**Exercise 1: Prometheus and Grafana Installation**

Here's a complete guide to **install Prometheus and Grafana** using:

* ✅ **Installation packages** (native install on your OS)
* ✅ **Docker Desktop** (easier, isolated, and portable)

## 🔧 OPTION 1: Native Installation (Installation Package)

### 🚀 A. Install ****Prometheus****

#### ✅ Step 1: Download

* Go to: <https://prometheus.io/download/>
* Download the **latest Prometheus** package for your OS.
* Extract it to a folder (e.g., C:\Prometheus or /opt/prometheus).

#### ✅ Step 2: Configure prometheus.yml

Example:

global:

scrape\_interval: 15s

scrape\_configs:

- job\_name: 'spring-boot'

metrics\_path: '/actuator/prometheus'

static\_configs:

- targets: ['localhost:8080']

Place it in the extracted folder.

#### ✅ Step 3: Run Prometheus

./prometheus --config.file=prometheus.yml

Prometheus UI: <http://localhost:9090>

### 🚀 B. Install ****Grafana****

#### ✅ Step 1: Download

* Go to: <https://grafana.com/grafana/download>
* Download the installer for your OS (Windows .msi, macOS .dmg, or Linux package).

#### ✅ Step 2: Install

Follow installer instructions. On Linux:

sudo apt install -y ./grafana-\*.deb

sudo systemctl start grafana-server

sudo systemctl enable grafana-server

#### ✅ Step 3: Access Grafana

Go to:

http://localhost:3000

* Default login: **admin / admin**
* Add **Prometheus** as a data source: http://localhost:9090

## 🐳 OPTION 2: Docker Desktop (Recommended for Simplicity)

### ✅ Step 1: Create prometheus.yml

global:

scrape\_interval: 15s

scrape\_configs:

- job\_name: 'spring-boot'

metrics\_path: '/actuator/prometheus'

static\_configs:

- targets: ['host.docker.internal:8080'] # Use IP if on Linux

Save it as prometheus.yml.

### ✅ Step 2: Run Prometheus in Docker

docker run -d \

--name prometheus \

-p 9090:9090 \

-v "$PWD/prometheus.yml":/etc/prometheus/prometheus.yml \

prom/prometheus

On Windows: Use "D:/path/to/prometheus.yml" with forward slashes.

### ✅ Step 3: Run Grafana in Docker

docker run -d \

--name grafana \

-p 3000:3000 \

grafana/grafana

### ✅ Step 4: Configure Grafana

1. Go to: <http://localhost:3000>
2. Login: **admin / admin**
3. Add **Prometheus** as a data source:
   * URL: http://host.docker.internal:9090 (or IP if on Linux)

## 🔍 Access Summary

| **Service** | **Port** | **URL** |
| --- | --- | --- |
| Prometheus | 9090 | <http://localhost:9090> |
| Grafana | 3000 | <http://localhost:3000> |

**Exercise 2: Spring Boot Application with Prometheus**

Create a **Spring Boot application** with a @RestController that displays **"Hello World"**, and integrates **Prometheus** metrics.

**✅ 1. Create Spring Boot Project**

Use <https://start.spring.io> or your IDE.

**✅ Choose:**

* **Project:** Maven or Gradle
* **Language:** Java
* **Dependencies:**
  + Spring Web
  + Spring Boot Actuator
  + Micrometer Prometheus

Download and unzip the project.

**✅ 2. Create HelloController**

Create a controller that returns "Hello World":

package com.example.demo;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RestController;

@RestController

public class HelloController {

@GetMapping("/hello")

public String hello() {

return "Hello World";

}

}

**✅ 3. Configure Prometheus Metrics**

**📄 application.properties (or use YAML)**

# Expose Prometheus endpoint

management.endpoints.web.exposure.include=health,info,prometheus

management.endpoint.prometheus.enabled=true

# Optional: Change server port if needed

# server.port=8080

You can also use application.yml if preferred.

**✅ 4. Run Your Application**

Use your IDE or terminal:

./mvnw spring-boot:run

Or if Gradle:

./gradlew bootRun

Check these URLs:

* http://localhost:8080/hello → should return **Hello World**
* http://localhost:8080/actuator/prometheus → should show raw Prometheus metrics

**✅ 5. Set Up Prometheus (in Docker)**

**📄 Create prometheus.yml**

global:

scrape\_interval: 15s

scrape\_configs:

- job\_name: 'spring-boot'

metrics\_path: '/actuator/prometheus'

static\_configs:

- targets: ['host.docker.internal:8080'] # use your actual host IP if on Linux

Put this file in a folder, e.g., D:/Demo/

**🐳 Run Prometheus Container**

docker run --name prometheus -d -p 9090:9090 -v "D:/Demo/prometheus.yml":/etc/prometheus/prometheus.yml prom/prometheus

* Prometheus UI: <http://localhost:9090>
* Status → Targets → should show Spring Boot target as **UP**

**✅ 6. View Metrics in Prometheus**

In Prometheus UI:

1. Type a metric like:
2. http\_server\_requests\_seconds\_count
3. Click **Execute**
4. Switch to **Graph** tab to visualize it

**✅ Bonus: Add a Custom Metric (Optional)**

import io.micrometer.core.instrument.Counter;

import io.micrometer.core.instrument.MeterRegistry;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RestController;

@RestController

public class HelloController {

private final Counter helloCounter;

public HelloController(MeterRegistry registry) {

this.helloCounter = registry.counter("custom\_hello\_requests\_total");

}

@GetMapping("/hello")

public String hello() {

helloCounter.increment();

return "Hello World";

}

}

Now check custom\_hello\_requests\_total in Prometheus after calling /hello.