Table of Contents

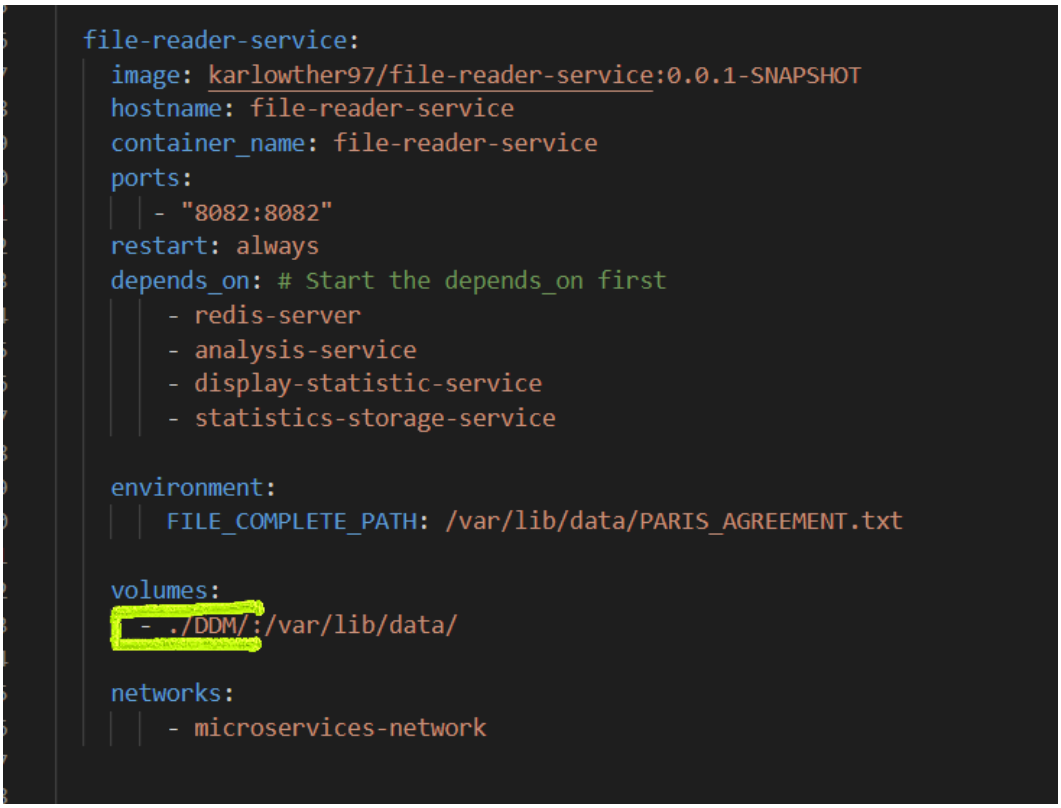
1. Instructions for Running Application

1.1. Docker must install

First step to you have must install docker in your machine.

1.2. Check and Setup Docker Compose File

First you needs to check you have docker-compose file current folder must DDM folder have must PARIS_AGREEMENT.txt with no space in the file.



```yaml
file-reader-service:
  image: karlowther97/file-reader-service:0.0.1-SNAPSHOT
  hostname: file-reader-service
  container_name: file-reader-service
  ports:
    - "8082:8082"
  restart: always
  depends_on: # Start the depends_on first
      - redis-server
      - analysis-service
      - display-statistic-service
      - statistics-storage-service

  environment:
      FILE_COMPLETE_PATH: /var/lib/data/PARIS_AGREEMENT.txt

  volumes:
    - ./DDM/:/var/lib/data/

  networks:
      - microservices-network
```

For two your words choice you can add your choice for word analysis.

You can add in against mention screen shot with highlight color.

```yaml
analysis-service:
  image: karlowther97/analysis-service:0.0.1-SNAPSHOT
  hostname: analysis-service
  container_name: analysis-service
  ports:
    - "8081:8081"
    - "9095:9095"
  restart: always
  depends_on: # Start the depends_on first
    - redis-server
    - statistics-storage-service
  environment:
    WORD_CHOICE_1: test1
    WORD_CHOICE_2: test2

  networks:
    - microservices-network
```

1.3. Login with Docker

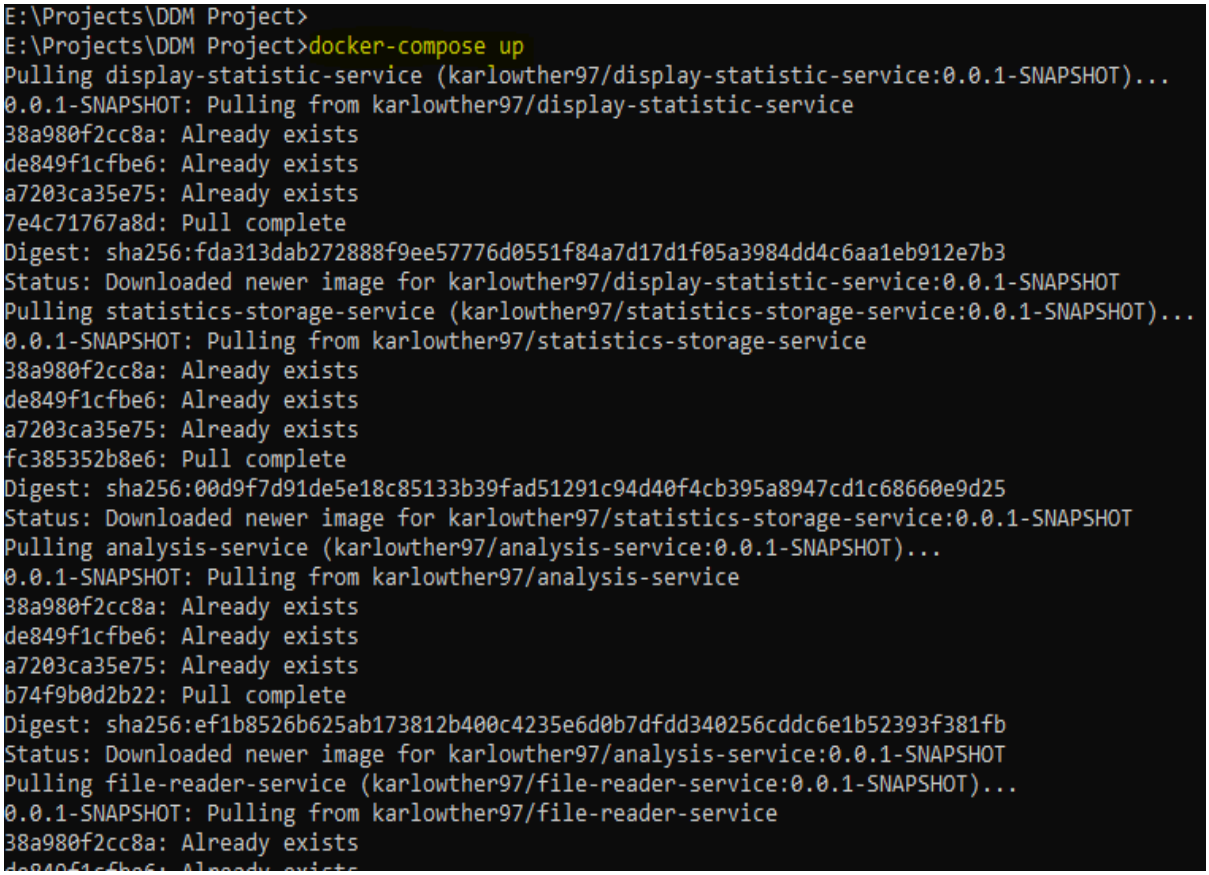
For docker login, you need to open cmd than you need to enter the mention command in screenshot.

```
E:\Projects\DDM Project>
E:\Projects\DDM Project>
E:\Projects\DDM Project>docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com to create one.
Username: karlowther97
Password:
Login Succeeded

Logging in with your password grants your terminal complete access to your account.
For better security, log in with a limited-privilege personal access token. Learn more at https://docs.docker.com/go/access-tokens/

E:\Projects\DDM Project>
```

1.4. Running Docker Compose File

Before running docker compose you must login and have correct configuration for your file folder.

For running, you need to enter the mention commands in screenshot.



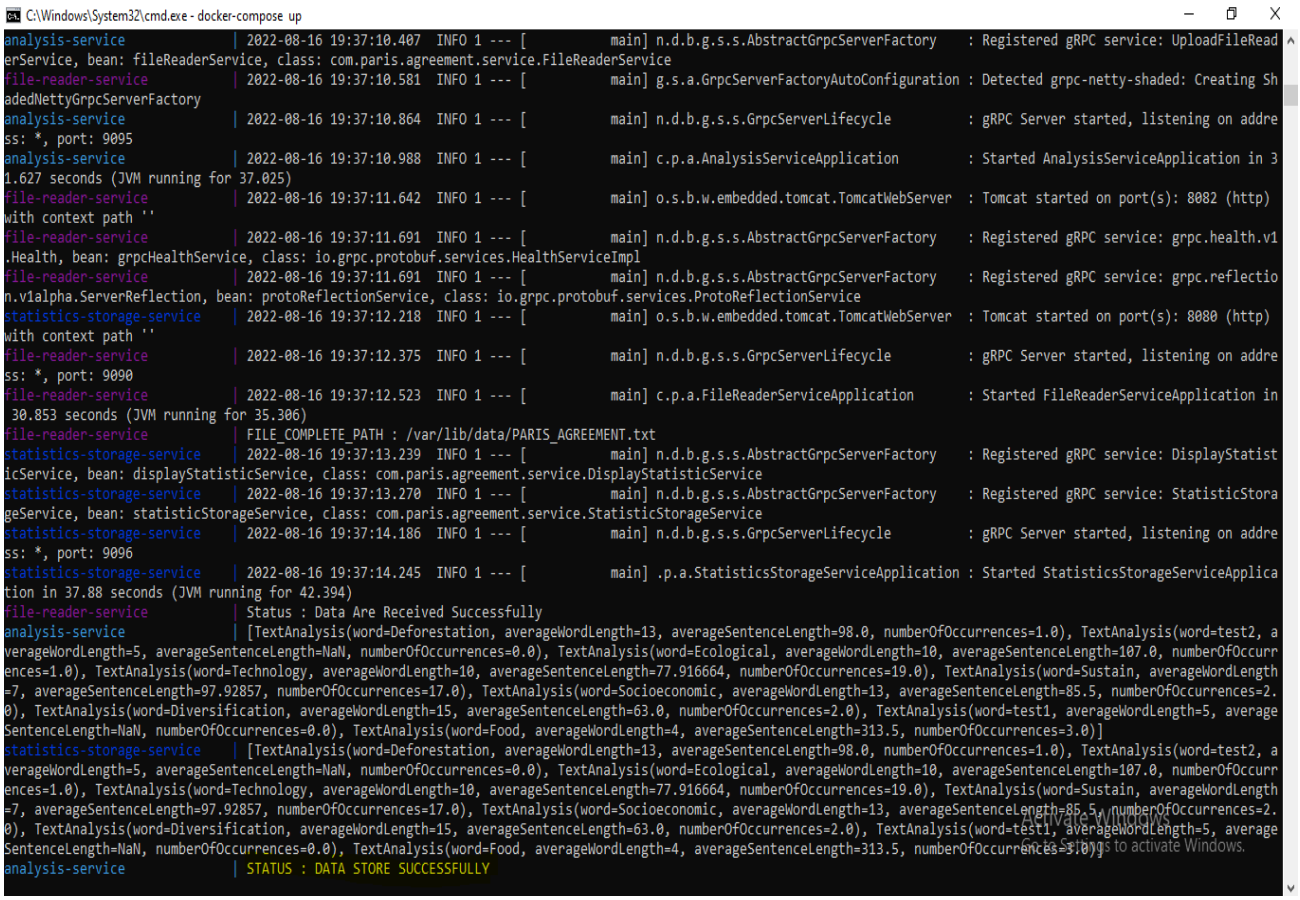
First, time its pull or download all images from your docker hub account and then create and run its container.

You need to wait for first time because it takes some time on download images from your account.

1.5. Check Statistic on Web App

For check statistic on web app, you to wait until this message show in the logs of docker compose.

This is highlight in screenshot.



After the message shown then need open your web browser and type ***localhost:8080*** and hit enter.

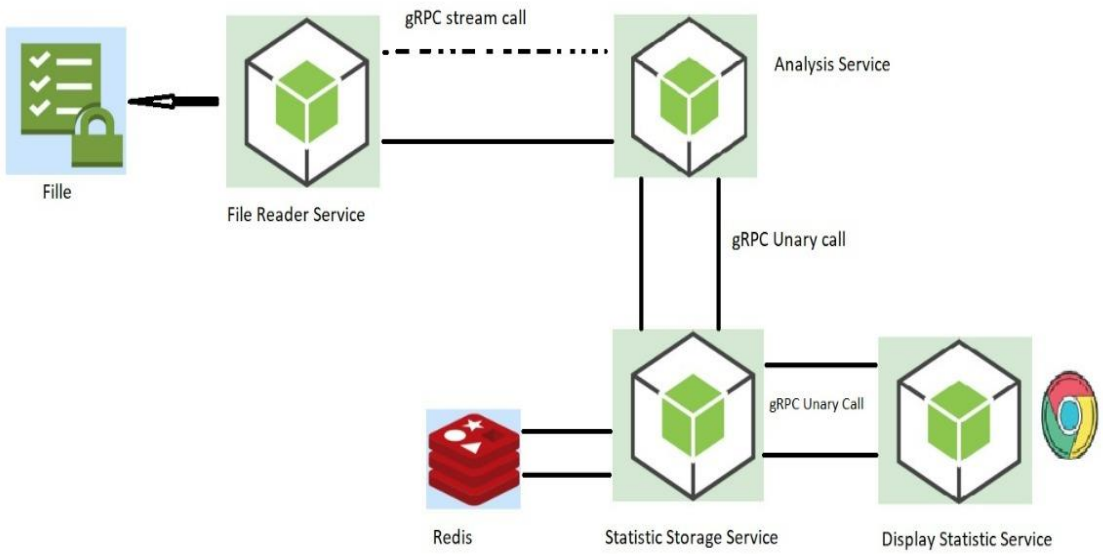
Therefore, you check your analysis on web app.

localhost:8083

Paris Agreement Text Analysis

Word	Average Word Length	Average Sentence Length	Number Of Occurrences
Ecological	10	107.0	1.0
test1	5	NaN	0.0
Technology	10	77.916664	19.0
Deforestation	13	98.0	1.0
test2	5	NaN	0.0
Socioeconomic	13	85.5	2.0
Sustain	7	97.92857	17.0
Diversification	15	63.0	2.0
Food	4	313.5	3.0

2. Application Architecture



3. Choosing the various integration and storage mechanisms

3.1. Integration

For integration of service we use gRPC (remote procedure call) instated of Rest API.

gRPC is large-scale microservices connections, real-time communication, low-power, low-bandwidth systems, and multi-language environments. Unlike REST, gRPC makes the most out of HTTP/2, with multiplexed streaming and binary protocol framing.

We use tow type of call in services communication with each other.

These call name is show below

- Unary call
- Client stream call

The unary is used three places in microservices communication and its use for single request and response.

While Client streaming is use just one place in microservices communication and it is use

for client multiple request and one response from server.

Therefore, I use client streaming when filer-reader-service is reader the file and send each word to analysis-service for analyzing.

While rest of APIs call is use as, unary calls because that APIs call is lightweight calls.

3.2. Storage

Redis is use here for storage the statistic of file text analysis.

Redis is a popular open-source database server for small and large applications alike. It is design to be fast, reliable, and high performing.

Redis is use for various purposes including caching, key-value storage, in-memory data structures and data structures with disk persistence.

Therefore, we use Redis storage here based upon above-mentioned points.

4. Test Architecture (compared with a more traditional monolith approach h)

4.1. Monolithic vs Microservices Architecture

Microservices architecture decomposes a large application into a set of small, independent services that can be developed, deployed, and scaled independently.

Monolithic applications are built as a single, large unit, and they typically easier to develop and test than microservices.

4.2. Advantages of Microservices Testing Over Monolithic

- Monolithic application cannot be develop and test reach module independently while Microservices this makes creating, testing, and deploying new features and fixes easier.

- Microservices are independently scalable if one service is experiencing high traffic, you can add more instances without affecting the other services. On the other hand, Monolithic applications because they can be scaled horizontally. Horizontal scalability means adding more servers to handle the load rather than scaling up (adding more CPU or memory to a single server)

- In Monolithic application, if there is an issue with one part of the system, it can be not easy to track down and fix. While it is easy to identify and fix the problem with microservices since each service runs in its process and is completely isolated from the others.

- With Microservices you can develop each service independently, which speeds up the development process and allows you to release new features more quickly while monolithic application can be time-consuming since all the code must be written at once.

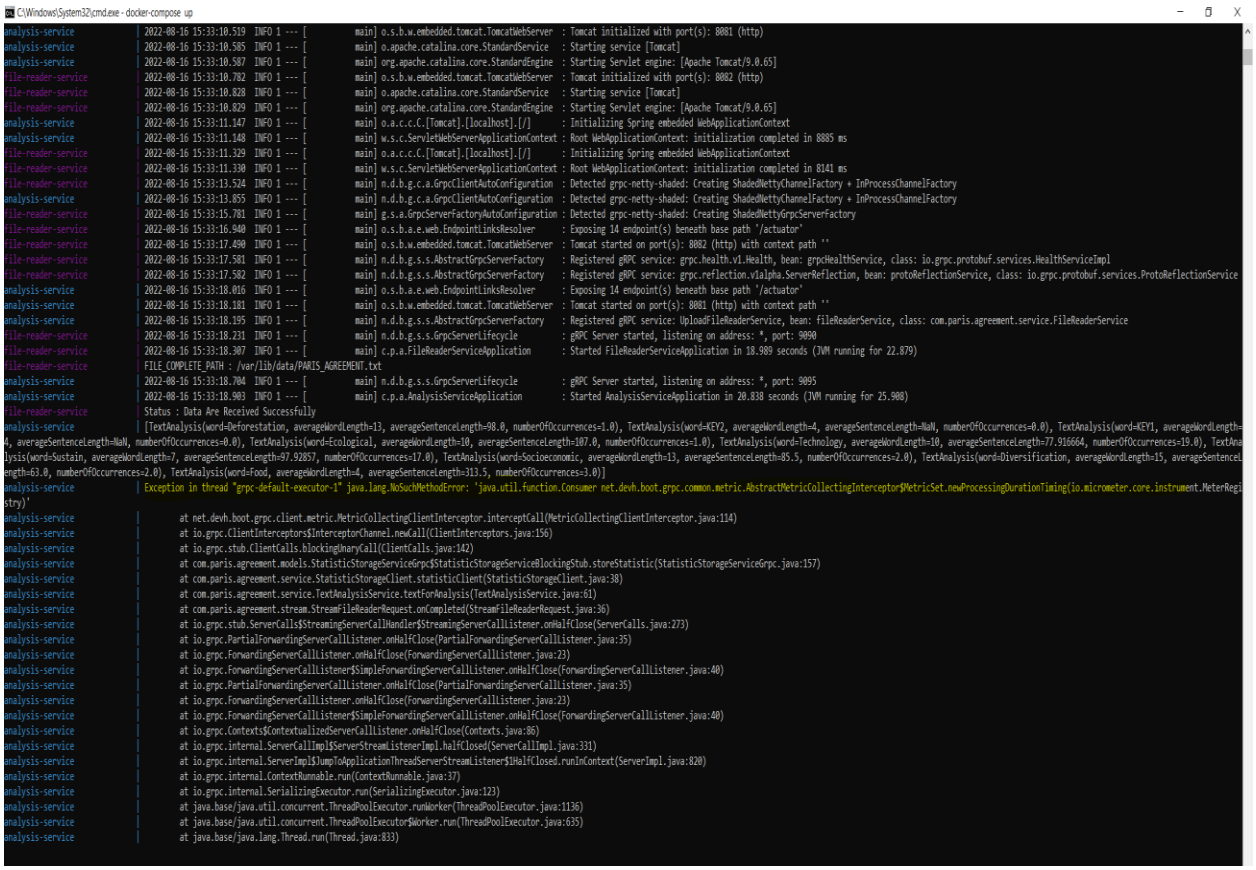
- Microservices easier to test each one thoroughly when you break an application down into smaller services. This increased test coverage and a higher degree of confidence in the quality of the code.

5. Monitoring of Microservices Architecture

For Monitoring of microservices I use Prometheus. Therefore, I add all configuration and dependency in each microservice. After configuration when I try run all microservices

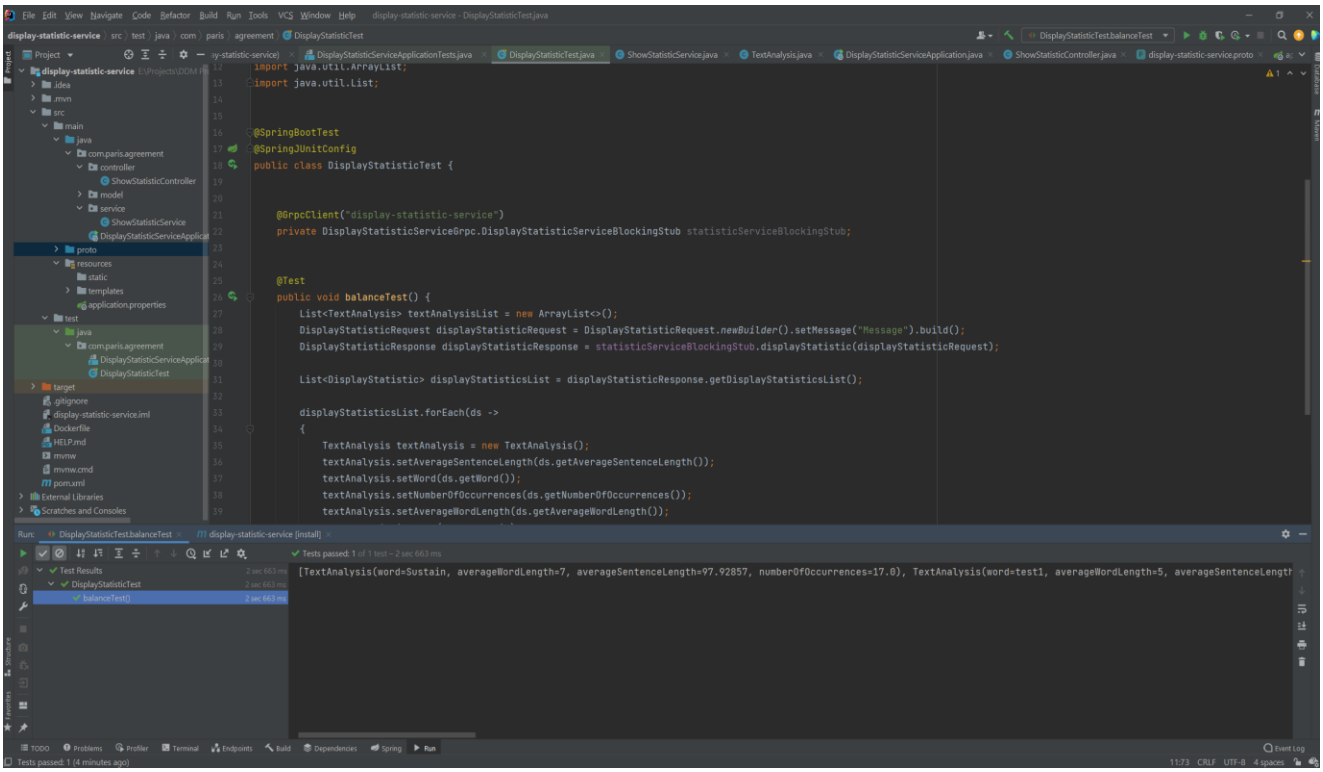
I got errors in each microservice logs. After investigate and do R&D about it. So most blogs and community says it is not possible in gRPC. Therefore, I left that part due to this issue.

The error log is show below screenshot.



6. Screenshots of Testing

6.1. Display Statistic Service



6.2. File Reader Service