

FACULTY OF INFORMATION TECHNOLOGY

UNIVERSITY OF SCIENCE, HO CHI MINH CITY, VIET NAM



fit@hcmus

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

TỔNG HỢP CÁC BÀI LAB MÔN ĐIỆN TOÁN ĐÁM MÂY

A. THÔNG TIN CHUNG

Học viên thực hiện: Nguyễn Thị Ngọc Trâm

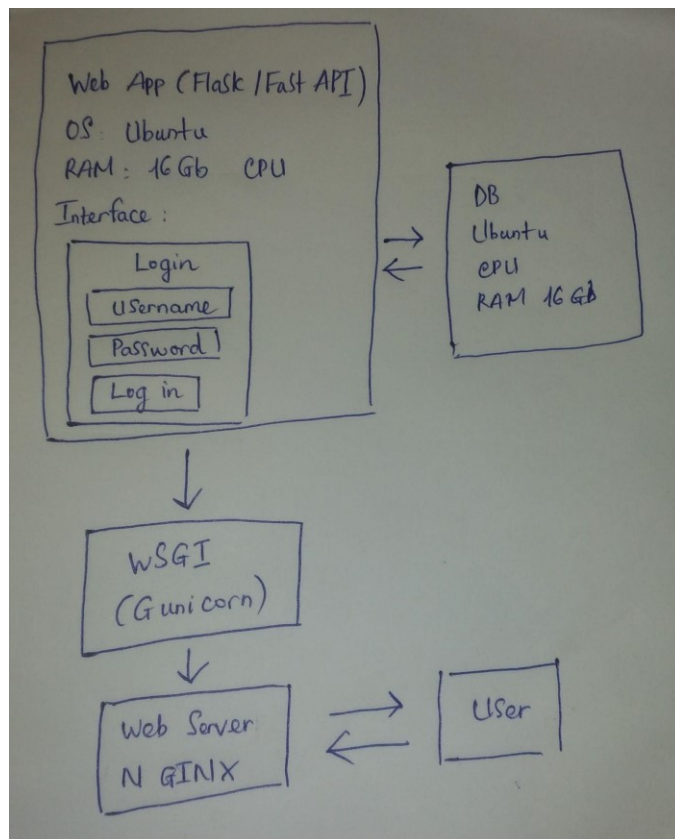
Mã số học viên: 21C11036

Khóa: 31 Ngành: Khoa học máy tính

B. NỘI DUNG BÁO CÁO

1. LAB 1: Amazon Web Services Amazon EC2

Deployment architecture



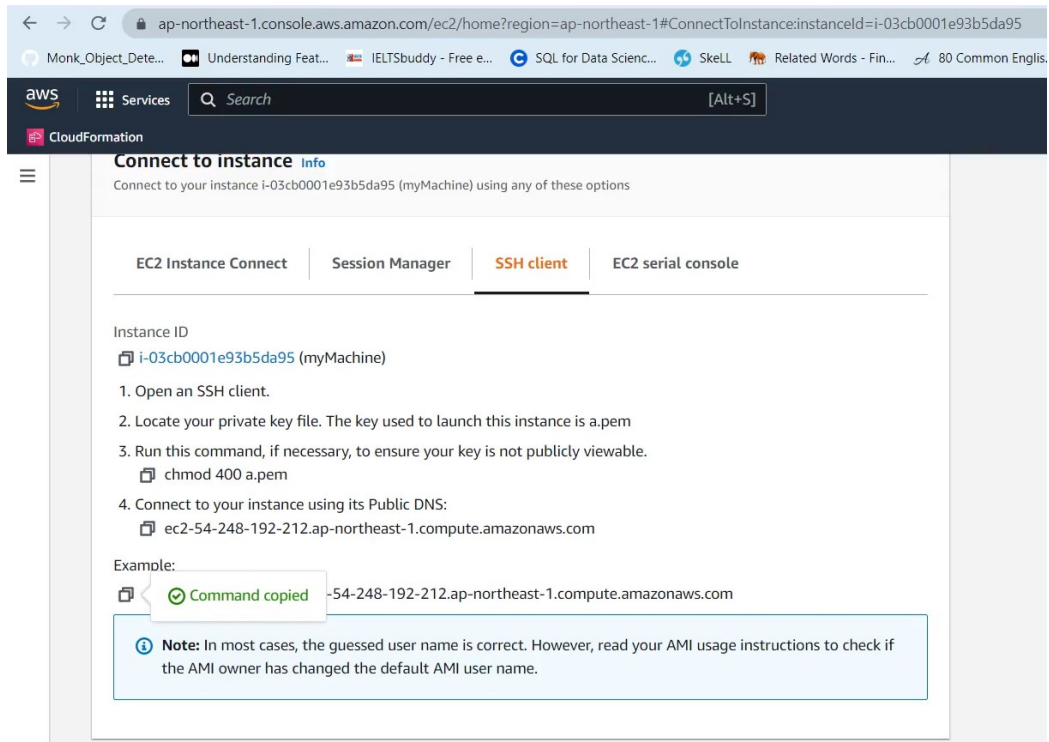
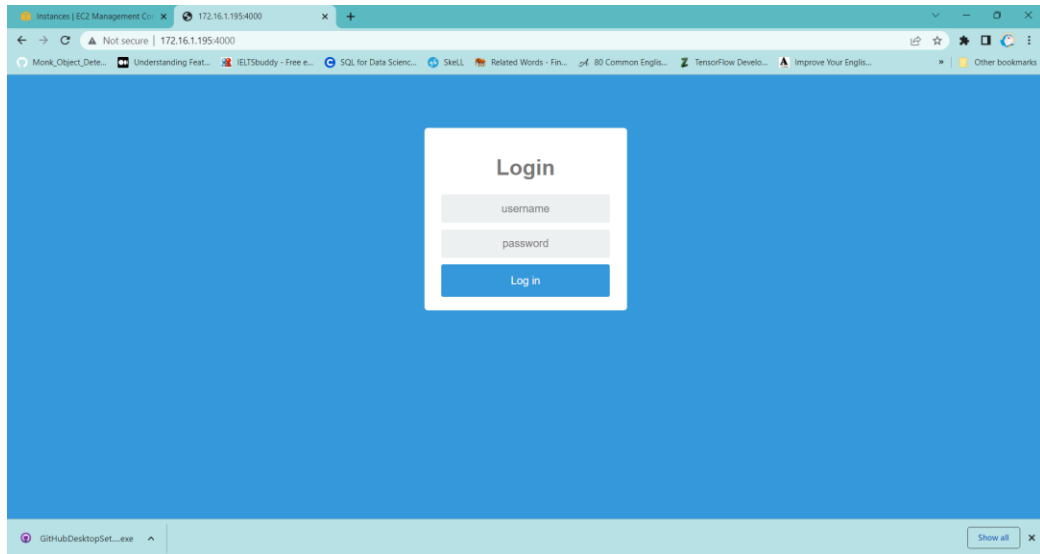
Demo video:

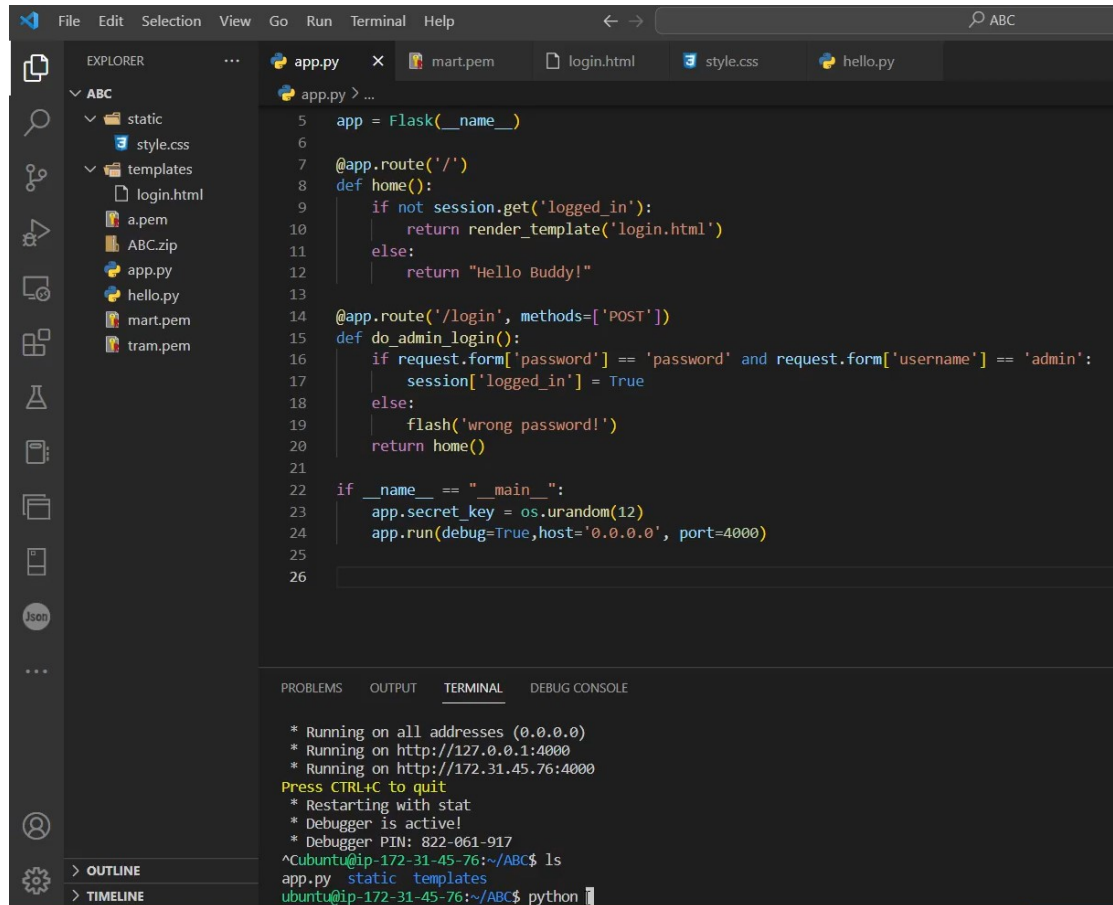
<https://drive.google.com/file/d/1Onogbb95GKzzyUTXMHONt4jtKuBp6D3u/view?usp=sharing>

Cloud provider URL

FACULTY OF INFORMATION TECHNOLOGY

UNIVERSITY OF SCIENCE, HO CHI MINH CITY, VIET NAM





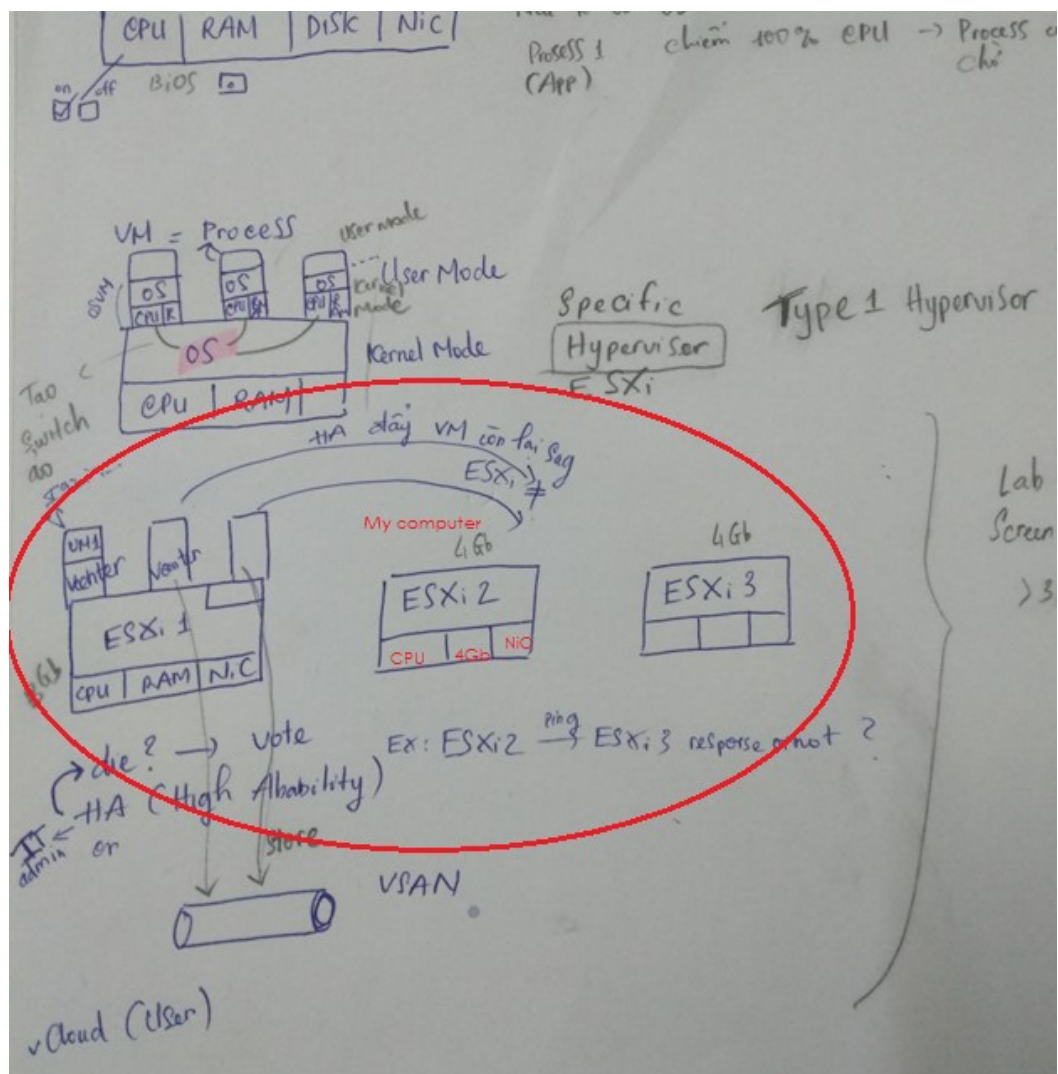
The image shows a Visual Studio Code editor window with a dark theme. The Explorer sidebar on the left shows a project structure for 'ABC' with folders 'static' and 'templates', and files 'style.css', 'login.html', 'a.pem', 'ABC.zip', 'app.py', 'hello.py', 'mart.pem', and 'tram.pem'. The main editor area displays the code for 'app.py'. The code is a Flask application with a home route and a login route. The terminal at the bottom shows the output of running the application, including the Flask version, the running addresses (0.0.0.0, 127.0.0.1, and 172.31.45.76), and the command to run the application (python app.py).

```
5 app = Flask(__name__)
6
7 @app.route('/')
8 def home():
9     if not session.get('logged_in'):
10         return render_template('login.html')
11     else:
12         return "Hello Buddy!"
13
14 @app.route('/login', methods=['POST'])
15 def do_admin_login():
16     if request.form['password'] == 'password' and request.form['username'] == 'admin':
17         session['logged_in'] = True
18     else:
19         flash('wrong password!')
20     return home()
21
22 if __name__ == "__main__":
23     app.secret_key = os.urandom(12)
24     app.run(debug=True, host='0.0.0.0', port=4000)
25
26
```

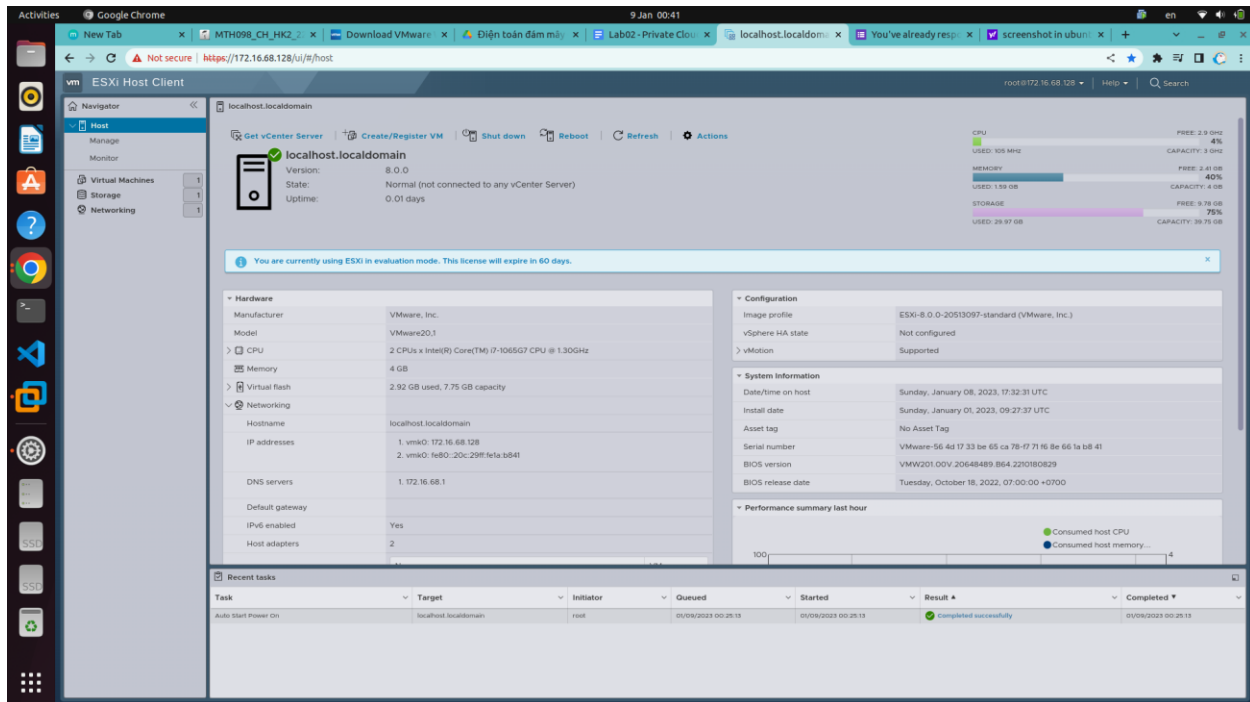
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:4000
* Running on http://172.31.45.76:4000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 822-061-917
^Cubuntu@ip-172-31-45-76:~/ABC$ ls
app.py  static  templates
ubuntu@ip-172-31-45-76:~/ABC$ python
```

Datacenter Architecture ESXi, Vcenter



Screenshot



3. LAB 3: Platform as a Service Lab

PaaS provider URL: <https://developer.salesforce.com/>

Application URL for admin: <https://hcmus2-dev-ed.develop.lightning.force.com/lightning/r/User/0052w000000Bx24iAAB/view>

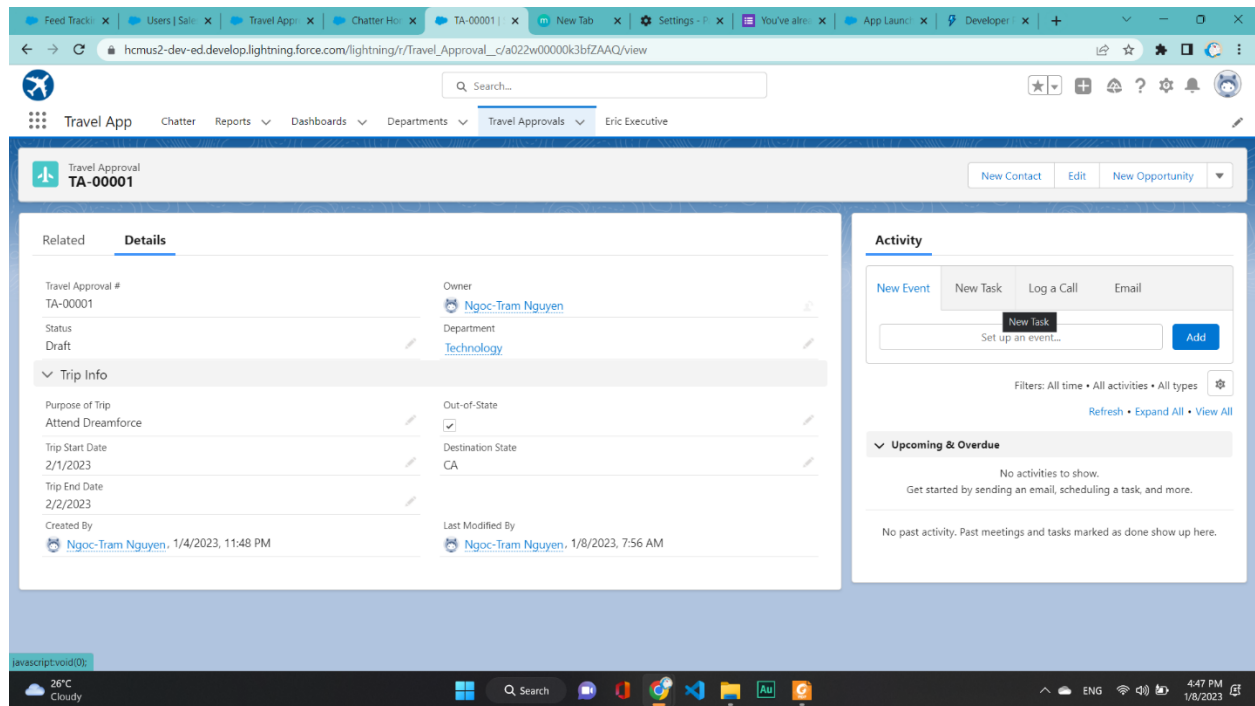
Admin email or username: tramnt.139@gmail.com

Admin password: ngocTram1309\$

Screenshot of admin UI:

FACULTY OF INFORMATION TECHNOLOGY

UNIVERSITY OF SCIENCE, HO CHI MINH CITY, VIET NAM



Application URL for user: https://hcmus2-dev-ed.develop.lightning.force.com/lightning/r/Travel_Approval__c/a022w00000k3bfZAAQ/view

User Email or username:

21c11036@student.hcmus.edu.vn

User password: ngocTram13

4. LAB 4: Cloud Computing - Storage in the Cloud: NAS + GFS

Install NAS

RAM available: 4Gb

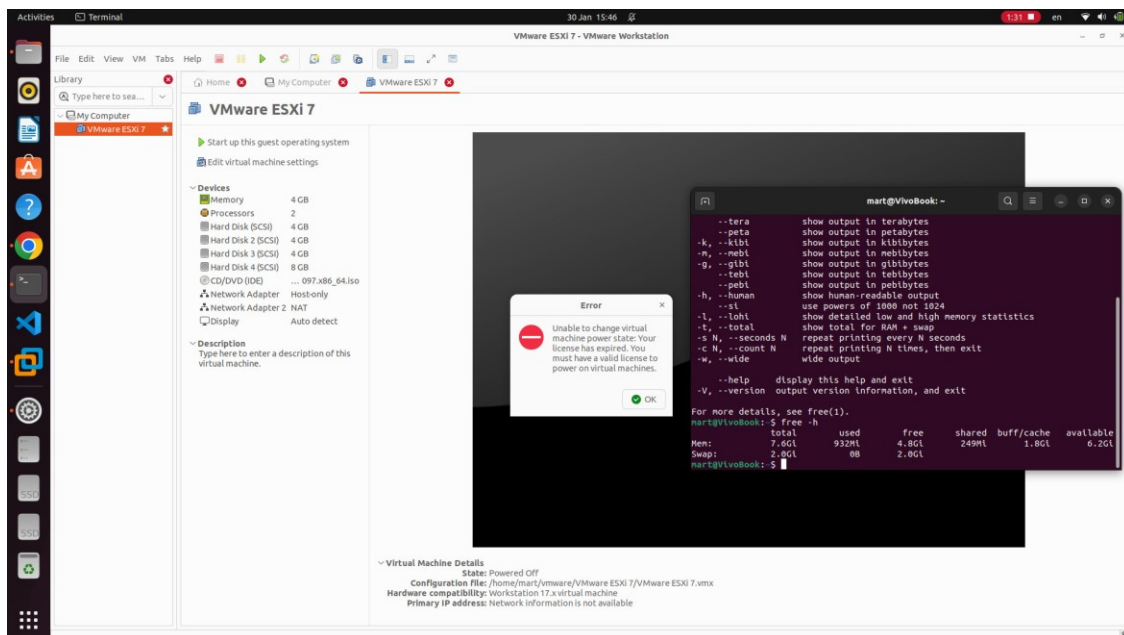
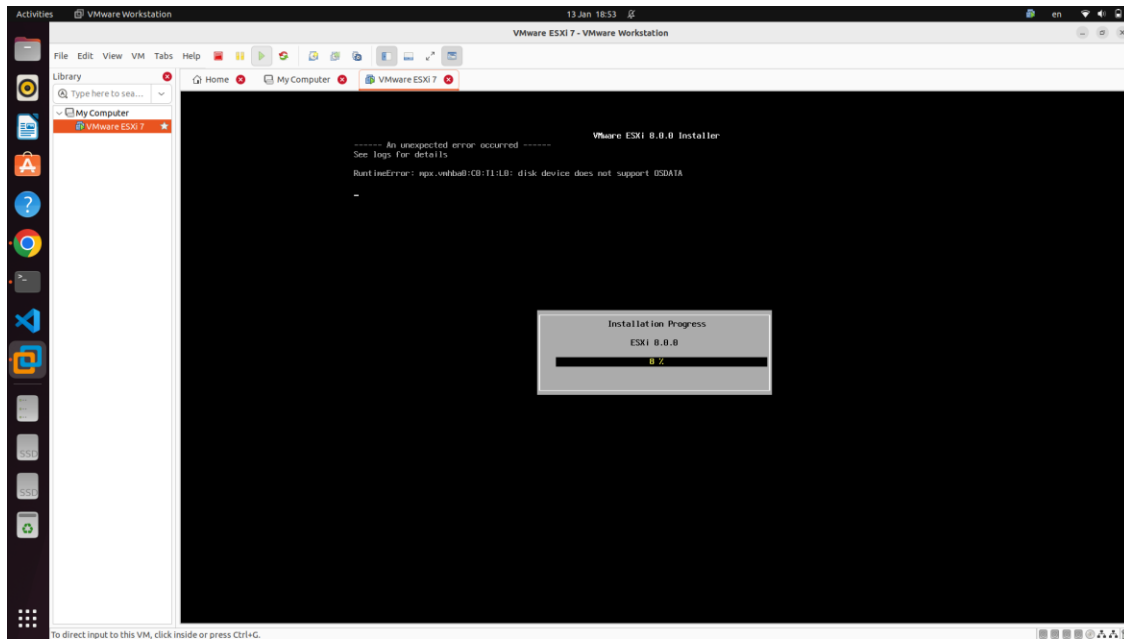
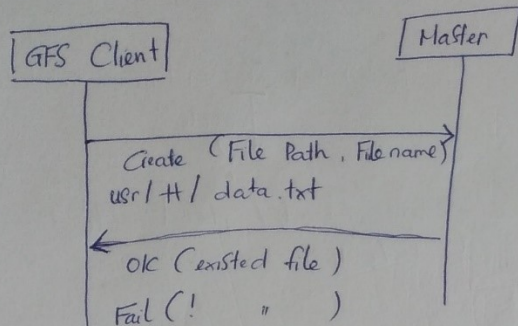


Figure 1:

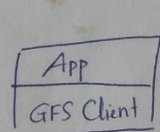
< Sequence Diagram >

① Create File



Chunk Mapping

Chunk ID	Chunk Server
abc	192. 0.10 0.11 0.12



Master

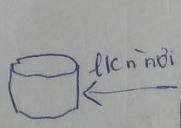
Path	Lock
usr / + / data.txt	True / False

/usr
/usr / + /
/usr / + / data.txt

Chunk Table

ID		
1	abc	64KB
2	def	
3		

Data



Historical Record

Date	Action	Data	Process
	Create file	usr / + / data.txt	✓ OK

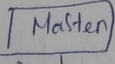
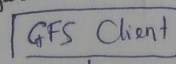
Save in disk

Data Structure : (S/Glow) Tree

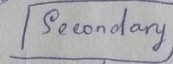
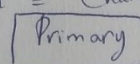
Choose Chunk (free Space)



Figure 2



Choose Chunk to write
 $164 + 1 = \text{Chunk Index}$



Chunk Server

write (Data, (Chunk index) location)

check đủ HB để ghi ?

Chunk table chọn (10, 11, 12) primary

(GOS)

Send Data

Send data

Data, ID 192...

Store in each

confirm write

if master lose communicate Client

ping xin them GOS

Describe a ***GFS read*** using a sequence diagram:

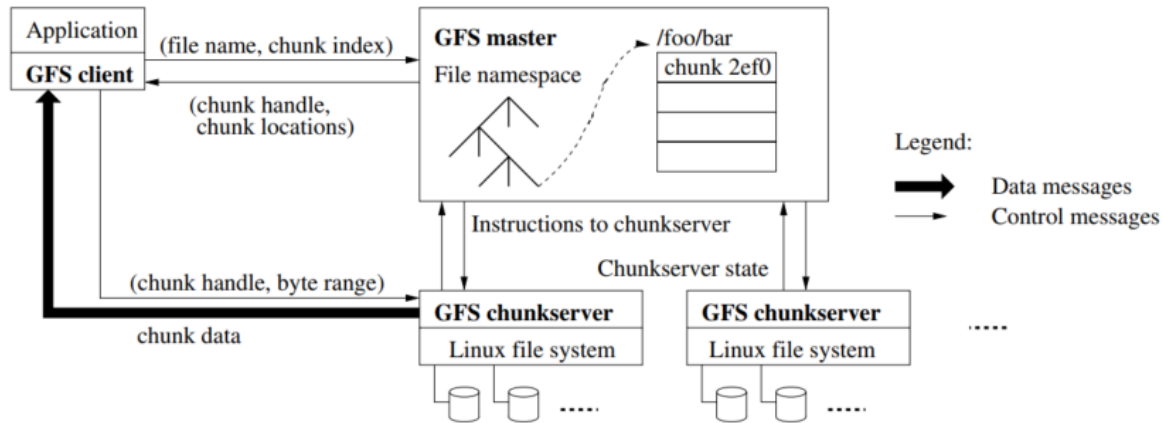


Figure 1: GFS Architecture

1. The client first communicates with the master, sending it a request containing ***the file name and the chunk index***. The client derives the chunk index from a combination of the file name and the byte offset the application wants to read from.
2. The master replies with the corresponding ***chunk handle and the location*** of its replicas.
3. The client caches this information using the file name and chunk index as the key.
4. The client then sends a request to one of the replicas specified by the client, usually the one closest to it, specifying the ***chunk handle and the byte range*** for the requested data.
5. By caching the information from the master, further reads to the same chunk do not require any more client-master interactions until the cached information expires or the file is reopened.

Describe a ***GFS write*** using a sequence diagram

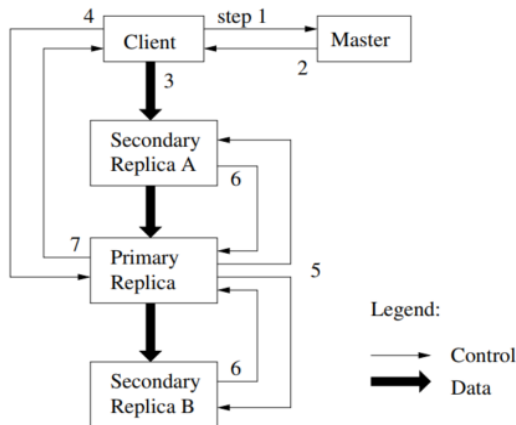


Figure 2: Write Control and Data Flow

3. The client pushes the data to all the chunk servers, not necessarily to the primary first. The servers will initially store this data in an internal LRU buffer cache until the data is used.

4. Once the client receives the acknowledgement that this data has been pushed successfully, it sends the write request to the primary chunkserver. The primary decides what serial order to apply the mutations in and applies them to the chunk.

5. After applying the mutations, the primary forwards the write request and the serial number order to all the secondaries for them to apply in the same order.

6. All secondaries reply to the primary once they have completed the operation.

7. The primary replies to the client, indicate whether the operation was a success or an error. Note:

- + If the write succeeds at the primary but fails at any of the secondaries, we'll have an inconsistent state and an error is returned to the client.
- + The client can retry steps 3 through 7.

Describe a **GFS append** using a sequence diagram

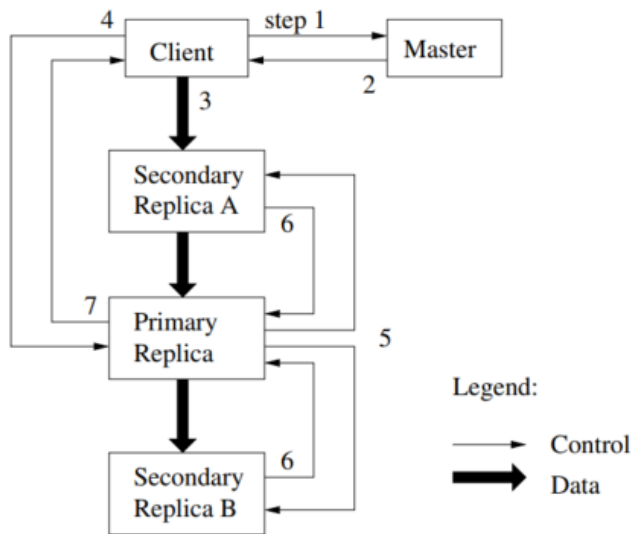


Figure 2: Write Control and Data Flow

1. The system interactions for record appends are largely the same as discussed for writes, with the following exceptions:
2. In step 4, the primary first checks to see if appending the record to the current chunk would exceed the maximum size of 64MB. If so, the primary pads the chunk, notifies the secondaries to the same, and then tells the client to retry the request on the next chunk.
3. If the record append fails on any of the replicas, the client must retry the operation. As discussed in the Consistency section, this means that replicas of the same chunk may contain duplicates.
4. A record append is successful only when the data has been written at the same offset on all the replicas of a chunk.

<https://timilearning.com/posts/mit-6.824/lecture-3-gfs/#single-master>

<https://www.linkedin.com/pulse/gfs-google-file-system-amit-kumar/>

<https://computer.howstuffworks.com/internet/basics/google-file-system.htm>

<https://cs.stanford.edu/~matei/courses/2015/6.S897/slides/gfs.pdf>

5. LAB 5: Cloud Computing - Amazon Aurora

Connect to Aurora DB

```
C:\Users\Ngoc Tram\Downloads>mysql -h database-1.cluster-cupg5wk6z5nv.ap-northeast-1.rds.amazonaws.com -P 3306 -u admin -p
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 385
Server version: 8.0.23 Source distribution

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

Use DBeaver to connect and create DB in Aurora DB

The screenshot shows the DBeaver 22.3.4 interface. The main window displays a script editor with the following SQL queries:

```
update BOMON set TRUONGBM = '007' where MADM='NPT';

insert into NGUOITHAN values ('001', N'Hùng', '1990-01-14', N'Nam');
insert into NGUOITHAN values ('001', N'Thủy', '1994-08-12', N'Nữ');
insert into NGUOITHAN values ('003', N'Thu', '1996-09-03', N'Nữ');
insert into NGUOITHAN values ('003', N'Hà', '1998-03-09', N'Nữ');
insert into NGUOITHAN values ('008', N'Mam', '1991-06-05', N'Mam');
insert into NGUOITHAN values ('010', N'Nguyễn', '2006-01-14', N'Nữ');
insert into NGUOITHAN values ('007', N'Vy', '2000-02-14', N'Nữ');
insert into NGUOITHAN values ('007', N'Mai', '2003-03-26', N'Nữ');
insert into NGUOITHAN values ('009', N'An', '1996-08-19', N'Nam');

select
from KHOA
where PHONG='B11'
and NAMTI='1995';

select LUONG
from GIAOVIE
where PHAI='Nam';

select distinct LUONG
from GIAOVIE
where PHAI='Nam';
```

The bottom panel shows a data grid view for the 'LUONG' column, with values 2,000, 2,500, 2,100, and 1,500. The status bar indicates 4 rows fetched in 112ms on 2023-02-11 at 00:24:20.

Query DB

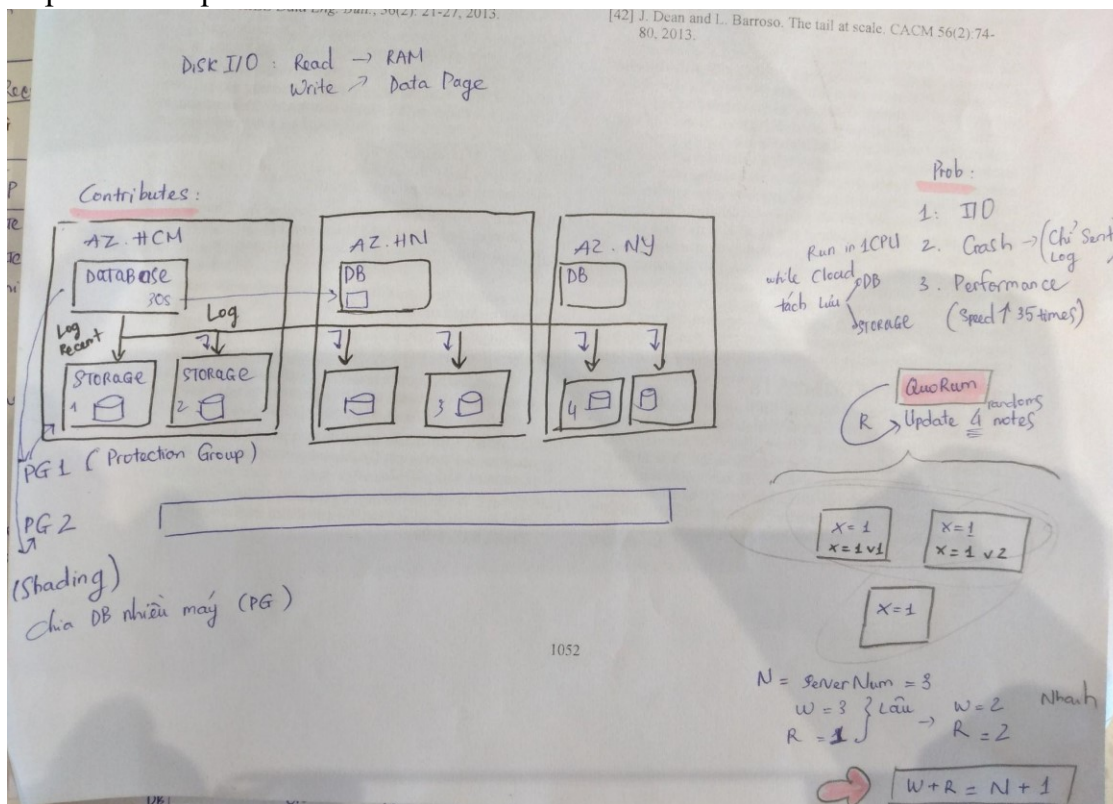
```

mysql> use QLDT;
Database changed
mysql> select LUONG from GIAOVIEN where PHAI='Nam'
-> select LUONG from GIAOVIEN where PHAI='Nam';
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version
for the right syntax to use near 'select LUONG from GIAOVIEN where PHAI='Nam'' at line 2
mysql> SELECT *
-> FROM KHOA
-> WHERE PHONG='B11'
-> AND NAMTL='1995'
-> ;
+-----+-----+-----+-----+-----+-----+
| MAKHOA | TENKHOA | NAMTL | PHONG | DIENTHOAI | TRUONGKHOA | NGAYNHANCHUC |
+-----+-----+-----+-----+-----+-----+
| CNTT  | Công nghệ thông tin | 1995 | B11   | 0838123456 | 002         | 2005-02-20 00:00:00 |
+-----+-----+-----+-----+-----+-----+
1 row in set (0.11 sec)

mysql> SELECT LUONG
-> FROM GIAOVIEN
-> WHERE PHAI='Nam';
+-----+
| LUONG |
+-----+
| 2000  |
| 2500  |
| 2100  |
| 2000  |
| 1500  |
+-----+

```

Explain how quorums work.



Describe how replication writes work in a mirrored MySQL database using a sequence diagram.

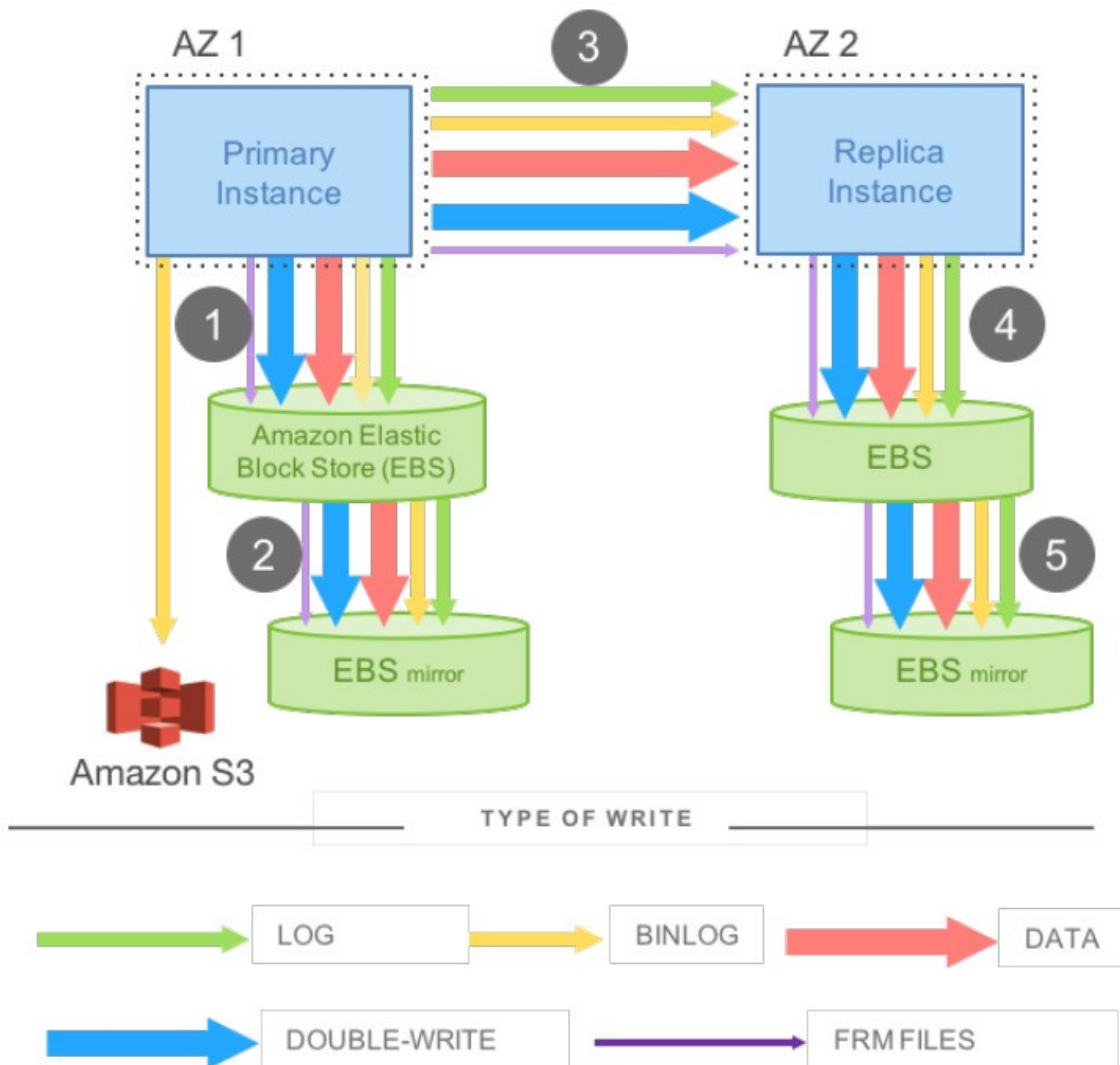


Figure 2: Network IO in mirrored MySQL

6. LAB 6: MapReduce

The screenshot shows a web browser window with a YouTube video titled "Apache Hadoop Distribution" and a Command Prompt window. The video player shows a timestamp of 2023-02-21 07:39:33,351. The Command Prompt window displays the following output:

```

at org.apache.hadoop.hdfs.server.namenode.FSImage.saveFSImageInAllDir
at org.apache.hadoop.hdfs.server.namenode.FSImage.format(FSImage.jav
at org.apache.hadoop.hdfs.server.namenode.NameNode.format(FSImage.jav
at org.apache.hadoop.hdfs.server.namenode.NameNode.createNameNode(Na
at org.apache.hadoop.hdfs.server.namenode.NameNode.main(NameNode.jav
2023-02-21 07:39:33,351 INFO namenode.FSNamesystem: Stopping services starte
2023-02-21 07:39:33,351 INFO namenode.FSNamesystem: Stopping services starte
2023-02-21 07:39:33,352 ERROR namenode.NameNode: Failed to start namenode.
java.io.IOException: No image directories available!
at org.apache.hadoop.hdfs.server.namenode.FSImage.saveFSImageInAllDir
at org.apache.hadoop.hdfs.server.namenode.FSImage.saveFSImageInAllDir
at org.apache.hadoop.hdfs.server.namenode.FSImage.format(FSImage.jav
at org.apache.hadoop.hdfs.server.namenode.NameNode.format(FSImage.jav
at org.apache.hadoop.hdfs.server.namenode.NameNode.createNameNode(Na
at org.apache.hadoop.hdfs.server.namenode.NameNode.main(NameNode.jav
2023-02-21 07:39:33,354 INFO util.ExitUtil: Exiting with status 1: java.io.I
2023-02-21 07:39:33,357 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****
SHUTDOWN_MSG: Shutting down NameNode at Casper/192.168.124.1
*****/
C:\Users\Ngoc Tram>cd ../
C:\Users>cd C:\hadoop-3.3.4\sbin
C:\hadoop-3.3.4\sbin>start-all.cmd
This script is Deprecated. Instead use start-dfs.cmd and start-yarn.cmd
starting yarn daemons
C:\hadoop-3.3.4\sbin>hadoop version
Hadoop 3.3.4
Source code repository https://github.com/apache/hadoop.git -r a585a73c3e02ac62350c136643a5e7f6095a3dbb
Compiled by stevel on 2022-07-29T12:32Z
Compiled with protoc 3.7.1
From source with checksum fb9dd8918a7b8a5b430d61af858f6ec
This command was run using /C:/hadoop-3.3.4/share/hadoop/common/hadoop-common-3.3.4.jar
C:\hadoop-3.3.4\sbin>

```

The screenshot shows a Command Prompt window with the following output:

```

java.io.IOException: No image directories available!
at org.apache.hadoop.hdfs.server.namenode.FSImage.saveFSImageInAllDir
at org.apache.hadoop.hdfs.server.namenode.FSImage.saveFSImageInAllDir
at org.apache.hadoop.hdfs.server.namenode.FSImage.format(FSImage.java:191)
at org.apache.hadoop.hdfs.server.namenode.NameNode.format(NameNode.java:1278)
at org.apache.hadoop.hdfs.server.namenode.NameNode.createNameNode(NameNode.java:1726)
at org.apache.hadoop.hdfs.server.namenode.NameNode.main(NameNode.java:1834)
2023-02-21 07:39:33,354 INFO util.ExitUtil: Exiting with status 1: java.io.IOException: No image directories available!
2023-02-21 07:39:33,357 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****
SHUTDOWN_MSG: Shutting down NameNode at Casper/192.168.124.1
*****/
C:\Users\Ngoc Tram>cd ../
C:\Users>cd C:\hadoop-3.3.4\sbin
C:\hadoop-3.3.4\sbin>start-all.cmd
This script is Deprecated. Instead use start-dfs.cmd and start-yarn.cmd
starting yarn daemons
C:\hadoop-3.3.4\sbin>hadoop version
Hadoop 3.3.4
Source code repository https://github.com/apache/hadoop.git -r a585a73c3e02ac62350c136643a5e7f6095a3dbb
Compiled by stevel on 2022-07-29T12:32Z
Compiled with protoc 3.7.1
From source with checksum fb9dd8918a7b8a5b430d61af858f6ec
This command was run using /C:/hadoop-3.3.4/share/hadoop/common/hadoop-common-3.3.4.jar
C:\hadoop-3.3.4\sbin>

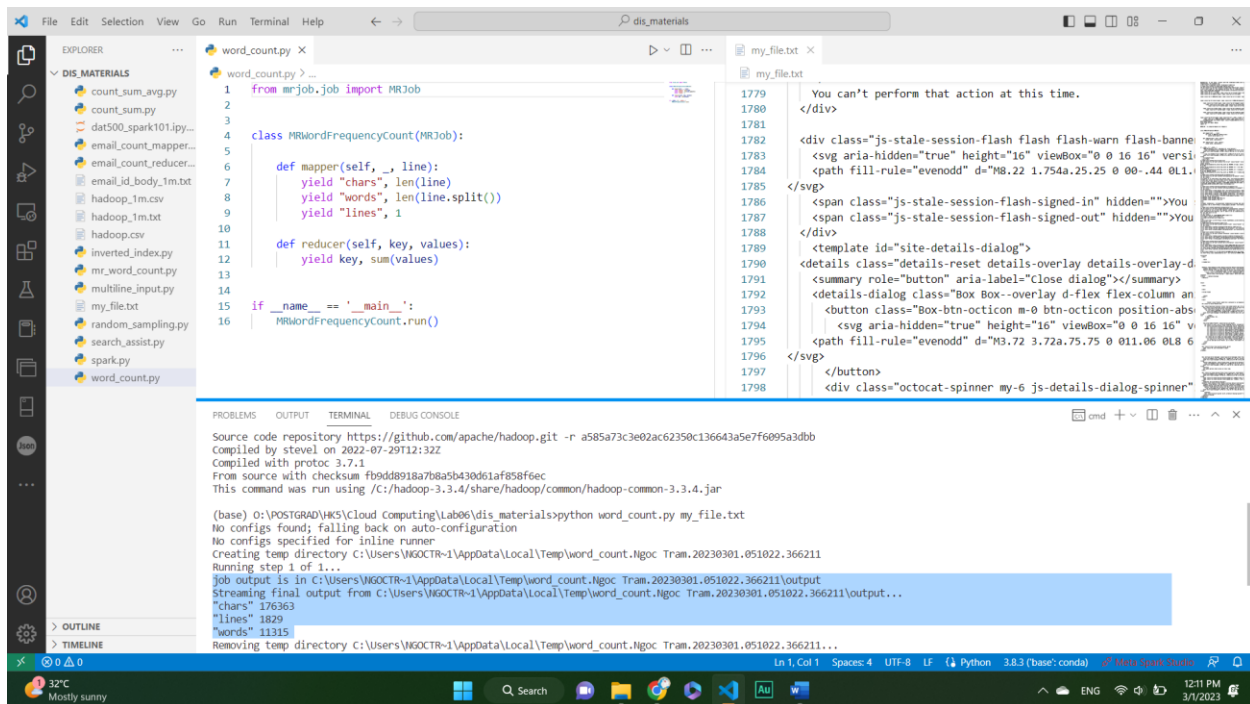
```



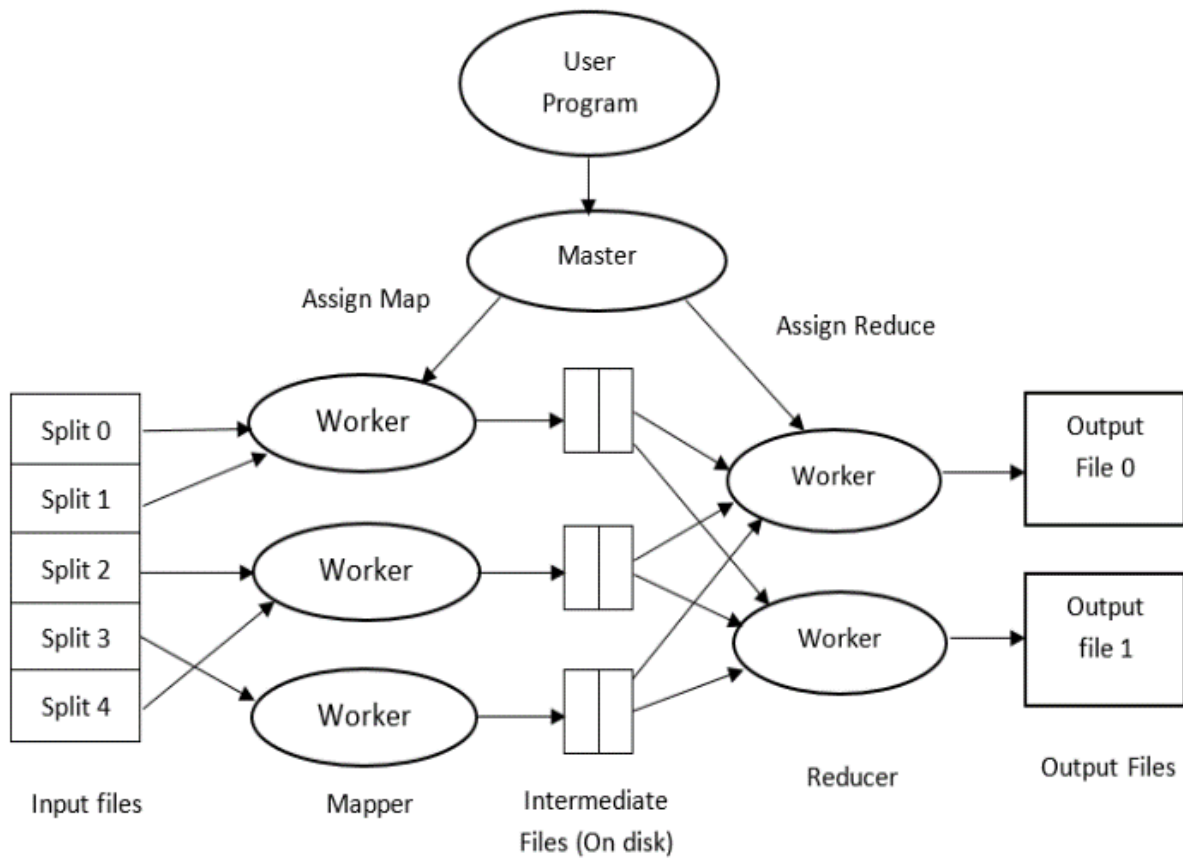
```
(base) O:\POSTGRAD\HK5\Cloud Computing\Lab06\dis_materials>hadoop version
Hadoop 3.3.4
Source code repository https://github.com/apache/hadoop.git -r a585a73c3e02ac62350c136643a5e7f6095a3dbb
Compiled by stevel on 2022-07-29T12:32Z
Compiled with protoc 3.7.1
From source with checksum fb9dd8918a7b8a5b430d61af858f6ec
This command was run using /C:/hadoop-3.3.4/share/hadoop/common/hadoop-common-3.3.4.jar
```

WordCount + myFile

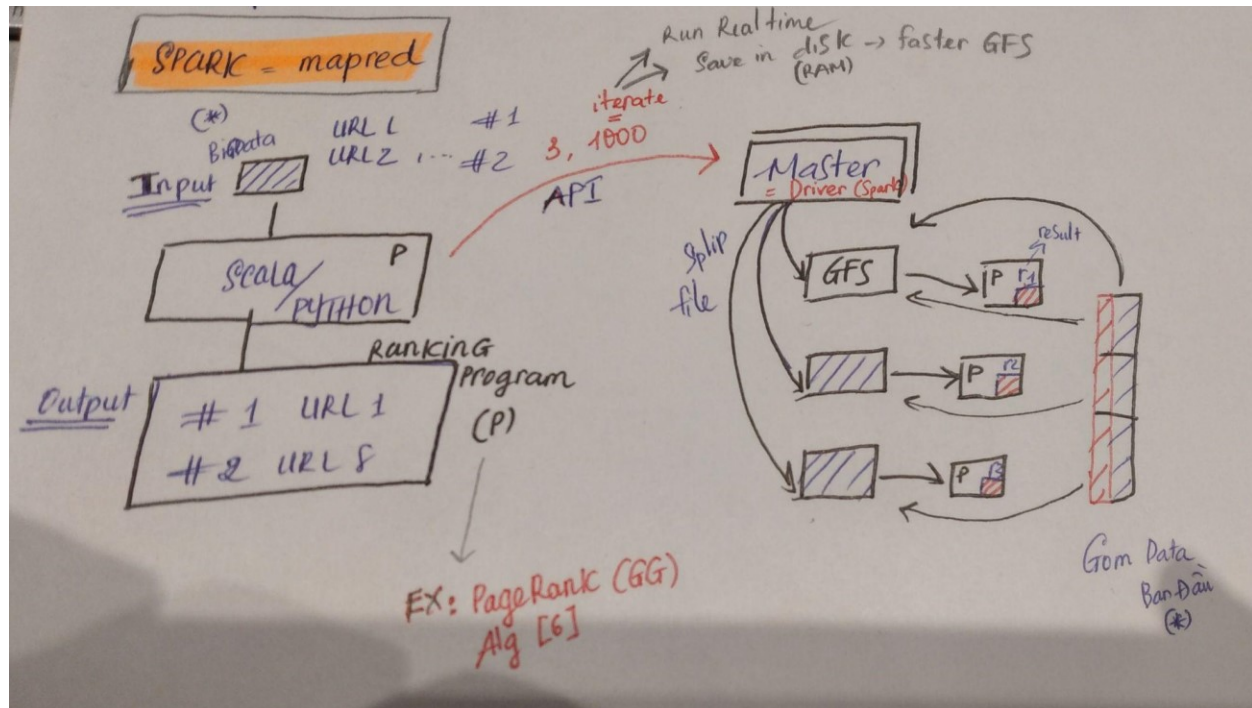
Screenshot captured after running your code.



Describe how a MapReduce program runs on 6 machines (1 Master, 3 Mappers, 2 Reducers) using a sequence diagram.



7. LAB 7: Cloud Computing (Spark)



Run SparkPageRank:

```

1  val lines = spark.read.textFile("in2.txt").rdd
2  val links1 = lines.map(s =>
3    val parts = s.split("\\s+")
4    (parts(0), parts(1))
5  )
6  val links2 = links1.distinct()
7  val links3 = links2.groupByKey()
8  val links4 = links3.cache()
9  var ranks = links4.mapValues(v => 1.0)
10
11 for (i <- 1 to 10) {
12   val jj = links4.join(ranks)
13   val contribs = jj.values.flatMap{
14     case (urls, rank) =>
15       urls.map(url => (url, rank / urls.size))
16   }
17   ranks = contribs.reduceByKey(_ + _).mapValues(_ * 0.15 + 0.85 * _)
18 }
19 val output = ranks.collect()
20 output.foreach(tup => println(s"${tup._1} has rank: ${tup._2} ."))

```

Terminal Output:

```

scala> :load SparkPageRank2.scala
loading SparkPageRank2.scala...
lines: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[4] at rdd at SparkPageRank2.scala:22
links1: org.apache.spark.rdd.RDD[(String, String)] = MapPartitionsRDD[5] at map at SparkPageRank2.scala:23
links2: org.apache.spark.rdd.RDD[(String, String)] = MapPartitionsRDD[8] at distinct at SparkPageRank2.scala:23
links3: org.apache.spark.rdd.RDD[(String, Iterable[String])] = ShuffledRDD[9] at groupByKey at SparkPageRank2.scala:23
links4: links3.type = ShuffledRDD[9] at groupByKey at SparkPageRank2.scala:23
ranks: org.apache.spark.rdd.RDD[(String, Double)] = MapPartitionsRDD[10] at mapValues at SparkPageRank2.scala:23
output: Array[(String, Double)] = Array((Spark,1.546762000020236), (Google,0.4916304117986312), (Baidu,1.1516189999898816), (HCMUS,0.8099885881912505))
Spark has rank: 1.546762000020236
Google has rank: 0.4916304117986312
Baidu has rank: 1.1516189999898816
HCMUS has rank: 0.8099885881912505

```