thuật để xây dựng ứng dụng phân tích dữ liệu trên cụm Hadoop. Chúng ta đã cùng nhau tìm hiểu về các bước cần thiết để triển khai và kiểm tra hiệu suất của ứng dụng, từ đó giúp tối ưu hóa hoạt động của ứng dụng trên cụm Hadoop. Nhờ vậy, chúng ta có thể áp dụng Apache Hadoop để xây dựng các ứng dụng phân tích dữ liệu với khả năng xử lý lớn và hiệu suất cao, đáp ứng nhu cầu của doanh nghiệp và giúp cho việc ra quyết định dựa trên dữ liệu trở nên chính xác và nhanh chóng hơn.

Em xin chân thành cảm ơn sự giúp đỡ nhiệt tình của thầy Trần Đăng Hoan đã hỗ trợ và gợi ý tài liệu cho nhóm em về bài luận lần này. Trong quá trình thực hiện đề tài, không thể tránh khỏi những thiếu sót nên em rất mong được sự đóng góp của thầy để bài luận của nhóm em được hoàn thiện tốt hơn.

# TÀI LIỆU THAM KHẢO

[1]Wikipedia.20/04/2023

[hthttps://vi.wikipedia.org/wiki/Apache\_Hadoop](https://vi.wikipedia.org/wiki/Apache_Hadoop)

[2]Hadoop.

[Apache Hadoop Streaming](https://hadoop.apache.org/docs/r1.2.1/streaming.html)

[3]Cloud Geeks Vietnam.

<https://cloudgeeks.net/tim-hieu-ve-kien-truc-hadoop-ecosystem/#Tham_khao>

[4] <https://spark.apache.org/docs/latest/submitting-applications.html>

[5] <https://viblo.asia/p/chay-mot-file-jar-tren-hadoop-ByEZkaX25Q0>

[6] <https://nguyenlediep.com/chay-wordcount-trong-hadoop>