CS 203 Assignment-8

Team 20

Dakshata Bhamare (23210027)

Chinmay Pendse (23110245)

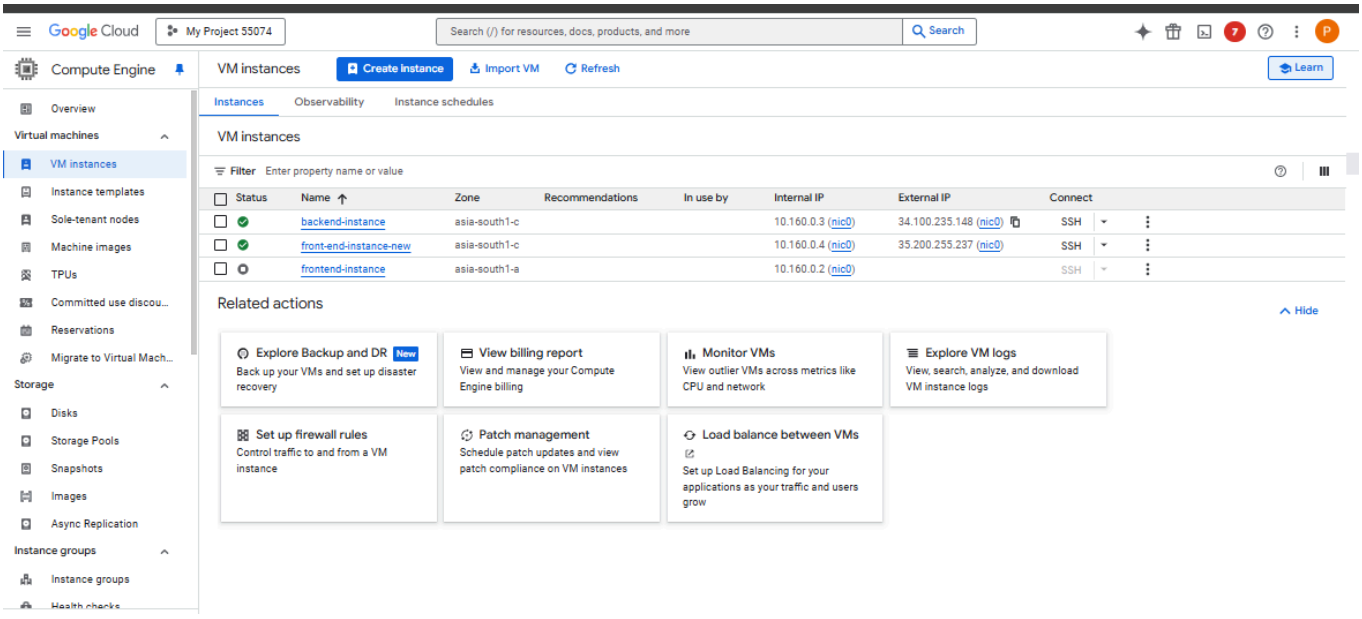
Introduction

Github Link: https://github.com/chinmayp995/CS-203-Assignment-8

This assignment majorly focussed on the connection of the two virtual machines and Docker Implementation. Through the codes present in the github repositories we created two machines: one for frontend and one for backend. In the frontend machine we had one container for fastapi and in backend we had two containers: one for fastapi and one for elasticsearch. The docker image instances links are given below.

Environment Setup

For using several Virtual Machines we used Google Cloud wherein we created Virtual Machines (VM) instances which were for frontend and backend as shown in the image below.



We have used **E2 medium level** machines as per our requirements, and then we then created the directories as described by the github repository. Now, we firstly install docker into both the

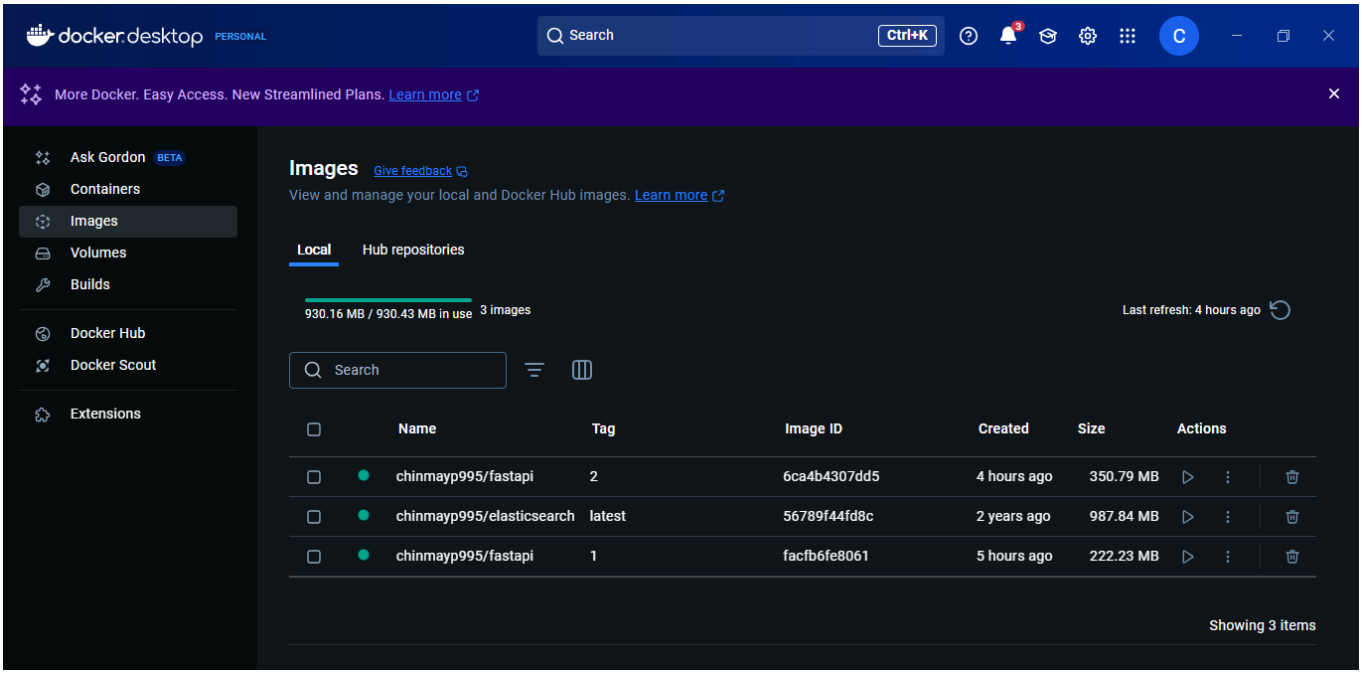
machines such that we can use the docker containers and images for the assignment. After generating all the images using the command

```
sudo docker build -t <the image> .
```

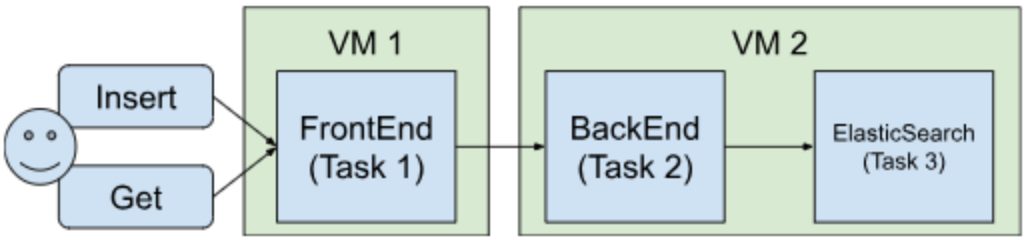
We then pushed these to the docker hub using the command by tagging the image which was generated in our machine (similarly for other two images too)

```
sudo docker tag c36aec59ba04 chinmayp995/fastapi:1
sudo docker push chinmayp995/fastapi:2
```

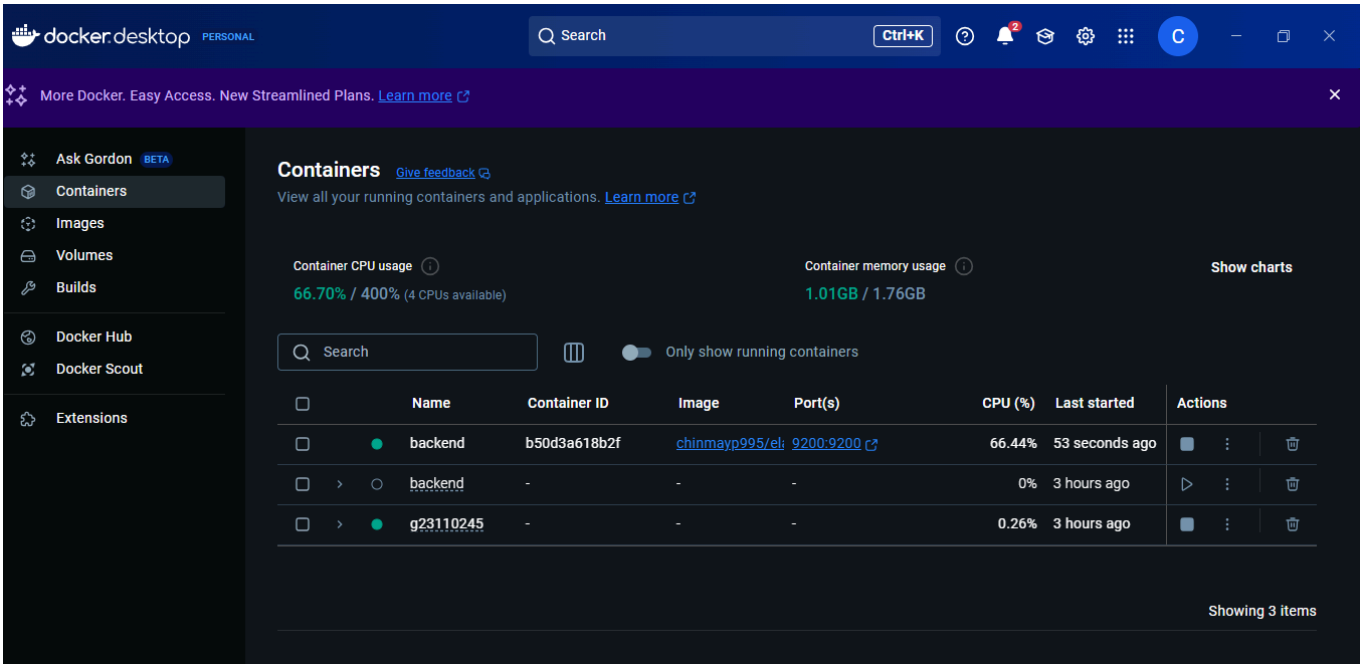
We get all our images on the Docker Hub



We then run the images properly assigning them containers as per the **architecture diagram** as shown below



We then get containers backend and frontend in the Docker as shown below (Pls neglect one backend container which couldn't be deleted)



After the successful connection between the frontend and backend we get the output through Docker as well. Although in the course of this assignment; we used the Google Console In browser SSH to run all our commands and get our desired output.

Links to Docker Images

1. fastapi:1 : Image Layer Details - chinmayp995/fastapi:1 | Docker Hub
2. fastapi:2: Image Layer Details - chinmayp995/fastapi:2 | Docker Hub
3. elasticsearch: Image Layer Details - chinmayp995/elasticsearch:latest | Docker Hub

Methodology of the Assignment

In this assignment; as discussed earlier we have both of codes for frontend and backend in two separate VM instances as you can see in the screenshot. Now, to run the python scripts within the container we have to build the docker container from scratch. Firstly, we remove all the containers and unrequired fileswhich are with the similar name such that there is no redundancy and docker is not able to process two containers with same name. So, we ran

```
sudo docker-compose down -v --remove-orphans
```

Which outputs

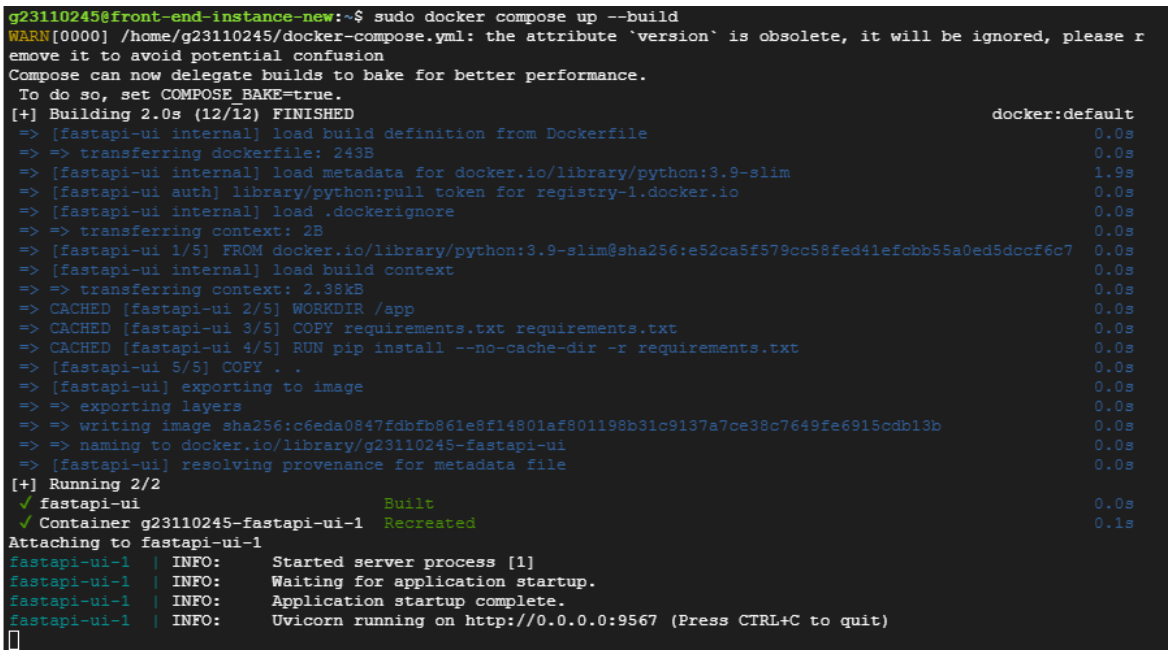
```
g23110245@backend-instance:~/backend$ sudo docker-compose down -v --remove-orphans
WARN[0000] /home/g23110245/backend/docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion
[+] Running 1/1
 ✓ Volume backend_es_data  Removed
```

Now we build the Docker containers which use the information from the docker-compose.yml and commands from Dockerfile ,which are present in both frontend and backend by using

command

```
sudo docker-compose up --build
```

Now this outputs in frontend as



And in backend as

```
g23110245@backend-instance:~/backend$ sudo docker-compose up --build
WARN[0000] /home/g23110245/backend/docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion
Compose can now delegate builds to bake for better performance.
 To do so, set COMPOSE_BAKE=true.
[+] Building 4.4s (11/11) FINISHED                                                                                                                  docker:default
 => [fastapi-app internal] load build definition from Dockerfile                                                                                             0.0s
 => => transferring dockerfile: 32B                                                                                                                          0.0s
 => [fastapi-app internal] load .dockerignore                                                                                                                0.0s
 => => transferring context: 2B                                                                                                                              0.0s
 => [fastapi-app internal] load metadata for docker.io/library/python:3.9-slim                                                                               1.9s
 => [fastapi-app auth] library/python:pull token for registry-1.docker.io                                                                                    0.0s
 => [fastapi-app 1/5] FROM docker.io/library/python:3.9-slim@sha256:e52ca5f579cc58fed41efcbb55a0ed5dccf6c7a156cba76acfb4ab42fc19dd00                         0.0s
 => [fastapi-app internal] load build context                                                                                                                0.2s
 => => transferring context: 294.45kB                                                                                                                        0.2s
 => CACHED [fastapi-app 2/5] WORKDIR /app                                                                                                                    0.0s
 => CACHED [fastapi-app 3/5] COPY requirements.txt .                                                                                                         0.0s
 => CACHED [fastapi-app 4/5] RUN pip install --no-cache-dir -r requirements.txt                                                                              0.0s
 => [fastapi-app 5/5] COPY . .                                                                                                                               1.6s
 => [fastapi-app] exporting to image                                                                                                                         0.6s
 => => exporting layers                                                                                                                                      0.6s
 => => writing image sha256:46addc690fea2511ec085fafa6a0a6728854af48bac36bfcc7623094c389a4c2                                                                 0.0s
 => => naming to docker.io/library/backend-fastapi-app                                                                                                       0.0s
[+] Running 5/5
 ✓ fastapi-app                          Built                                                                                                               0.0s
 ✓ Network elastic-network              Created                                                                                                             0.1s
 ✓ Volume "backend_es_data"             Created                                                                                                             0.0s
 ✓ Container backend-elasticsearch-1    Created                                                                                                             0.1s
 ✓ Container fastapi-2                  Created                                                                                                             0.0s
Attaching to elasticsearch-1, fastapi-2
```

```
elasticsearch-1  | {"type": "server", "timestamp": "2025-03-27T17:45:06,809Z", "level": "INFO", "component": "o.e.x.i.a.TransportPutLifecycleAction", "cluster.name": "do
r-cluster", "node.name": "ca47b2f9fa88", "message": "adding index lifecycle policy [.deprecation-indexing-ilm-policy]", "cluster.uuid": "AL9nK5afRKSQsxEOYD-irg", "node.i
 "UceZTezDRVaZ0aybaXhRDA"  }
elasticsearch-1  | {"type": "server", "timestamp": "2025-03-27T17:45:06,899Z", "level": "INFO", "component": "o.e.x.i.a.TransportPutLifecycleAction", "cluster.name": "do
r-cluster", "node.name": "ca47b2f9fa88", "message": "adding index lifecycle policy [.fleet-actions-results-ilm-policy]", "cluster.uuid": "AL9nK5afRKSQsxEOYD-irg", "node.
: "UceZTezDRVaZ0aybaXhRDA"  }
elasticsearch-1  | {"type": "server", "timestamp": "2025-03-27T17:45:06,988Z", "level": "INFO", "component": "o.e.i.g.GeoIpDownloader", "cluster.name": "docker-cluster",
ode.name": "ca47b2f9fa88", "message": "updating geoip databases", "cluster.uuid": "AL9nK5afRKSQsxEOYD-irg", "node.id": "UceZTezDRVaZ0aybaXhRDA"  }
elasticsearch-1  | {"type": "server", "timestamp": "2025-03-27T17:45:06,990Z", "level": "INFO", "component": "o.e.i.g.GeoIpDownloader", "cluster.name": "docker-cluster",
ode.name": "ca47b2f9fa88", "message": "fetching geoip databases overview from [https://geoip.elastic.co/v1/database?elastic_geoip_service_tos=agree]", "cluster.uuid": "A
K5afRKSQsxEOYD-irg", "node.id": "UceZTezDRVaZ0aybaXhRDA"  }
elasticsearch-1  | {"type": "server", "timestamp": "2025-03-27T17:45:07,389Z", "level": "INFO", "component": "o.e.l.LicenseService", "cluster.name": "docker-cluster", "n
.name": "ca47b2f9fa88", "message": "license [88d7e31c-bcc5-4fa8-9386-aec38fdc539c] mode [basic] - valid", "cluster.uuid": "AL9nK5afRKSQsxEOYD-irg", "node.id": "UceZTezDR
0aybaXhRDA"  }
elasticsearch-1  | {"type": "server", "timestamp": "2025-03-27T17:45:09,956Z", "level": "INFO", "component": "o.e.c.m.MetadataCreateIndexService", "cluster.name": "docke
luster", "node.name": "ca47b2f9fa88", "message": "[.geoip_databases] creating index, cause [auto(bulk api)], templates [], shards [1]/[0]", "cluster.uuid": "AL9nK5afRKSQ
OYD-irg", "node.id": "UceZTezDRVaZ0aybaXhRDA"  }
elasticsearch-1  | {"type": "server", "timestamp": "2025-03-27T17:45:10,444Z", "level": "INFO", "component": "o.e.c.r.a.AllocationService", "cluster.name": "docker-clust
 "node.name": "ca47b2f9fa88", "message": "Cluster health status changed from [YELLOW] to [GREEN] (reason: [shards started [[.geoip_databases][0]]]).", "cluster.uuid": "
aK5afRKSQsxEOYD-irg", "node.id": "UceZTezDRVaZ0aybaXhRDA"  }
fastapi-2        | INFO:elastic_transport.transport:HEAD http://elasticsearch:9200/ [status:200 duration:0.215s]
fastapi-2        | INFO:__main__:Successfully connected to Elasticsearch
fastapi-2        | INFO:elastic_transport.transport:HEAD http://elasticsearch:9200/myindex [status:404 duration:0.039s]
elasticsearch-1  | {"type": "server", "timestamp": "2025-03-27T17:45:11,330Z", "level": "INFO", "component": "o.e.c.m.MetadataCreateIndexService", "cluster.name": "docke
luster", "node.name": "ca47b2f9fa88", "message": "[myindex] creating index, cause [api], templates [], shards [1]/[1]", "cluster.uuid": "AL9nK5afRKSQsxEOYD-irg", "node.i
 "UceZTezDRVaZ0aybaXhRDA"  }
fastapi-2        | INFO:elastic_transport.transport:PUT http://elasticsearch:9200/myindex [status:200 duration:0.183s]
fastapi-2        | INFO:__main__:Created index myindex
fastapi-2        | INFO:elastic_transport.transport:PUT http://elasticsearch:9200/myindex/_doc/1 [status:201 duration:0.041s]
fastapi-2        | INFO:elastic_transport.transport:PUT http://elasticsearch:9200/myindex/_doc/2 [status:201 duration:0.036s]
fastapi-2        | INFO:elastic_transport.transport:PUT http://elasticsearch:9200/myindex/_doc/3 [status:201 duration:0.012s]
fastapi-2        | INFO:elastic_transport.transport:PUT http://elasticsearch:9200/myindex/_doc/4 [status:201 duration:0.011s]
fastapi-2        | INFO:__main__:Inserted initial sample documents
fastapi-2        | INFO:     Started server process [8]
fastapi-2        | INFO:     Waiting for application startup.
fastapi-2        | INFO:     Application startup complete.
fastapi-2        | INFO:     Uvicorn running on http://0.0.0.0:9567 (Press CTRL+C to quit)
elasticsearch-1  | {"type": "server", "timestamp": "2025-03-27T17:45:11,976Z", "level": "INFO", "component": "o.e.i.g.DatabaseNodeService", "cluster.name": "docker-clust
 "node.name": "ca47b2f9fa88", "message": "retrieve geoip database [GeoLite2-ASN.mmdb] from [.geoip_databases] to [/tmp/elasticsearch-16819359439271066139/geoip-database
```

This now completes creating all of our containers; and it is clearly visible that uvicorn has being started on the port http://0.0.0.0:9567 in both the cases. So, we can directly access the frontend webpage by the external ip of the webpage which is like

Home

Enter text you wanna insert or search

Insert Document | Search Document

Also, we can use the command **curl http://0.0.0.0:9567
If we get the HTML code it confirms that the webpage is working

And we got that :)

```
g23110245@front-end-instance-new:~$ curl http://0.0.0.0:9567

<!DOCTYPE html>
<html>
<head>
    <title>Home</title>
</head>
<body>
    <h1>Home</h1>
    <input size=100 type="text" id="docInput" placeholder="Enter text you wanna insert or search"><br><br>
    <button onclick="insertDocument()">Insert Document</button>
    <button onclick="searchDocument()">Search Document</button>
    <p id="output"></p>
    <script>
        async function insertDocument() {
            let text = document.getElementById('docInput').value;
            let response = await fetch('http://34.100.235.148:9567/insert', {
                method: 'POST',
                headers: { 'Content-Type': 'application/json' },
                body: JSON.stringify({ text: text })
            });
            let data = await response.json();
            document.getElementById('output').innerText = JSON.stringify(data, null, 2);
        }

        async function searchDocument() {
            let query = document.getElementById('docInput').value;
            let response = await fetch('http://34.100.235.148:9567/search?query=' + encodeURIComponent(query));
            let data = await response.json();
            document.getElementById('output').innerText = JSON.stringify(data, null, 2);
        }
    </script>
</body>
</html>
```

In backend we run elastic search on its genral port 9200 So, we use that port here and get the information which suggests that everything is correct.

```
g23110245@backend-instance:~/backend$ curl http://0.0.0.0:9567
{"detail":"Not Found"}g23110245@backend-instance:~/backend$ curl http://0.0.0.0:9200
{
  "name" : "ca47b2f9fa88",
  "cluster_name" : "docker-cluster",
  "cluster_uuid" : "AL9nK5afRKSQsxEOYD-irg",
  "version" : {
    "number" : "7.17.9",
    "build_flavor" : "default",
    "build_type" : "docker",
    "build_hash" : "ef48222227ee6b9e70e502f0f0daa52435ee634d",
    "build_date" : "2023-01-31T05:34:43.305517834Z",
    "build_snapshot" : false,
    "lucene_version" : "8.11.1",
    "minimum_wire_compatibility_version" : "6.8.0",
    "minimum_index_compatibility_version" : "6.0.0-beta1"
  },
  "tagline" : "You Know, for Search"
```

Important ScreenShots

Dockerfiles

We tried to keep them as simple aspossible which becomes easy for debugging later

```
g23110245@backend-instance:~/backend$ cat Dockerfile
FROM python:3.9-slim
WORKDIR /app
COPY requirements.txt .
RUN pip install --no-cache-dir -r requirements.txt
COPY . .
EXPOSE 9567
ENV PYTHONUNBUFFERED=1
CMD ["uvicorn", "app:app", "--host", "0.0.0.0", "--port", "9567"]
```

```
g23110245@front-end-instance-new:~$ cat Dockerfile
FROM python:3.9-slim

WORKDIR /app

COPY requirements.txt requirements.txt
RUN pip install --no-cache-dir -r requirements.txt

COPY . .

CMD ["uvicorn", "main:app", "--host", "0.0.0.0", "--port", "9567"]
```

The building commands are the outputs are being shared in the last section of the document. For composing the document the most important we require is docker-compose.yml for both frontend and backend which stores the configuration for the containers which are to be built here are their screenshots

```
g23110245@backend-instance:~/backend$ cat docker-compose.yml
version: '3.8'

services:
  fastapi-app:
    container_name: fastapi-2
    build: .
    ports:
      - "9567:9567"
    depends_on:
      elasticsearch:
        condition: service_healthy
    volumes:
      - ./logs.json:/app/logs.json
    networks:
      - esnet
    environment:
      - ELASTICSEARCH_HOST=elasticsearch
      - ELASTICSEARCH_PORT=9200
    command: ["sh", "-c", "sleep 15 && python app.py"]

  elasticsearch:
    image: docker.elastic.co/elasticsearch/elasticsearch:7.17.9
    #this environment ensures low memory and doesnt uses unncessry space
    environment:
      - discovery.type=single-node
      - ES_JAVA_OPTS=-Xms256m -Xmx256m
```

```
  elasticsearch:
    image: docker.elastic.co/elasticsearch/elasticsearch:7.17.9
    #this environment ensures low memory and doesnt uses unncessry space
    environment:
      - discovery.type=single-node
      - ES_JAVA_OPTS=-Xms256m -Xmx256m
      - xpack.security.enabled=false
      - xpack.ml.enabled=false
      - xpack.graph.enabled=false
      - xpack.watcher.enabled=false
      - cluster.routing.allocation.disk.threshold_enabled=false
    ports:
      - "9200:9200"
    volumes:
      - es_data:/usr/share/elasticsearch/data
    networks:
      - esnet
    healthcheck:
      test: ["CMD-SHELL", "curl -sSf http://localhost:9200/_cluster/health?wait_for_status=yellow&timeout=2m ||
exit 1"]
      interval: 10s
      timeout: 180s
      retries: 15
    ulimits:
      memlock:
        soft: -1
        hard: -1

volumes:
  es_data:
    driver: local

networks:
  esnet:
    driver: bridge
    name: elastic-network
```

For the frontend

```
g23110245@front-end-instance-new:~$ cat docker-compose.yml
version: '3.8'

services:
  fastapi-ui:
    build: .
    ports:
      - "9567:9567"
    environment:
      - BACKEND_URL=http://34.100.235.148:9567
    restart: unless-stopped
```

For separating FastApi and ElasticSearch Ports

The docker compose.yml files given in the previous parts have a parameters which describe their particular ports; So for fastapi it is 9567 and for Elasticsearch it becomes 9200. The curl command is in the same direction of thought.

Also while calling elasticsearch we called on the 9200 port of localhost like this.

```
ES_HOST = "elasticsearch"
ES_PORT = 9200
INDEX_NAME = "myindex"
LOG_FILE = "logs.json"

def get_es_connection():
    """Create and verify Elasticsearch connection"""
    try:
        es = Elasticsearch(
            [f"http://{ES_HOST}:{ES_PORT}"],
            request_timeout=30,
            max_retries=3,
            retry_on_timeout=True,
            sniff_on_start=False,
            sniff_on_node_failure=False
        )
```

netstat -antp | grep LISTEN

```
g23110245@front-end-instance-new:~$ netstat -antp | grep LISTEN
(Not all processes could be identified, non-owned process info
 will not be shown, you would have to be root to see it all.)
tcp        0      0 0.0.0.0:20202           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:5355            0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.54:53           0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:9567            0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      -
tcp6       0      0 :::20201                :::*                    LISTEN      -
tcp6       0      0 ::1:25                  :::*                    LISTEN      -
tcp6       0      0 :::5355                 :::*                    LISTEN      -
tcp6       0      0 :::9567                 :::*                    LISTEN      -
tcp6       0      0 :::22                   :::*                    LISTEN      -
```

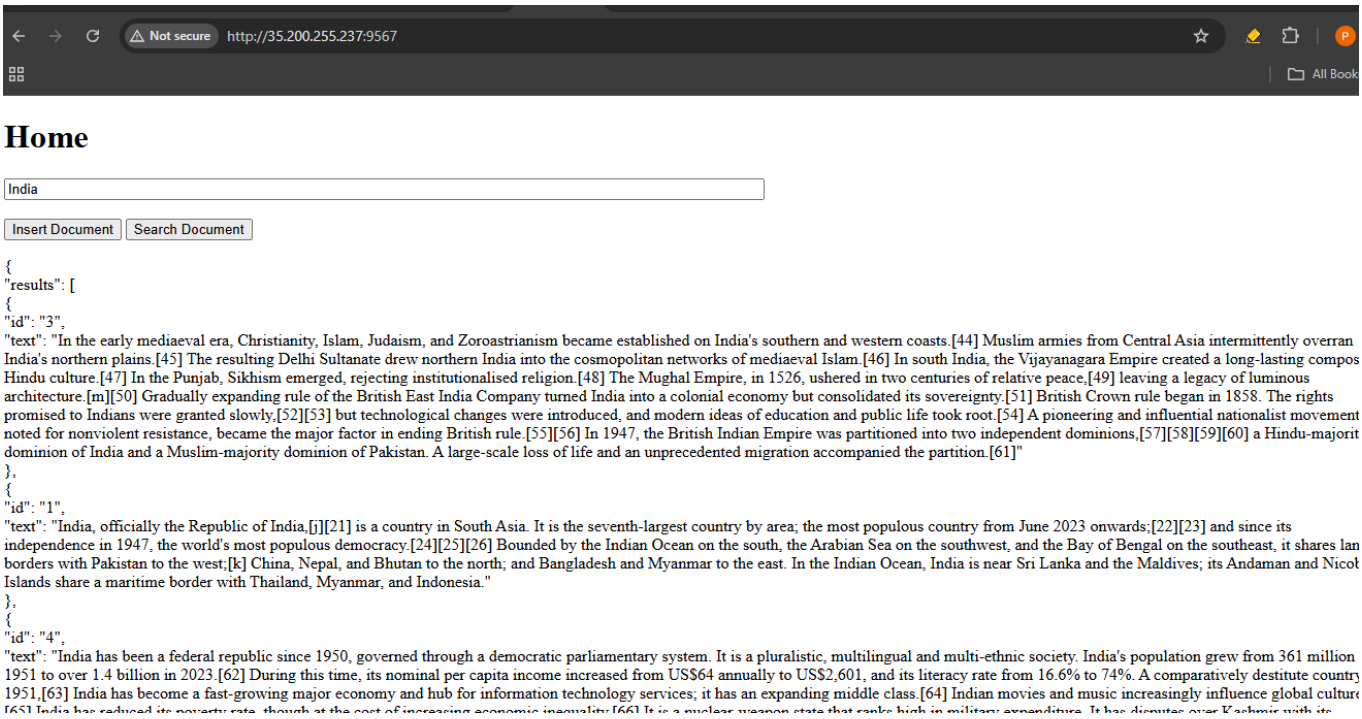
```
g23110245@backend-instance:~/backend$ netstat -antp | grep LISTEN
(Not all processes could be identified, non-owned process info
 will not be shown, you would have to be root to see it all.)
tcp        0      0 0.0.0.0:20202           0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:9567            0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:5355            0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:9200            0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.54:53           0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:35983         0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      -
tcp6       0      0 :::20201                :::*                    LISTEN      -
tcp6       0      0 ::1:25                  :::*                    LISTEN      -
tcp6       0      0 :::9567                 :::*                    LISTEN      -
tcp6       0      0 :::5355                 :::*                    LISTEN      -
tcp6       0      0 :::9200                 :::*                    LISTEN      -
tcp6       0      0 :::22                   :::*                    LISTEN      -
```

docker images

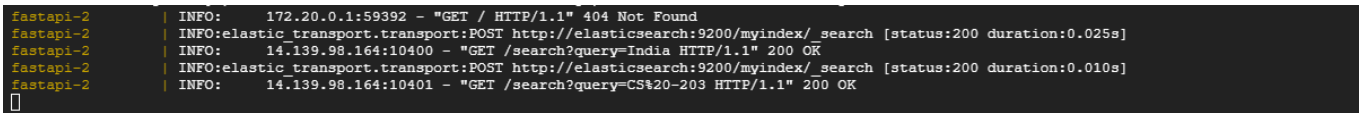
```
g23110245@backend-instance:~/backend$ sudo docker images
REPOSITORY                                   TAG       IMAGE ID       CREATED          SIZE
backend-fastapi-app                          latest    46addc690fea   23 minutes ago   236MB
chinmayp995/fastapi                          1         c36aec59ba04   5 hours ago      236MB
chinmayp995/fastapi                          2         c36aec59ba04   5 hours ago      236MB
<none>                                       <none>    a49a915eadee   6 hours ago      236MB
<none>                                       <none>    aca7271901be   11 hours ago     236MB
<none>                                       <none>    686286ee4e09   11 hours ago     236MB
<none>                                       <none>    e71f140a4265   11 hours ago     236MB
docker.elastic.co/elasticsearch/elasticsearch   7.17.9    7e1effda4391   2 years ago      620MB
chinmayp995/elasticsearch                    latest    7e1effda4391   2 years ago      620MB
docker.elastic.co/elasticsearch/elasticsearch   7.17.0    6fe993d6e7ed   3 years ago      612MB
```

```
g23110245@front-end-instance-new:~$ sudo docker images
REPOSITORY              TAG       IMAGE ID       CREATED          SIZE
g23110245-fastapi-ui    latest    c6eda0847fdb   24 minutes ago   147MB
chinmayp995/fastapi     1         a9d90d68f858   6 hours ago      147MB
chinmayp995/fastapi     2         a9d90d68f858   6 hours ago      147MB
<none>                  <none>    d8feb2e8917d   11 hours ago     147MB
<none>                  <none>    8901ab774941   11 hours ago     147MB
<none>                  <none>    cc2a9bf13aa0   11 hours ago     147MB
```

Testing of get query



We get this logging in the fastapi output



Thus Get working properly!

TEsting of Insert

Home

Just chill !

Insert Document | Search Document

{
"message": "Document inserted successfully"
}

Home

chill

Insert Document | Search Document

{
"results": [
{
"id": "_rXL2JUBKz0BhoN9ukjy",
"text": "Just chill !"
}
]
}


```
fastapi-2       | INFO:elastic_transport.transport:POST http://elasticsearch:9200/myindex/_doc?refresh=true [status:201 duration:0.024s]
fastapi-2       | INFO:     14.139.98.164:10402 - "POST /insert HTTP/1.1" 200 OK
fastapi-2       | INFO:     14.139.98.164:10403 - "GET /search?query=chill HTTP/1.1" 200 OK
fastapi-2       | INFO:elastic_transport.transport:POST http://elasticsearch:9200/myindex/_search [status:200 duration:0.008s]
```

Docker Inspect

In Backend

```
g23110245@backend-instance:~/backend$ sudo docker inspect backend-fastapi-app

[
    {
        "Id": "sha256:46addc690fea2511ec085fafa6a0a6728854af48bac36bfcc7623094c389a4c2",
        "RepoTags": [
            "backend-fastapi-app:latest"
        ],
        "RepoDigests": [],
        "Parent": "",
        "Comment": "buildkit.dockerfile.v0",
        "Created": "2025-03-27T17:43:02.240168232Z",
        "Container": "",
        "ContainerConfig": {
            "Hostname": "",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": null,
            "Cmd": null,
            "Image": "",
            "Volumes": null,
            "WorkingDir": "",
            "Entrypoint": null,
            "OnBuild": null,
            "Labels": null
        },
        "DockerVersion": "",
        "Author": "",
```

For elasticsearch

```
g23110245@backend-instance:~/backend$ sudo docker inspect 7e1effda4391
[
    {
        "Id": "sha256:7e1effda4391287fa2f751348ac771b6396e0107c2be7578f1fc5b93941eb514",
        "RepoTags": [
            "docker.elastic.co/elasticsearch/elasticsearch:7.17.9",
            "chinmayp995/elasticsearch:latest"
        ],
        "RepoDigests": [
            "docker.elastic.co/elasticsearch/elasticsearch@sha256:59b37f77bd8b015d5b60f75bebb22d06028f7f
559d2b7c16ece74db",
            "chinmayp995/elasticsearch@sha256:56789f44fd8c451fdeb40a095c5089367e588c7a24e0a03cdbd6ba53eb
        ],
        "Parent": "",
        "Comment": "",
        "Created": "2023-01-31T05:40:18.764053573Z",
        "Container": "12b0f23b3d54e6f18210828f71cafda1b50ffe97c4b93318f27ddbd7132db461",
```

In frontend

```
g23110245@front-end-instance-new:~$ sudo docker inspect  c6eda0847fdb
[
    {
        "Id": "sha256:c6eda0847fdbfb861e8f14801af801198b31c9137a7ce38c7649fe6915cdb13b",
        "RepoTags": [
            "g23110245-fastapi-ui:latest"
        ],
        "RepoDigests": [],
        "Parent": "",
        "Comment": "buildkit.dockerfile.v0",
        "Created": "2025-03-27T17:42:03.852195621Z",
        "DockerVersion": "",
        "Author": "",
        "Config": {
            "Hostname": "",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
```

docker ps -a (to get all containers)

```
g23110245@backend-instance:~/backend$ sudo docker ps -a
CONTAINER ID   IMAGE                                                COMMAND                  CREATED
STATUS                    PORTS                                                 NAMES
3f498d0c3d53   backend-fastapi-app                                  "sh -c 'sleep 15 && …"   30 minutes ago
Up 30 minutes             0.0.0.0:9567->9567/tcp, :::9567->9567/tcp             fastapi-2
ca47b2f9fa88   docker.elastic.co/elasticsearch/elasticsearch:7.17.9   "/bin/tini -- /usr/l…"   30 minutes ago
Up 30 minutes (healthy)   0.0.0.0:9200->9200/tcp, :::9200->9200/tcp, 9300/tcp   backend-elasticsearch-1
g23110245@backend-instance:~/backend$
```

```
g23110245@front-end-instance-new:~$ sudo docker ps -a
CONTAINER ID   IMAGE                  COMMAND                  CREATED          STATUS          PORTS
                          NAMES
41b66496bef0   g23110245-fastapi-ui   "uvicorn main:app --…"   33 minutes ago   Up 33 minutes   0.0.0.0:9567->95
67/tcp, [::]:9567->9567/tcp   g23110245-fastapi-ui-1
g23110245@front-end-instance-new:~$
```

(Total 3 containers)

Optimisation

For optimisation in this assignment we have used python3.9- slim which is a lighter and stable version than the current other versions.

Also, for elastic search image we used elasticsearch 7.17.9 which is quite newer, stable and less memory consuming than the other versions we tried like 8.x. We also tried using 2.x and 3.x which were only 80 MB but they had other vulnerabilities and didnt worked well with the current python versions.

Also, to decrease the memory environment we setup the envronment for elasticsearch in yml like this which reduced the constraints a lot. We also removed the unncessary things like ml, graph, watcher etc. which were taking extra space.

```
environment:
      - discovery.type=single-node
      - ES_JAVA_OPTS=-Xms256m -Xmx256m
```

```
      - xpack.security.enabled=false
      - xpack.ml.enabled=false
      - xpack.graph.enabled=false
      - xpack.watcher.enabled=false
      - cluster.routing.allocation.disk.threshold_enabled=false
```

To know the time taken by the container we used command

```
time docker run --rm busybox true
```

And got surprisingly less time for starting the container!!