Graded Assignment on MERN Application with Grafana and Prometheus

Step 1. MERN Application Setup:-

- Clone the below github repository GitHub repository:-
  git clone https://github.com/UnpredictablePrashant/TravelMemory.git

```
HP@LAPTOP-JRN3DQ8O MINGW64 /c/Gke-mern-deployment/Mern-promethius and graffana
$ git clone https://github.com/UnpredictablePrashant/TravelMemory.git
Cloning into 'TravelMemory'...
remote: Enumerating objects: 116, done.
remote: Counting objects: 100% (64/64), done.
remote: Compressing objects: 100% (49/49), done.
remote: Total 116 (delta 27), reused 23 (delta 15), pack-reused 52 (from 1)
Receiving objects: 100% (116/116), 198.32 KiB | 137.00 KiB/s, done.
Resolving deltas: 100% (37/37), done.
```

Step 2: Install Dependencies: -

- Navigate to the `frontend/` and `backend/` folders, and install dependencies: -

```
HP@LAPTOP-JRN3DQ8O MINGW64 /c/Gke-mern-deployment/Mern-promethius and graffana/TravelMemory/frontend (main)
$ npm install
npm warn deprecated stable@0.1.8: Modern JS already guarantees Array#sort() is a stable sort, so this library is deprecated. See the compatibility table
on MDN: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/sort#browser_compatibility
npm warn deprecated rollup-plugin-terser@7.0.2: This package has been deprecated and is no longer maintained. Please use @rollup/plugin-terser
npm warn deprecated sourcemap-codec@1.4.8: Please use @jridgewell/sourcemap-codec instead
npm warn deprecated w3c-hr-time@1.0.2: Use your platform's native performance.now() and performance.timeOrigin.
npm warn deprecated workbox-cacheable-response@6.6.0: workbox-background-sync@6.6.0
npm warn deprecated svgo@1.3.2: This SVGO version is no longer supported. Upgrade to v2.x.x.

added 1504 packages, and audited 1505 packages in 29s

236 packages are looking for funding
  run `npm fund` for details

27 vulnerabilities (1 low, 10 moderate, 15 high, 1 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
```

- For backend

```
HP@LAPTOP-JRN3DQ8O MINGW64 /c/Gke-mern-deployment/Mern-promethius and graffana/TravelMemory/backend (main)
$ npm install

added 117 packages, and audited 118 packages in 3s

13 packages are looking for funding
  run `npm fund` for details

13 vulnerabilities (1 low, 3 moderate, 8 high, 1 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
```

Step 3: Set up environment variables as below: -

```
 GNU nano 7.2                                                    .env
MONGO_URI='mongodb+srv://Rajpreetmongo:rajeemongo@cluster0.umhetht.mongodb.net/mernprofileservice'
PORT=3001
```

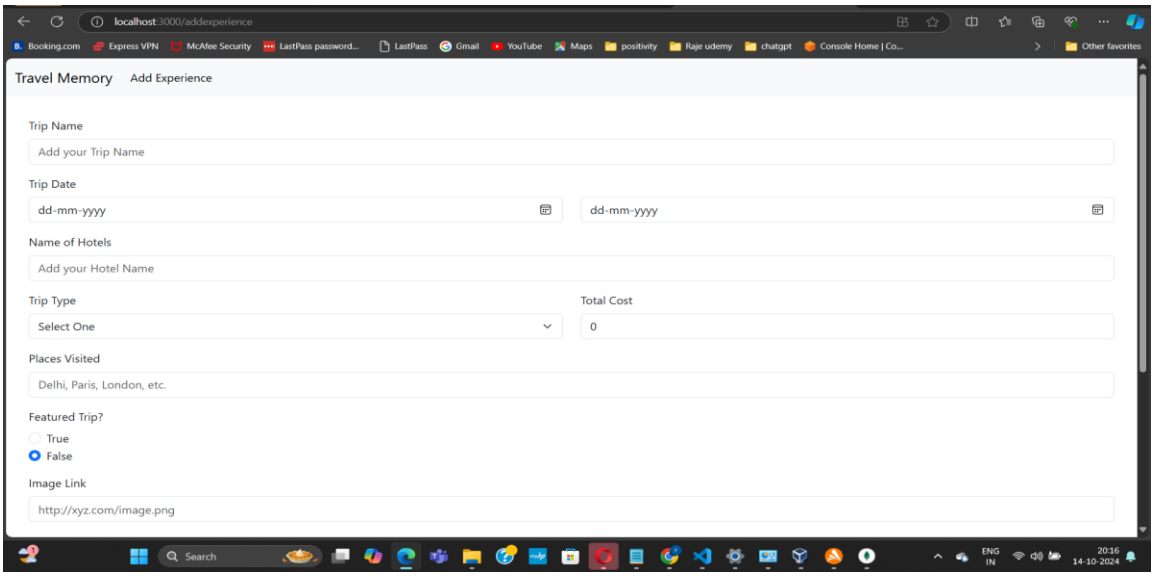
Create a .env file in both backend directories with necessary environment variables (e.g., MongoDB connection string, its port ,etc).

Step 5: Run the MERN application: -
Open two terminals: one for the frontend and one for the backend.

```
HP@LAPTOP-JRN3DQ8O MINGW64 /c/Gke-mern-deployment/Mern-promethius and graffana/TravelMemory/backend (main)
$ node index.js
Server started at http://localhost:3001
```

For frontend: -

You need to hit npm start in the frontend directory of the project.



Step6. Integrate Prometheus for Node.js Backend Metrics.

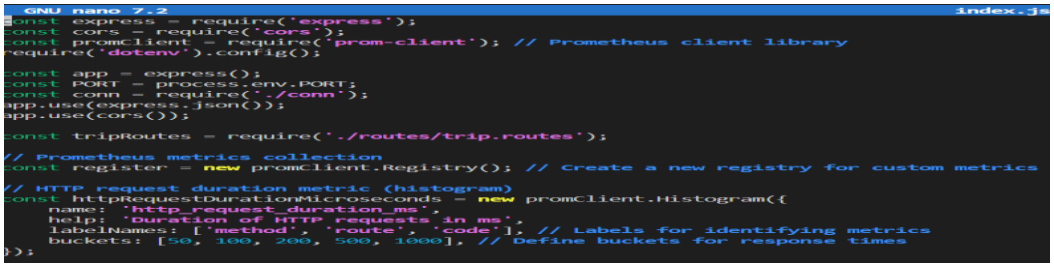
2. **Integrate Prometheus for Node.js Backend Metrics: -**

Step 1: Install Prometheus client library

In the backend, install the Prometheus client for Node.js. By hitting below command “npm install prom-client".

```
HP@LAPTOP-JRN3DQ8O MINGW64 /c/Gke-mern-deployment/Mern-promethius and graffana/TravelMemory/backend (main)
$ npm install prom-client

added 4 packages, and audited 122 packages in 5s

13 packages are looking for funding
  run `npm fund` for details
```

- Step 2: Expose Prometheus metrics
  In your `backend/server.js`, import the `prom-client` package and expose metrics like API response times, request counts, and error rates.

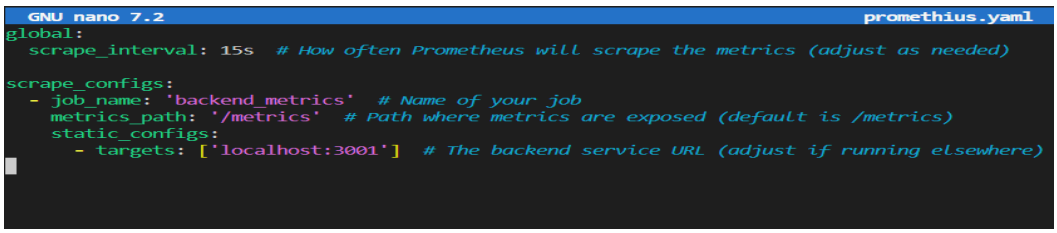


- Step 3: Set up MongoDB monitoring

Use MongoDB Exporter to monitor the database.

- Install MongoDB Exporter:
  “docker run -d --name mongodb_exporter -p 9216:9216 bitnami/mongodb-exporter”
- Configure Prometheus to scrape MongoDB metrics.

- Step 4: Configure Prometheus: -
  Set up a `prometheus.yml` configuration file to scrape your Node.js backend and MongoDB exporter.



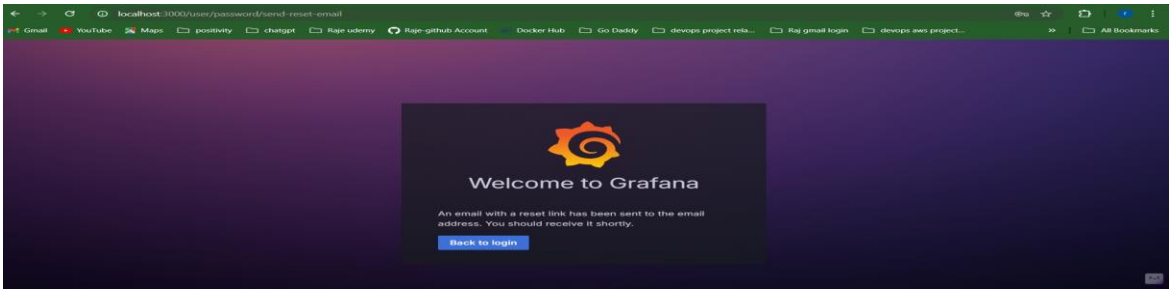
3. **Enhance Grafana Dashboards**

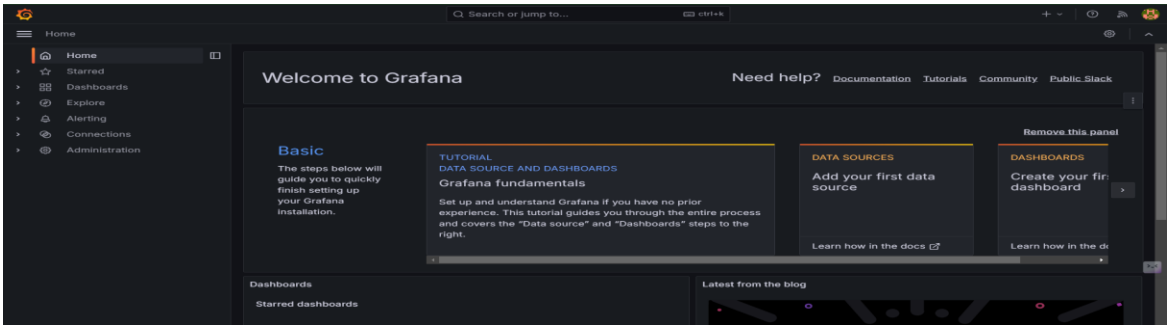
Step 1: Install and set up Grafana

Install Grafana and configure Prometheus as a data source.

```
HP@LAPTOP-JRN3DQ8O MINGW64 /c/Gke-mern-deployment/Mern-promethius and graffana/TravelMemory/backend (main)
$ docker run -d --name=grafana -p 3000:3000 grafana/grafana
Unable to find image 'grafana/grafana:latest' locally
latest: Pulling from grafana/grafana
27a3c8ebdfbf: Download complete
39aee5fd3406: Download complete
592f1e71407c: Download complete
ab3c28da242b: Download complete
bde37282dfba: Download complete
66aec874ce0c: Download complete
4abcf2066143: Download complete
ef2b3f3f597e: Download complete
b6982d0733af: Download complete
e4892977d944: Download complete
Digest: sha256:408afb9726de5122b00a2576763a8a57a3c86d5b0eff5305bc994ceb3eb96c3f
Status: Downloaded newer image for grafana/grafana:latest
041b9b0b531879355a60c37a763b676dabed63b09693c41f2a060accdc7ae8ee
```

- Open Grafana UI at http://localhost:3000





Note:- you need to login as admin in uername and password section you need to hit admin as well then you can change you password to your choice and proceeed further.

- Step 3: Create Dashboards: -
- Create custom dashboards in Grafana for:
- **MongoDB health** (connections, memory usage, query time)
- **Frontend performance** (use Prometheus metrics if applicable)

4. **Log Aggregation with Loki: -**

- Step 1: Install Loki and Promtail

Use Loki for log aggregation.

```
HP@LAPTOP-JRN3DQ8O MINGW64 /c/Gke-mern-deployment/Mern-promethius and graffana/TravelMemory/backend (main)
$ docker run -d --name=loki -p 3100:3100 grafana/loki:2.7.1
Unable to find image 'grafana/loki:2.7.1' locally
2.7.1: Pulling from grafana/loki
4624c9a20c87: Download complete
476d1aacbcf9: Download complete
5889a79d2ca2: Download complete
213ec9aee27d: Download complete
2a1b49b65c00: Download complete
b10017a0d09d: Download complete
0b0825e005dd: Download complete
Digest: sha256:262d91088fb8cc31d3a6a0591aea52f74fa448884c961e63abd35decd3570a88
Status: Downloaded newer image for grafana/loki:2.7.1
f951bd10c4bc9372d72a5965a04f0caf8709f165c9ef3721992823a538ea41bf
```

```
HP@LAPTOP-JRN3DQ8O MINGW64 /c/Gke-mern-deployment/Mern-promethius and graffana/TravelMemory/backend (main)
$ docker run -d --name=promtail -p 9080:9080 grafana/promtail:2.7.1
Unable to find image 'grafana/promtail:2.7.1' locally
2.7.1: Pulling from grafana/promtail
d2d785a138bb: Download complete
025c56f98b67: Download complete
7018e3d57cbe: Download complete
0704b6e8cfb2: Download complete
0ccd6e7d847b: Download complete
6e6914193de4: Download complete
Digest: sha256:aa7333bb912b1017d35f16bd5d7f6d191963d6cee78bc0bd0df0a56990a3e42
Status: Downloaded newer image for grafana/promtail:2.7.1
e5dfb05be3467d73f8f2abad7ff001402bd0b3ad27f0cbd13cf7fe31b6d6f049
```

Step 2: Configure Prom tail to collect logs:-
Create a `promtail-config.yml` file to scrape logs from your Node.js application.

- Step 3: Create Grafana Dashboard for logs
- Add Loki as a data source in Grafana.
- Create dashboards to visualize logs in Grafana.

5. **Implement Distributed Tracing with Jaeger:-**

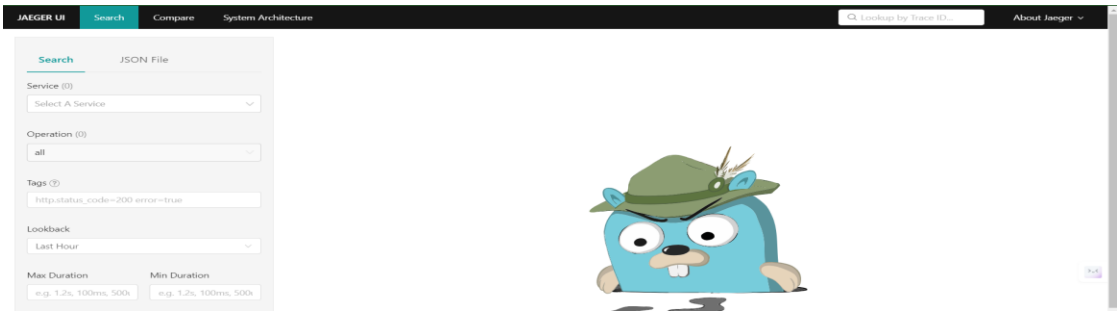
- Step 1: Install Jaeger
- Run Jaeger as a Docker container.
  docker run -d --name=jaeger -e COLLECTOR_ZIPKIN_HTTP_PORT=9411 -p 5775:5775/udp \
  -p 6831:6831/udp -p 6832:6832/udp -p 5778:5778 \
  -p 16686:16686 -p 14268:14268 -p 14250:14250 \
  -p 9411:9411 jaegertracing/all-in-one:1.32

```
HP@LAPTOP-JRN3DQ8O MINGW64 /c/Gke-mern-deployment/Mern-promethius and graffana/TravelMemory/backend (main)
$ docker run -d --name=jaeger -e COLLECTOR_ZIPKIN_HTTP_PORT=9411 -p 5775:5775/udp \
  -p 6831:6831/udp -p 6832:6832/udp -p 5778:5778 \
  -p 16686:16686 -p 14268:14268 -p 14250:14250 \
  -p 9411:9411 jaegertracing/all-in-one:1.32
Unable to find image 'jaegertracing/all-in-one:1.32' locally
1.32: Pulling from jaegertracing/all-in-one
97518928ae5f: Download complete
ee9b4236e8dd: Download complete
2ef9eb07f6be: Download complete
d4d8c2032836: Download complete
Digest: sha256:1540c94ec378a146e4df09832f685d95e8ca4625799f99af70292d74940a5971
Status: Downloaded newer image for jaegertracing/all-in-one:1.32
a7463f17ca617fc9ea3a3606bceeb436449d546c502d4298e8900766b22d7b75
```

- Step 2: Integrate Jaeger with Node.js
  Use OpenTelemetry to send traces to Jaeger.

```
HP@LAPTOP-JRN3DQ8O MINGW64 /c/Gke-mern-deployment/Mern-promethius and graffana/TravelMemory/backend (main)
$ npm install @opentelemetry/api @opentelemetry/sdk-node @opentelemetry/exporter-jaeger

added 88 packages, changed 1 package, and audited 210 packages in 20s

21 packages are looking for funding
  run `npm fund` for details

13 vulnerabilities (1 low, 3 moderate, 8 high, 1 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.

HP@LAPTOP-JRN3DQ8O MINGW64 /c/Gke-mern-deployment/Mern-promethius and graffana/TravelMemory/backend (main)
$
```

  Modify your backend to send traces to Jaeger:
  This will start Jaeger, and you can access the Jaeger UI at http://localhost:16686.



Step 1: Install OpenTelemetry for Node.js

In your backend project directory, install the necessary OpenTelemetry libraries for tracing:

```
HP@LAPTOP-JRN3DQ8O MINGW64 /c/Gke-mern-deployment/Mern-promethius and graffana/TravelMemory/backend (main)
$ npm install @opentelemetry/api @opentelemetry/sdk-node @opentelemetry/auto-instrumentations-node @opentelemetry/exporter-jaeger

added 83 packages, and audited 293 packages in 23s

23 packages are looking for funding
  run `npm fund` for details

13 vulnerabilities (1 low, 3 moderate, 8 high, 1 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
```

Step 2: Modify `index.js` to Enable Tracing: -

Create a new file (optional for better organization): Create a file named `tracing.js` to configure OpenTelemetry. This will keep your code cleaner.

```
 GNU nano 7.2                                                    tracing.js
// tracing.js
const opentelemetry = require('@opentelemetry/sdk-node');
const { getNodeAutoInstrumentations } = require('@opentelemetry/auto-instrumentations-node');
const { JaegerExporter } = require('@opentelemetry/exporter-jaeger');

const sdk = new opentelemetry.NodeSDK({
    traceExporter: new JaegerExporter({
        endpoint: 'http://localhost:14268/api/traces', // Jaeger collector endpoint
    }),
    instrumentations: [getNodeAutoInstrumentations()], // Automatically instruments popular libraries like Express
});

sdk.start()
    .then(() => {
        console.log('Tracing initialized');
    })
    .catch((error) => {
        console.log('Error initializing tracing', error);
    });

process.on('SIGTERM', () => {
    sdk.shutdown().then(() => {
        console.log('Tracing terminated');
                                                        [ Read 25 lines ]
```

2. **Modify `index.js`**: Import `tracing.js` at the top of your `index.js` file so that the tracing is initialized when the backend starts.

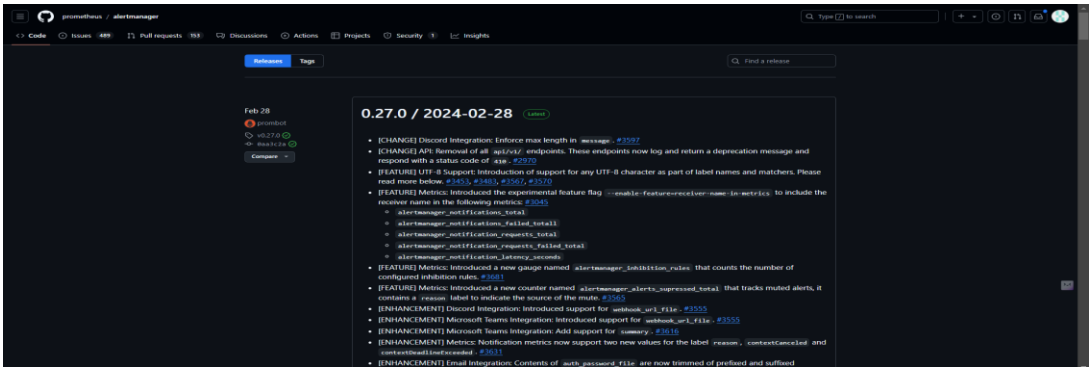
```
backend > JS index.js > ...
      You, 1 second ago | 2 authors (You and one other)
1     require('./tracing'); // Import tracing module for Jaeger integration
2
3     const express = require('express');       You, now • Uncommitted changes
4     const cors = require('cors');
5     const promClient = require('prom-client'); // Prometheus client library
6     require('dotenv').config();
7
8     const  const PORT: string | undefined
9     const PORT = process.env.PORT;
10    const conn = require('./conn');
11    app.use(express.json());
12    app.use(cors());
13
14    const tripRoutes = require('./routes/trip.routes');
15
16    // Prometheus metrics collection
17    const register = new promClient.Registry(); // Create a new registry for custom metrics
18
19    // HTTP request duration metric (histogram)
20    const httpRequestDurationMicroseconds = new promClient.Histogram({
21        name: 'http_request_duration_ms',
22        help: 'Duration of HTTP requests in ms',
23        labelNames: ['method', 'route', 'code'], // Labels for identifying metrics
24        buckets: [50, 100, 200, 500, 1000], // Define buckets for response times
25    });
26
27    // Register the metrics with Prometheus registry
28    register.registerMetric(httpRequestDurationMicroseconds);
29
```

6. **Alerting and Anomaly Detection:-**

- Step 1: Create Alerting Rules
  Set up alerting rules in Prometheus for application-specific metrics, such as response time exceeding a threshold.

Create the `alert.rules.yml` file:-

- Step 2: Configure Alertmanager
- **Install Alertmanager**:
- **Visit the official Alertmanager releases page** on GitHub:



- **Download the appropriate version** for your system:
  alertmanager-0.27.0.illumos-amd64.tar.gz in my case.
  Example for Linux:-
  wget

https://github.com/prometheus/alertmanager/releases/download/v0.26.0/alertmanager-0.26.0.linux-amd64.tar.gz

```
ubuntu@LAPTOP-JRN3DQ8O:/mnt/c/Gke-mern-deployment/Mern-promethius and graffana/TravelMemory$ wget https://github.com/prometheus/alertmanager/releases/download/v0.26.0/alertmanager-0.26.0.linux-amd64.tar.gz
--2024-10-15 13:41:56--  https://github.com/prometheus/alertmanager/releases/download/v0.26.0/alertmanager-0.26.0.linux-amd64.tar.gz
Resolving github.com (github.com)... 20.207.73.82
Connecting to github.com (github.com)|20.207.73.82|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/11452538/84ce640f-af00-45ce-9500-60309e9f598d?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=releaseassetproduction%2F20241015%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20241015T081157Z&X-Amz-Expires=300&X-Amz-Signature=3929a0dac6061191986358a8f8fe89353d10fbb760a16a07cee45ede30aa014d&X-Amz-SignedHeaders=host&response-content-disposition=attachment%3B%20filename%3Dalertmanager-0.26.0.linux-amd64.tar.gz&response-content-type=application%2Foctet-stream [following]
--2024-10-15 13:41:57--  https://objects.githubusercontent.com/github-production-release-asset-2e65be/11452538/84ce640f-af00-45ce-9500-60309e9f598d?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=releaseassetproduction%2F20241015%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20241015T081157Z&X-Amz-Expires=300&X-Amz-Signature=3929a0dac6061191986358a8f8fe89353d10fbb760a16a07cee45ede30aa014d&X-Amz-SignedHeaders=host&response-content-disposition=attachment%3B%20filename%3Dalertmanager-0.26.0.linux-amd64.tar.gz&response-content-type=application%2Foctet-stream
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 29717412 (28M) [application/octet-stream]
Saving to: 'alertmanager-0.26.0.linux-amd64.tar.gz'

alertmanager-0.26.0.linux-amd64.tar.gz   24%[=================>                                                    ]   6.81M   146KB/s    eta 2m 53s
```

Step 2: Extract the Files:-

Extract the downloaded file to a directory:

- For Linux/macOS:-

```
ubuntu@LAPTOP-JRN3DQ8O:/mnt/c/Gke-mern-deployment/Mern-promethius and graffana/TravelMemory$ tar -xvzf alertmanager-0.26.0.linux-amd64.tar.gz
alertmanager-0.26.0.linux-amd64/
alertmanager-0.26.0.linux-amd64/alertmanager.yml
alertmanager-0.26.0.linux-amd64/NOTICE
alertmanager-0.26.0.linux-amd64/amtool
alertmanager-0.26.0.linux-amd64/alertmanager
alertmanager-0.26.0.linux-amd64/LICENSE
ubuntu@LAPTOP-JRN3DQ8O:/mnt/c/Gke-mern-deployment/Mern-promethius and graffana/TravelMemory$
```

Navigate to the extracted folder: -

```
ubuntu@LAPTOP-JRN3DQ8O:/mnt/c/Gke-mern-deployment/Mern-promethius and graffana/TravelMemory$ cd alertmanager-0.26.0.linux-amd64
ubuntu@LAPTOP-JRN3DQ8O:/mnt/c/Gke-mern-deployment/Mern-promethius and graffana/TravelMemory/alertmanager-0.26.0.linux-amd64$
```

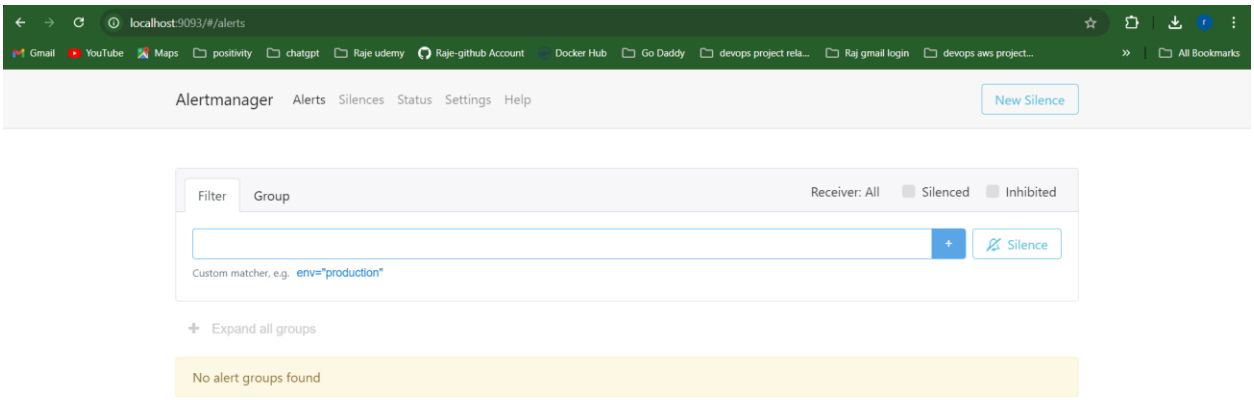
Step 3: Configure Alertmanager:-

Create a configuration file (`alertmanager.yml`) in the Alertmanager directory. This file will define how Alertmanager handles and routes alerts.

Customize the `route` section based on how you want the alerts to be grouped and routed. For more advanced configurations, check out the Alertmanager configuration docs.

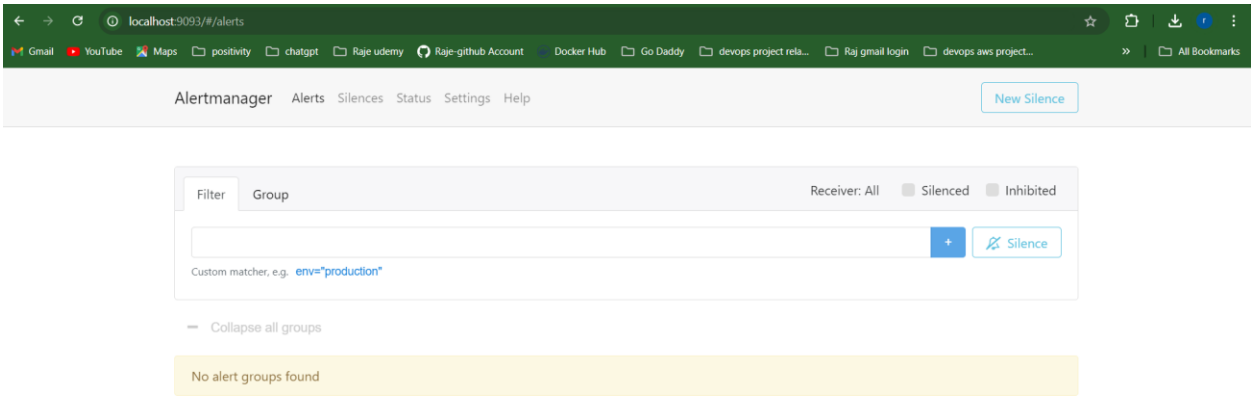
Step 4: Run Alertmanager:-

```
ubuntu@LAPTOP-JRN3DQ8O:/mnt/c/Gke-mern-deployment/Mern-prometheus and graffana/TravelMemory/backend/alertmanager-0.26.0.linux-amd64$ ./alertmanager --config.
file=alertmanager.yml
ts=2024-10-15T08:30:04.424Z caller=main.go:245 level=info msg="Starting Alertmanager" version="(version=0.26.0, branch=HEAD, revision=d7b4f0c7322e7151d6e3b1e
31cbc15361e295d8d)"
ts=2024-10-15T08:30:04.424Z caller=main.go:246 level=info build_context="(go=go1.20.7, platform=linux/amd64, user=root@df8d7debeef4, date=20230824-11:11:58,
tags=netgo)"
ts=2024-10-15T08:30:04.644Z caller=cluster.go:186 level=info component=cluster msg="setting advertise address explicitly" addr=172.21.64.139 port=9094
ts=2024-10-15T08:30:04.654Z caller=cluster.go:683 level=info component=cluster msg="Waiting for gossip to settle..." interval=2s
ts=2024-10-15T08:30:04.762Z caller=coordinator.go:113 level=info component=configuration msg="Loading configuration file" file=alertmanager.yml
ts=2024-10-15T08:30:04.764Z caller=coordinator.go:126 level=info component=configuration msg="Completed loading of configuration file" file=alertmanager.yml
ts=2024-10-15T08:30:04.884Z caller=tls_config.go:274 level=info msg="Listening on" address=[::]:9093
ts=2024-10-15T08:30:04.890Z caller=tls_config.go:277 level=info msg="TLS is disabled." http2=false address=[::]:9093
ts=2024-10-15T08:30:06.655Z caller=cluster.go:708 level=info component=cluster msg="gossip not settled" polls=0 before=0 now=1 elapsed=2.00038471s
ts=2024-10-15T08:30:14.658Z caller=cluster.go:700 level=info component=cluster msg="gossip settled; proceeding" elapsed=10.003296515s
```



Step 5: Integrate Alertmanager with Prometheus:-

`prometheus.yml`:-



Github repo:- https://github.com/Rajpreetsingh12/TravelMemory.git