

# Bigdata Technologies – CSP554

## Project Report

By,

Abhishek Bonageri

A20492521

[abonageri@hawk.iit.edu](mailto:abonageri@hawk.iit.edu)

GitHub repo: <https://github.com/AbhishekBonageri-75/BigDataProject>

## **Project: Building Elastic Search Feature on HDFS and search using Map-Reduce**

### **Abstract :**

#### **Objective and Development summary:**

Behemoth is an Apache Hadoop-based open-source platform for processing massive amounts of documents. It consists of a few modules that work with these documents as well as a straightforward annotation-based implementation of a document.

Objective here is to solve 2 of the GitHub issues in this open-source repository which are, building search functionality for a file stored in HDFS using Map reduce, Elastic Search and test cases for the same.

### **Overview:**

#### **Solution outline and proposed system**

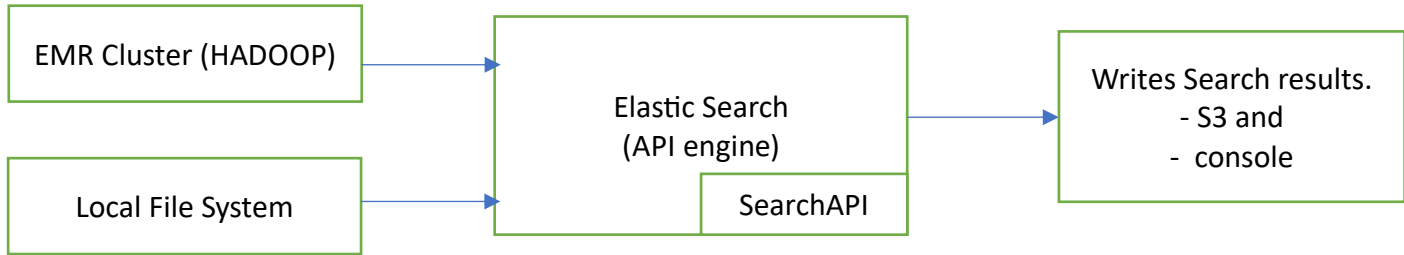
Elastic Search:

- Using elastic search and its API is a nice way to perform operations on a document.
- So, I have used the “Search API” of elastic search to check if the word exists in a document or not.
- Before any of the Search api, the file has to be indexed and I have written the python code to index the file and also wrote a python program that uses search api provided by elastic search by using the indexed file.
- The program prints out the name of the file if the search is success otherwise it stops the execution.

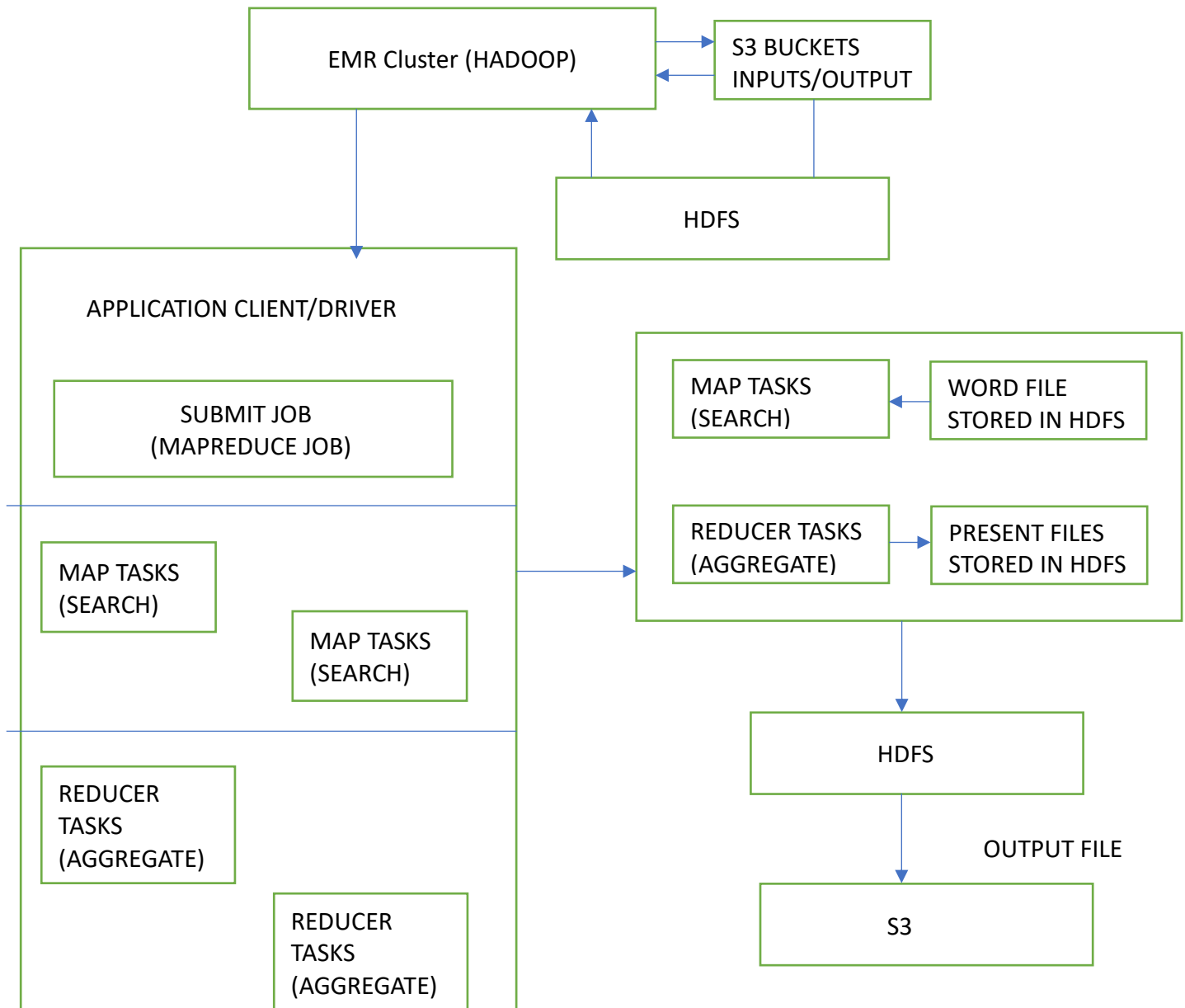
Map-reduce:

- Wrote mapper and reducers to search for the word inside a file which is stored on HDFS.
- The input files are pulled from AWS S3
- Mappers perform the operation to search for the word in the file.
- And then the result of search is pushed back to S3 bucket as depicted in the architecture.

## Architecture 1 (Search Using Elastic Search):



## Architecture 2 (Search Using Map Reduce):



## Components of the architecture:

- 1- AWS S3
  - The input file and output result will be stored in this Amazon Web Services (AWS) cloud-based storage service. The data to be searched input file will be kept in an S3 bucket, and the word search's findings will be written back into another S3 bucket.
- 2- AWS EMR
  - EMR is a managed big data processing service offered by AWS that processes sizable data sets using the Hadoop and Spark frameworks. When processing large amounts of data, EMR uses the MapReduce programming model in a scalable and economical manner. In this architecture, the MapReduce program will be run through EMR to search for a word in the input file kept in HDFS.
- 3- HDFS
  - A distributed file system called HDFS is a component of Apache Hadoop. It is used to distribute large amounts of data across several Hadoop cluster nodes. The input file to be searched will be kept in HDFS on AWS EMR in this architecture.
- 4- Elastic Search API engine
  - A distributed, open-source search and analytics engine built on top of Apache Lucene is called Elasticsearch. It is made to be incredibly scalable, quick, and fault-tolerant, and it offers a RESTful API for data access. It is frequently employed in applications that call for the instantaneous search and evaluation of sizable data sets.
  - I have only used "Search API" provided by the Elastic Search API engine.

## Design/Flow:

### When Using Elasticsearch:

- Every file must be indexed and added to HDFS before performing any operation on the file using API provided by Elastic Search
- Once the data file is indexed using indexing.py program, elastic search engine APIs can be used to retrieve data or perform any operations in split seconds on the data file using its index.

### When Using Map Reduce:

General or basic understanding of the flow:

- This part of the project is a small chunk from a bigger project which only deals with searching for text/words from a file stored in the HDFS using mappers and reducers and store the result back in S3 buckets from where it fetches the initial data.

The Flow:

- Retrieve Input File: The input file containing the searchable data will be kept in an S3 bucket. The input file is taken from the S3 bucket and stored in HDFS on the AWS EMR as the first step in the design process. The S3 API offered by AWS, which enables reading and writing files to and from S3 buckets, can be used for this.
- Map Phase: The MapReduce program will be run on AWS EMR after the input file has been stored in HDFS. The input file is processed in parallel across several EMR cluster nodes during the Map phase. Each node will read a portion of the input file from HDFS and then apply the Map function to every record in that portion. Words will be extracted from the records by the Map function, which then emits key-value pairs with the word as the key and the record as the value.
- Reduce Phase: To create the output, the Reduce phase processes the key-value pairs created in the Shuffle and Sort phases. The word search operation will be carried out on the values by the Reduce function after receiving the key-value pairs for each key. The Reduce function will output a key-value

**Writing Output Files:** After the word search, the output key-value pairs produced by the Reduce function will be written back into an S3 bucket. The S3 API offered by AWS, which enables writing files to S3 buckets, can be used for this.

```
> cd elasticsearch-8.7.0
> ls
LICENSE.txt    README.asciidoc config      lib         modules
NOTICE.txt     bin        jdk.app   logs       plugins
> cd bin
> ls
elasticsearch                  elasticsearch-plugin
elasticsearch-certgen          elasticsearch-reconfigure-node
elasticsearch-certutil         elasticsearch-reset-password
elasticsearch-cli               elasticsearch-saml-metadata
elasticsearch-create-enrollment-token elasticsearch-service-tokens
elasticsearch-croneval         elasticsearch-setup-passwords
elasticsearch-env              elasticsearch-shard
elasticsearch-env-from-file    elasticsearch-sql-cli
elasticsearch-geopip           elasticsearch-sql-cli-8.7.0.jar
elasticsearch-keystore         elasticsearch-syskeygen
elasticsearch-node             elasticsearch-users
> ./elasticsearch
[2023-04-27T14:05:57,041][INFO ][o.e.n.Node                    ] [Abhisheks-MacBook-Pro.local] version[8.7.0], pid[80939], build[tar/09520b59b6bc1057340b5575018646ea715e30e/2023-03-27T16:31:09.816451435Z], OS[M
ac OS X/j13.3.1/aarch64], JVM[Oracle Corporation/OpenJDK 64-Bit Server VM/19.0.2/19.0.2+7-44]
[2023-04-27T14:05:57,044][INFO ][o.e.n.Node                    ] [Abhisheks-MacBook-Pro.local] JVM home [/Users/ab/Downloads/elasticsearch-8.7.0/jdk.app/Contents/Home], using bundled JDK [true]
[2023-04-27T14:05:57,044][INFO ][o.e.n.Node                    ] [Abhisheks-MacBook-Pro.local] JVM arguments [-Des.networkaddress.cache.ttl=60,-Des.networkaddress.cache.negative.ttl=10,-Djava.security.manager=
allow,-XX:+AlwaysPreTouch,-Xss1m,-Djava.awt.headless=true,-Dfile.encoding=UTF-8,-Djna.nosys=true,-XX:-OmitStackTraceInFastThrow,-Dio.netty.noUnsafe=true,-Dio.netty.noKeySetOptimization=true,-Dio.ne
tty.recycler.maxCapacityPerThread=0,-Dlog4j.shutdownHookEnabled=false,-Dlog4j2.disable.jmx=true,-Dlog4j2.formatMsgNoLookups=true,-Djava.locale.providers=SPI,CMPAT,--add-opens=java.base/java.io=ALL-UNN
AMED,-XX:+UseG1GC,-Djava.io.tmpdir=/var/folders/g3/2jy2tpvn6lcbjdgg20q1shh0BQnpn/T/elasticsearch-5764348323160997086,-XX:+HeapDumpOnOutOfMemoryError,-XX:+ExitOnOutOfMemoryError,-XX:HeapDumpPath=data
,-XX:ErrorFile=logs/hc_err_pidmp.log,-Xlog:gc*,gc+age=trace,safepoint:file=logs/gc_log:utctime,lvl=pid,tags=file,count=32,filesize=64m,-Xms4096m,-Xmx4096m,-XX:MaxDirectMemorySize=2147483648,-XX:G1HeapR
egionSize=4m,-XX:InitiatingHeapOccupancyPercent=30,-XX:G1ReservePercent=15,-Des.distribution.type=tar,--module-path=/Users/ab/Downloads/elasticsearch-8.7.0/lib,--add-modules=jdk.net,-Djdk.module.main=
org.elasticsearch.server]
[2023-04-27T14:05:57,801][INFO ][c.a.c.i.j.JacksonVersion     ] [Abhisheks-MacBook-Pro.local] Package versions: jackson-core=2.13.4, jackson-databind=2.13.4.2, jackson-dataformat-xml=2.13.4, jackson-datatype-js
r310=2.13.4, azure-core=1.34.0, Troubleshooting version conflicts: https://aka.ms/aszsd/k/java/dependency/troubleshoot
[2023-04-27T14:05:58,602][INFO ][o.e.p.PluginsService       ] [Abhisheks-MacBook-Pro.local] loaded module [aggregations]
[2023-04-27T14:05:58,603][INFO ][o.e.p.PluginsService       ] [Abhisheks-MacBook-Pro.local] loaded module [analysis-common]
[2023-04-27T14:05:58,603][INFO ][o.e.p.PluginsService       ] [Abhisheks-MacBook-Pro.local] loaded module [apm]
[2023-04-27T14:05:58,603][INFO ][o.e.p.PluginsService       ] [Abhisheks-MacBook-Pro.local] loaded module [blob-cache]
[2023-04-27T14:05:58,603][INFO ][o.e.p.PluginsService       ] [Abhisheks-MacBook-Pro.local] loaded module [constant-keyword]
```

11%

3.7 GB

0.0 kB/s

0.0 kB/s

Elasticsearch security features have been automatically configured!  
Authentication is enabled and cluster connections are encrypted.

Password for the **elastic** user (reset with '`'bin/elasticsearch-reset-password -u elastic'`):  
`5S*etImIQ2++Lln2VB`

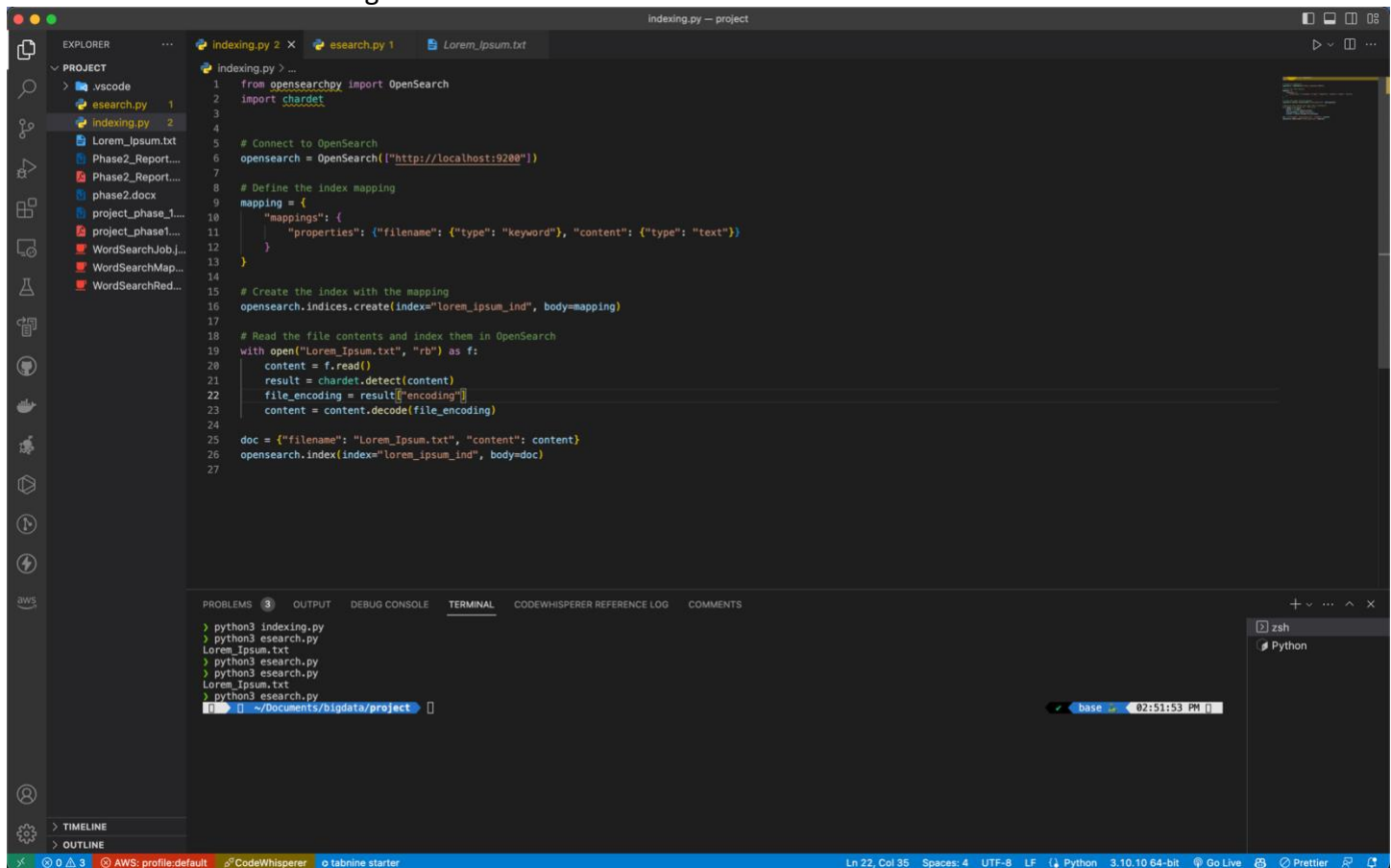
HTTP CA certificate SHA-256 fingerprint:  
`62fbdbd9a30f5a36ca492bcc8f2c32a62ce08b59db169a58ea0bf793c6134c8`

Configure Kibana to use this cluster:

Run Kibana and click the configuration link in the terminal when Kibana starts.  
Copy the following enrollment token and paste it into Kibana in your browser (valid for the next 30 minutes):  
`eYJZZXiIoi14ljcuMCSimFaciIGwyIxMDQuMTK0BLExNS4ynJo5MJAmlIsImZnciI6IjVvZjh1ZHBlbnNTNmhmZUluZSQSMmJaYZmhbmWzMmE2ZWlmNDMA4tVSJSxIXlhNYThBTjZjcSMkZDMTM0BgvgZjlCJRzk1OIJDOWtieEljQnrOQZhLT0SImmR2SzpyR2RhHwZ  
QWJldRudndBOvWBcpwnInfo=`

Configure other nodes to join this cluster:  
On this node:  
Create an enrollment token with '`'bin/elasticsearch-create-enrollment-token -s node'`.  
Uncomment the `transport.host` setting at the end of `config/elasticsearch.yml`.  
Restart Elasticsearch.  
On other nodes:  
Start Elasticsearch with '`'bin/elasticsearch --enrollment-token <token>`', using the enrollment token that you generated.

## Code for File content indexing...



The screenshot shows a VS Code editor with a project named 'indexing.py - project'. The Explorer sidebar on the left shows the project structure, including files like 'eSearch.py', 'indexing.py', 'Lorem Ipsum.txt', and various report files. The main editor displays the 'indexing.py' file, which contains the following Python code:

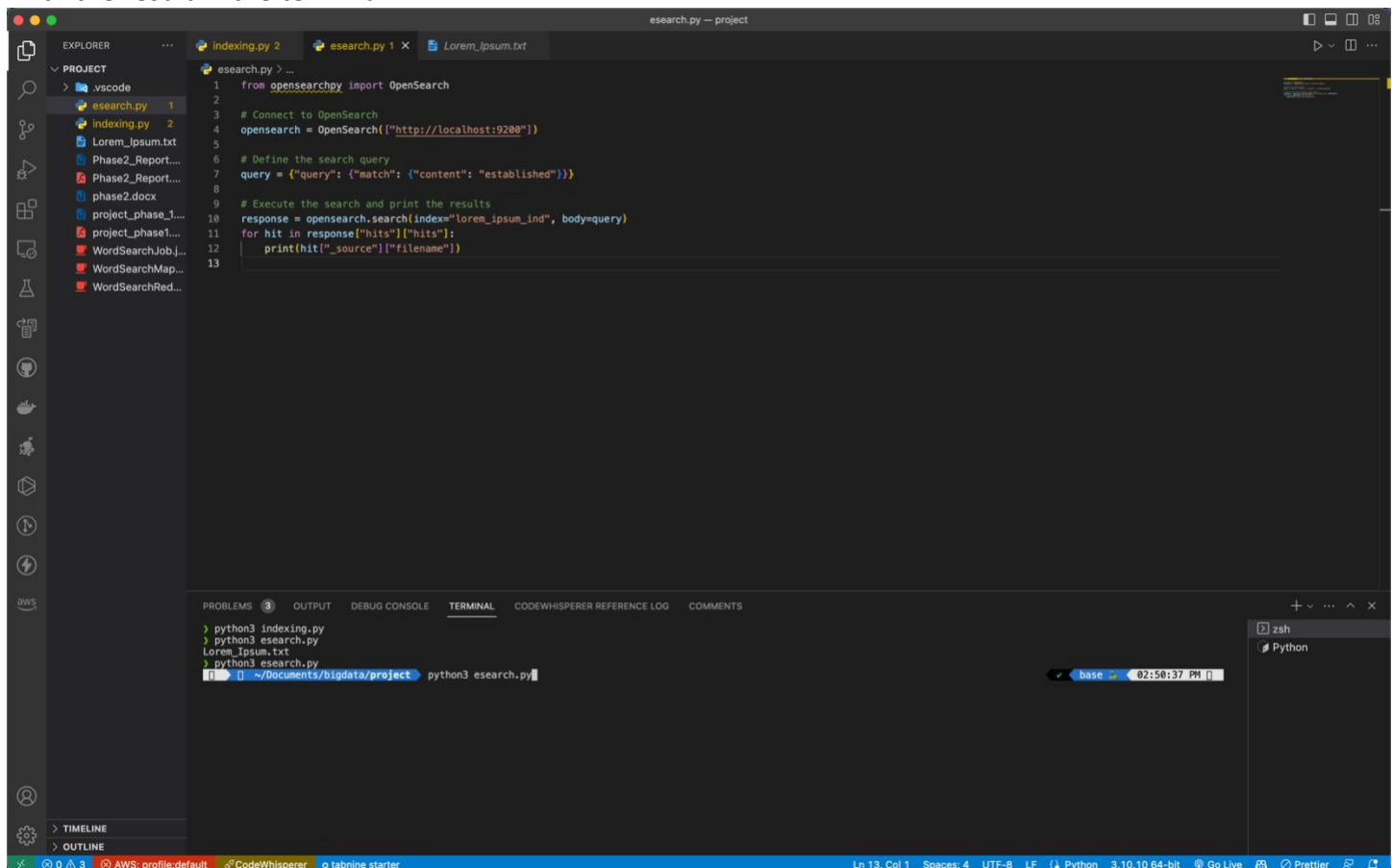
```
1 from opensearchpy import OpenSearch
2 import chardet
3
4 # Connect to OpenSearch
5 opensearch = OpenSearch(["http://localhost:9200"])
6
7 # Define the index mapping
8 mapping = {
9     "mappings": {
10         "properties": {"filename": {"type": "keyword"}, "content": {"type": "text"}}
11     }
12 }
13
14 # Create the index with the mapping
15 opensearch.indices.create(index="lorem_ipsum_ind", body=mapping)
16
17 # Read the file contents and index them in OpenSearch
18 with open("Lorem Ipsum.txt", "rb") as f:
19     content = f.read()
20     result = chardet.detect(content)
21     file_encoding = result["encoding"]
22     content = content.decode(file_encoding)
23
24 doc = {"filename": "Lorem Ipsum.txt", "content": content}
25 opensearch.index(index="lorem_ipsum_ind", body=doc)
26
27
```

The bottom panel shows the TERMINAL tab with the following commands and output:

```
> python3 indexing.py
> python3 eSearch.py
Lorem Ipsum.txt
> python3 eSearch.py
> python3 eSearch.py
Lorem Ipsum.txt
> python3 eSearch.py
~/Documents/bigdata/project
```

The status bar at the bottom indicates the file is at Line 22, Column 35, using UTF-8 encoding.

## Code for searching for the content in the file With the result in the terminal



The screenshot shows a VS Code editor with a project named 'eSearch.py - project'. The Explorer sidebar on the left shows the project structure, including files like 'eSearch.py', 'indexing.py', 'Lorem Ipsum.txt', and various report files. The main editor displays the 'eSearch.py' file, which contains the following Python code:

```
1 from opensearchpy import OpenSearch
2
3 # Connect to OpenSearch
4 opensearch = OpenSearch(["http://localhost:9200"])
5
6 # Define the search query
7 query = {"query": {"match": {"content": "established"}}}
8
9 # Execute the search and print the results
10 response = opensearch.search(index="lorem_ipsum_ind", body=query)
11 for hit in response["hits"]["hits"]:
12     print(hit["_source"]["filename"])
13
```

The bottom panel shows the TERMINAL tab with the following commands and output:

```
> python3 indexing.py
> python3 eSearch.py
Lorem Ipsum.txt
> python3 eSearch.py
python3 eSearch.py
```

The status bar at the bottom indicates the file is at Line 13, Column 1, using UTF-8 encoding.

## Word search using Map reduce (MR job)

### Mapper.py

```
mapper.py 1 x  Lorem Ipsum.txt  reducer.py 1  Extension: Python
mapper.py > ...
1  from mrjob.job import MRJob
2
3  class WordSearchMapper(MRJob):
4      def configure_args(self):
5          super(WordSearchMapper, self).configure_args()
6          self.add_passthru_arg('--word', type=str, help='The word to search for')
7
8      def mapper(self, _, line):
9          word = self.options.word.lower()
10         line = line.lower()
11         words = line.split()
12         for w in words:
13             if w == word:
14                 yield word, 1
15
16 if __name__ == '__main__':
17     WordSearchMapper.run()
18
```

### Reducer.py

```
reducer.py — hadoop [S
mapper.py 1  Lorem Ipsum.txt  reducer.py 1 x  Extension: Py
reducer.py > ...
1  from mrjob.job import MRJob
2
3  class WordSearchReducer(MRJob):
4      def reducer(self, key, values):
5          yield key, sum(values)
6
7  if __name__ == '__main__':
8      WordSearchReducer.run()
9
10
```



```
> ssh -i bigdata.pem hadoop@ec2-3-237-30-155.compute-1.amazonaws.com
The authenticity of host 'ec2-3-237-30-155.compute-1.amazonaws.com (3.237.30.155)' can't be established.
ED25519 key fingerprint is SHA256:szO60D5IR+MgwbwOUiBqBxMDAk3gARfX5bX8MurNIZI.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-3-237-30-155.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
```

```
--|  --|_ )
_| (      /  Amazon Linux 2 AMI
---|\---|---
```

<https://aws.amazon.com/amazon-linux-2/>

```
EEEEEEEEEEEEEEEEEEEE MMMMMMMM          MMMMMMMM RRRRRRRRRRRRRRRR
E::::::::::::::::::::E M::::::::M          M::::::::M R::::::::::::R
EE::::::::EEEEEEEE::::E M::::::::M          M::::::::M R::::RRRRRR::::R
  E::::E      EEEEE M::::::::M          M::::::::M RR::::R      R::::R
  E::::E      M::::M::M          M::M::::M          R::R      R::::R
  E::::EEEEEEEEEE M::::M M::M M::M M::::M          R::RRRRRR::::R
  E::::::::::::E M::::M M::M::M M::::M          R::::::::::::RR
  E::::EEEEEEEEEE M::::M M::::M M::::M          R::RRRRRR::::R
  E::::E      M::::M          M::M          M::::M          R::R      R::::R
  E::::E      EEEEE M::::M          MMM          M::::M          R::R      R::::R
EE::::::::EEEEEEEE::::E M::::M          M::::M          R::R      R::::R
E::::::::::::E M::::M          M::::M RR::::R          R::::R
EEEEEEEEEEEEEEEEEEEE MMMMMMMM          MMMMMMMM RRRRRRR      RRRRRR
```

```
[hadoop@ip-172-31-72-68 ~]$ pwd
/home/hadoop
```

```
[hadoop@ip-172-31-89-208 ab2]$ javac -cp $HADOOP_CLASSPATH WordSearchM^C
[hadoop@ip-172-31-89-208 ab2]$ javac -cp $HADOOP_CLASSPATH WordSearchJob.java
[hadoop@ip-172-31-89-208 ab2]$ javac -cp $HADOOP_CLASSPATH WordSearchMapper.java
[hadoop@ip-172-31-89-208 ab2]$ javac -cp $HADOOP_CLASSPATH WordSearchReducer.java
[hadoop@ip-172-31-89-208 ab2]$ ls
WordSearchJob.class WordSearchJob.java WordSearchMapper.class WordSearchMapper.java WordSearchReducer.class WordSearchReducer.java
[hadoop@ip-172-31-89-208 ab2]$ jar cf wordsearch.jar WordSearchJob.class WordSearchMapper.class WordSearchReducer.class
[hadoop@ip-172-31-89-208 ab2]$ ls
wordsearch.jar WordSearchJob.class WordSearchJob.java WordSearchMapper.class WordSearchMapper.java WordSearchReducer.class WordSearchReducer.java
[hadoop@ip-172-31-89-208 ab2]$ jar tvf wordsearch.jar
  0 Sun Apr 16 19:52:56 UTC 2023 META-INF/
 66 Sun Apr 16 19:52:56 UTC 2023 META-INF/MANIFEST.MF
1643 Sun Apr 16 19:52:26 UTC 2023 WordSearchJob.class
2327 Sun Apr 16 19:52:32 UTC 2023 WordSearchMapper.class
1702 Sun Apr 16 19:52:40 UTC 2023 WordSearchReducer.class
[hadoop@ip-172-31-89-208 ab2]$ hadoop jar wordsearch.jar WordSearchJob /input/Lorem_Ipsum.txt /output755 Lorem
23/04/16 19:54:04 INFO client.RMProxy: Connecting to ResourceManager at ip-172-31-89-208.ec2.internal/172.31.89.208:8032
23/04/16 19:54:04 INFO client.AHSProxy: Connecting to Application History server at ip-172-31-89-208.ec2.internal/172.31.89.208:10200
23/04/16 19:54:04 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
23/04/16 19:54:04 WARN mapreduce.JobResourceUploader: No job jar file set. User classes may not be found. See Job or Job#setJar(String).
23/04/16 19:54:04 INFO input.FileInputFormat: Total input files to process : 1
23/04/16 19:54:04 INFO lzo.GPLNativeCodeLoader: Loaded native gpl library
23/04/16 19:54:04 INFO lzo.LzoCodec: Successfully loaded & initialized native-lzo library [hadoop-lzo rev 049362b7cf53ff5f739d6b1532457f2c6cd495e8]
23/04/16 19:54:05 INFO mapreduce.JobSubmitter: number of splits:1
23/04/16 19:54:05 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1681664091004_0016
23/04/16 19:54:05 INFO mapred.YARNRunner: Job jar is not present. Not adding any jar to the list of resources.
23/04/16 19:54:05 INFO conf.Configuration: resource-types.xml not found
23/04/16 19:54:05 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
23/04/16 19:54:05 INFO resource.ResourceUtils: Adding resource type - name = memory-mb, units = Mi, type = COUNTABLE
23/04/16 19:54:05 INFO resource.ResourceUtils: Adding resource type - name = vcores, units = , type = COUNTABLE
23/04/16 19:54:06 INFO impl.YarnClientImpl: Submitted application application_1681664091004_0016
23/04/16 19:54:06 INFO mapreduce.Job: The url to track the job: http://ip-172-31-89-208.ec2.internal:20888/proxy/application_1681664091004_0016/
23/04/16 19:54:06 INFO mapreduce.Job: Running job: job_1681664091004_0016
23/04/16 19:54:14 INFO mapreduce.Job: Job job_1681664091004_0016 running in uber mode : false
23/04/16 19:54:14 INFO mapreduce.Job: map 0% reduce 0%
```

```

23/04/16 19:54:38 INFO mapreduce.Job: map 100% reduce 100%
23/04/16 19:54:38 INFO mapreduce.Job: Job job_1681664091004_0016 failed with state FAILED due to: Task failed task_1681664091004_0016_m_000000
Job failed as tasks failed. failedMaps:1 failedReduces:0

23/04/16 19:54:38 INFO mapreduce.Job: Counters: 10
  Job Counters
    Failed map tasks=4
    Killed reduce tasks=1
    Launched map tasks=4
    Other local map tasks=3
    Data-local map tasks=1
    Total time spent by all maps in occupied slots (ms)=739440
    Total time spent by all reduces in occupied slots (ms)=0
    Total time spent by all map tasks (ms)=15405
    Total vcore-milliseconds taken by all map tasks=15405
    Total megabyte-milliseconds taken by all map tasks=23662080

ImportError: You must install boto3 to connect to S3
(myenv) [hadoop@ip-172-31-72-68 ~]$ pip install boto3
Defaulting to user installation because normal site-packages is not writeable
Collecting boto3
  Downloading boto3-1.26.114-py3-none-any.whl (135 kB)
    |#####| 135 kB 35.1 MB/s
Requirement already satisfied: jmespath<2.0.0,>=0.7.1 in /usr/local/lib/python3.7/site-packages (from boto3) (1.0.0)
Collecting botocore<1.30.0,>=1.29.114
  Downloading botocore-1.29.114-py3-none-any.whl (10.6 MB)
    |#####| 10.6 MB 40.5 MB/s
Collecting s3transfer<0.7.0,>=0.6.0
  Downloading s3transfer-0.6.0-py3-none-any.whl (79 kB)
    |#####| 79 kB 14.7 MB/s
Collecting urllib3<1.27,>=1.25.4
  Downloading urllib3-1.26.15-py2.py3-none-any.whl (140 kB)
    |#####| 140 kB 42.5 MB/s
Collecting python-dateutil<3.0.0,>=2.1
  Downloading python_dateutil-2.8.2-py2.py3-none-any.whl (247 kB)
    |#####| 247 kB 38.5 MB/s
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/site-packages (from python-dateutil<3.0.0,>=2.1->botocore<1.30.0,>=1.29.114->boto3) (1.13.0)
Installing collected packages: urllib3, python-dateutil, botocore, s3transfer, boto3
Successfully installed boto3-1.26.114 botocore-1.29.114 python-dateutil-2.8.2 s3transfer-0.6.0 urllib3-1.26.15

(myenv) [hadoop@ip-172-31-72-68 ~]$ python mapper.py -r emr s3://mybuck75785/Lorem Ipsum.txt --output s3://mybuck75785out/output2/ --word "Lorem "
No configs found; falling back on auto-configuration
No configs specified for emr runner
Using s3://mrjob-b44a3cd1c25ebcee/tmp/ as our temp dir on S3
Creating temp directory /tmp/mapper.hadoop.20230416.210931.414980
uploading working dir files to s3://mrjob-b44a3cd1c25ebcee/tmp/mapper.hadoop.20230416.210931.414980/files/wd...
Copying other local files to s3://mrjob-b44a3cd1c25ebcee/tmp/mapper.hadoop.20230416.210931.414980/files/
Can't access IAM API, trying default instance profile: EMR_EC2_DefaultRole
Can't access IAM API, trying default service role: EMR_DefaultRole
Created new cluster j-GYU923J2N6WE
Added EMR tags to cluster j-GYU923J2N6WE: __mrjob_label=mapper, __mrjob_owner=hadoop, __mrjob_version=0.7.4
Waiting for Step 1 of 1 (s-16SSKPTF1087A) to complete...
  PENDING (cluster is STARTING)
  PENDING (cluster is STARTING)
  PENDING (cluster is STARTING)
  PENDING (cluster is STARTING)
  PENDING (cluster is STARTING: Configuring cluster software)
  PENDING (cluster is STARTING: Configuring cluster software)
  PENDING (cluster is STARTING: Configuring cluster software)
  PENDING (cluster is STARTING: Configuring cluster software)
  PENDING (cluster is STARTING: Configuring cluster software)
  PENDING (cluster is STARTING: Configuring cluster software)
  PENDING (cluster is STARTING: Configuring cluster software)
  master node is ec2-34-221-247-97.us-west-2.compute.amazonaws.com
  PENDING (cluster is RUNNING: Running step)
  PENDING (cluster is RUNNING: Running step)
  PENDING (cluster is RUNNING: Running step)
  COMPLETED
Attempting to fetch counters from logs...
Waiting for cluster (j-GYU923J2N6WE) to terminate...
  TERMINATING
  TERMINATING
  TERMINATED
Looking for step log in s3://mrjob-b44a3cd1c25ebcee/tmp/logs/j-GYU923J2N6WE/steps/s-16SSKPTF1087A...
  Parsing step log: s3://mrjob-b44a3cd1c25ebcee/tmp/logs/j-GYU923J2N6WE/steps/s-16SSKPTF1087A/syslog.gz
Counters: 38
  ...
job output is in s3://mybuck75785out/output2/
Removing s3 temp directory s3://mrjob-b44a3cd1c25ebcee/tmp/mapper.hadoop.20230416.210931.414980/...
Removing temp directory /tmp/mapper.hadoop.20230416.210931.414980...
Removing log files in s3://mrjob-b44a3cd1c25ebcee/tmp/logs/j-GYU923J2N6WE/...
Terminating cluster: j-GYU923J2N6WE

```



## Test case execution

```
Abhisheks-MacBook-Pro:project ab$ python3 -m unittest unittests.py
F.F

FAIL: test_search_with_multiple_terms (unittests.TestSearchForWordInDocument)

Traceback (most recent call last):
  File "/Users/ab/Documents/bigdata/project/unittests.py", line 43, in test_search_with_multiple_terms
    self.assertEqual(result["hits"]["total"]["value"], 1)
AssertionError: 0 != 1

FAIL: test_word_exists_in_document (unittests.TestSearchForWordInDocument)

Traceback (most recent call last):
  File "/Users/ab/Documents/bigdata/project/unittests.py", line 24, in test_word_exists_in_document
    self.assertEqual(result["hits"]["total"]["value"], 1)
AssertionError: 0 != 1

Ran 3 tests in 0.227s

FAILED (failures=2)
Abhisheks-MacBook-Pro:project ab$
```

## Unittest.py

```
unittests.py — project
indexing.py 2, U  unittests.py 1, U  x  esearch.py 1, U  Lorem Ipsum.txt U

unittests.py > TestSearchForWordInDocument > setUp
1  import unittest
2  from opensearchpy import OpenSearch
3
4
5  class TestSearchForWordInDocument(unittest.TestCase):
6      def setUp(self):
7          self.opensearch = OpenSearch()
8          self.index_name = "lorem_ipsum_ind2"
9          self.doc_id = "1"
10         self.doc = {
11             "title": "lorem_ipsum.txt",
12             "content": "This is a test document for search",
13             "tags": ["test", "search", "document"],
14         }
15
16         self.opensearch.index(index=self.index_name, id=self.doc_id, body=self.doc)
17
18         # def tearDown(self):
19         #     self.opensearch.delete(index=self.index_name, id=self.doc_id)
20
21     def word_exists_test(self):
22         query = {"query": {"match": {"content": "lorem"}}}
23         result = self.opensearch.search(index=self.index_name, body=query)
24         self.assertEqual(result["hits"]["total"]["value"], 1)
25
26     def word_does_not_exist_test(self):
27         query = {"query": {"match": {"content": "hello"}}}
28         result = self.opensearch.search(index=self.index_name, body=query)
29         self.assertEqual(result["hits"]["total"]["value"], 0)
30
31     def multiple_terms_test(self):
32         query = {
33             "query": {
34                 "bool": {
35                     "must": [
36                         {"match": {"content": "test"}},
37                         {"match": {"title": "document"}},
38                     ]
39                 }
40             }
41         }
42         result = self.opensearch.search(index=self.index_name, body=query)
43         self.assertEqual(result["hits"]["total"]["value"], 1)
44
```

**Conclusion:**

- I was able to build search feature using both, Map reduce and Elastic search and write test cases. I should have learnt more about the architecture of elastic search and
- My findings:  
there is a free version of Elastic Search offered by AWS called OpenSearch which basically works like elastic search.
- There were difficulties while trying out the algorithm on a huge file, but the results obtained by elastic search were much quicker (almost instant) than that of Map reduce algorithm as we index the file before using Elastic Search API.

**References**

- <https://coralogix.com/blog/elasticsearch-hadoop-tutorial-with-hands-on-examples/->  
Integrating elastic search with HDFS.
- <https://hub.packtpub.com/integrating-elasticsearch-hadoop-ecosystem>
- <https://aws.amazon.com/blogs/big-data/getting-started-with-elasticsearch-and-kibana-on-amazon-emr/>
- <https://docs.aws.amazon.com/opensearch-service/latest/developerguide/what-is.html>
- <https://docs.aws.amazon.com/opensearch-service/latest/developerguide/gsg.html>
- <https://www.elastic.co/guide/en/cloud/current/index.html>