Great! Here's how to run everything together — **.NET App**, **Prometheus**, and **Grafana** — using **Docker Compose**.

**✅ Folder Structure**

my-monitoring-app/

├── prometheus.yml

├── docker-compose.yml

├── YourApp/ # Your .NET project

│ ├── YourApp.csproj

│ └── Program.cs

└── YourApp/Dockerfile

**📁 1. YourApp/Dockerfile**

Make sure your Dockerfile looks like this:

FROM mcr.microsoft.com/dotnet/aspnet:6.0 AS base

WORKDIR /app

EXPOSE 5000

FROM mcr.microsoft.com/dotnet/sdk:6.0 AS build

WORKDIR /src

COPY . .

RUN dotnet publish -c Release -o /app/publish

FROM base AS final

WORKDIR /app

COPY --from=build /app/publish .

ENTRYPOINT ["dotnet", "YourApp.dll"]

Replace YourApp.dll with your actual DLL name.

**🧾 2. prometheus.yml**

global:

scrape\_interval: 5s

scrape\_configs:

- job\_name: 'dotnet-app'

static\_configs:

- targets: ['dotnet-app:5000']

Prometheus will scrape the metrics from the dotnet-app service defined in Docker Compose.

**🧪 3. docker-compose.yml**

services:

dotnet-app:

build:

context: ./YourApp

ports:

- "5000:5000"

networks:

- monitoring

prometheus:

image: prom/prometheus

volumes:

- ./prometheus.yml:/etc/prometheus/prometheus.yml

ports:

- "9090:9090"

networks:

- monitoring

grafana:

image: grafana/grafana

ports:

- "3000:3000"

networks:

- monitoring

environment:

- GF\_SECURITY\_ADMIN\_USER=admin

- GF\_SECURITY\_ADMIN\_PASSWORD=admin

networks:

monitoring:

**⚙️ 4. Instrument Your .NET App (Program.cs)**

Make sure you’ve set up metrics like this (for minimal APIs):

using Prometheus;

var builder = WebApplication.CreateBuilder(args);

var app = builder.Build();

app.UseRouting();

app.UseHttpMetrics(); // Collect HTTP request metrics

app.UseEndpoints(endpoints =>

{

endpoints.MapMetrics(); // Exposes /metrics

});

app.MapGet("/", () => "Hello from .NET!");

app.Run();

**🚀 5. Run Everything**

In the root folder (my-monitoring-app), run:

docker-compose up --build

* .NET App: <http://localhost:5000>
* Prometheus: <http://localhost:9090>
* Grafana: <http://localhost:3000>

Grafana login: admin / admin

**📊 6. Configure Grafana (First Time)**

1. Go to Grafana UI: <http://localhost:3000>
2. Add data source:
   * Type: **Prometheus**
   * URL: http://prometheus:9090
3. Import a dashboard:
   * You can import one from https://grafana.com/grafana/dashboards/ (e.g., **Prometheus .NET Core metrics**)

**Query your metric**

Using Prometheus, in the *Query* window in **"Expression"** search box, type your metric name: e.g. in this case, we created a gauge metric in .Net Application with name “***my\_custom\_metric***”.

**1. Show Query Data in Prometheus UI**

Prometheus has a built-in UI for querying and viewing metrics.

**Steps:**

1. Open Prometheus UI in your browser:  
   http://localhost:9090
2. Click on the **“Graph”** tab at the top.
3. In the **“Expression”** input box, type your PromQL query. Example:

***up***

Or ***my\_custom\_metric***

1. Click **“Execute”**.
2. You’ll see the query results as a **table** below, and if you switch to the **Graph** tab (right next to the Expression input), you can see the data plotted over time.

**2. Show Query Data in Grafana**

Grafana visualizes data from Prometheus or other data sources using dashboards and panels.

**Steps:**

1. Open Grafana in your browser:  
   http://localhost:3000
2. Go to **Explore** (compass icon on the left sidebar).
3. Select your **Prometheus** data source from the dropdown at the top.
4. In the query input box, write your PromQL query. Example:

***up***

Or ***my\_custom\_metric***

1. Press **Enter** or click **Run query**.
2. Grafana will show the query results as a graph by default. You can switch to **Table** or other visualization options using the panel settings.