**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**



**МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**Факультет Информационных технологий**

***Кафедра Информатики и информационных технологий***

**направление подготовки**

**09.03.02 «Информационные системы и технологии»**

**ЛАБОРАТОРНАЯ РАБОТА № \_18\_**

**Дисциплина:** \_Backend разработка

**Тема:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Выполнил(а): студент(ка) группы \_\_231-336\_\_**

\_\_\_\_\_\_\_\_Канищев И.М\_\_\_\_\_\_\_\_\_\_\_\_

(Фамилия И.О.)

**Дата, подпись** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Проверил: \_\_*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_***

(Фамилия И.О., степень, звание) **(Оценка)**

**Дата, подпись** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  ***\_\_\_\_\_\_\_\_\_\_\_***

(Дата) (Подпись)

**Замечания:**

**Москва**

**2025**

## **1. Настройка проекта**

### ****1.1 Установка пакета****

dotnet add package prometheus-net.AspNetCore

**1.2 Program.cs**

using Microsoft.AspNetCore.Builder;

using Microsoft.Extensions.DependencyInjection;

using Microsoft.Extensions.Hosting;

using Prometheus;

var builder = WebApplication.CreateBuilder(args);

// Добавляем MVC

builder.Services.AddControllersWithViews();

var app = builder.Build();

// Middleware Prometheus для сбора метрик HTTP

app.UseHttpMetrics();

// Статические файлы и маршрутизация

app.UseStaticFiles();

app.UseRouting();

// Endpoint для Prometheus метрик

app.UseEndpoints(endpoints =>

{

endpoints.MapMetrics("/metrics"); // Prometheus собирает метрики здесь

endpoints.MapControllerRoute(

name: "default",

pattern: "{controller=Home}/{action=Index}/{id?}");

});

app.Run();

## **3. Представления**

### ****Views/Home/Index.cshtml****

@{

ViewData["Title"] = "Главная";

}

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

<title>@ViewData["Title"]</title>

</head>

<body>

<h1>Мониторинг через Prometheus</h1>

<p>Эта страница демонстрирует сбор метрик HTTP-запросов и пользовательских счетчиков.</p>

<a href="/Home/Error">Сгенерировать ошибку</a>

</body>

</html>

**Views/Home/Error.cshtml**

@{

ViewData["Title"] = "Ошибка";

}

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

<title>@ViewData["Title"]</title>

<style>

body {

font-family: Arial, sans-serif;

background-color: #f8d7da;

color: #721c24;

text-align: center;

padding: 50px;

}

h1 { font-size: 48px; }

p { font-size: 20px; }

a { color: #0056b3; text-decoration: none; font-weight: bold; }

a:hover { text-decoration: underline; }

</style>

</head>

<body>

<h1>Что-то пошло не так</h1>

<p>Произошла ошибка при обработке вашего запроса.</p>

<p>Вернитесь на <a href="/">главную страницу</a>.</p>

</body>

</html>

## **4. Установка Grafana и подключение к Prometheus**

**1. Установка Prometheus**

1. Скачиваем Prometheus с официального сайта: https://prometheus.io/download/
2. Распакуем архив в удобную директорию.
3. В корне создаем файл конфигурации prometheus.yml:

global:

scrape\_interval: 5s

scrape\_configs:

- job\_name: 'aspnetcore-app'

static\_configs:

- targets: ['localhost:7247'] # Порт вашего ASP.NET Core приложения

1. Запускаем Prometheus:

Prometheus будет доступен по адресу: http://localhost:9090

Endpoint /metrics приложения должен быть доступен (например, http://localhost:7247/metrics).

## **2. Установка Grafana**

1. Скачиваем Grafana с официального сайта: https://grafana.com/grafana/download
2. Установим и запустите Grafana.
3. По умолчанию Grafana доступна по адресу: http://localhost:3000

 Логин: admin

 Пароль: admin (потом можно изменить)

## **3. Подключение Prometheus к Grafana**

1. В Grafana перейдём в **Configuration → Data Sources → Add data source**
2. Выбираем **Prometheus**
3. В поле URL указываем адрес Prometheus, например:

http://localhost:9090

1. Нажмите **Save & Test** — Grafana должна подтвердить подключение.

## **4. Создание дашборда для метрик ASP.NET Core**

1. В Grafana выберите **Create → Dashboard → Add new panel**
2. В поле **Query** выберите источник Prometheus.
3. Введите нужную метрику, например:

home\_index\_requests\_total

1. Настройте визуализацию: график, счетчик, таблицу.
2. Сохраните дашборд.

Теперь мы видим в реальном времени:

* количество запросов к Home/Index
* стандартные HTTP метрики (http\_request\_duration\_seconds\_sum, http\_requests\_received\_total)
* ошибки и статус-коды

## **5. Полезные Prometheus метрики для ASP.NET Core**

* **home\_index\_requests\_total** — пользовательский счетчик запросов
* **http\_requests\_total** — количество HTTP запросов
* **http\_request\_duration\_seconds** — время обработки запросов
* **http\_response\_size\_bytes** — размер ответа

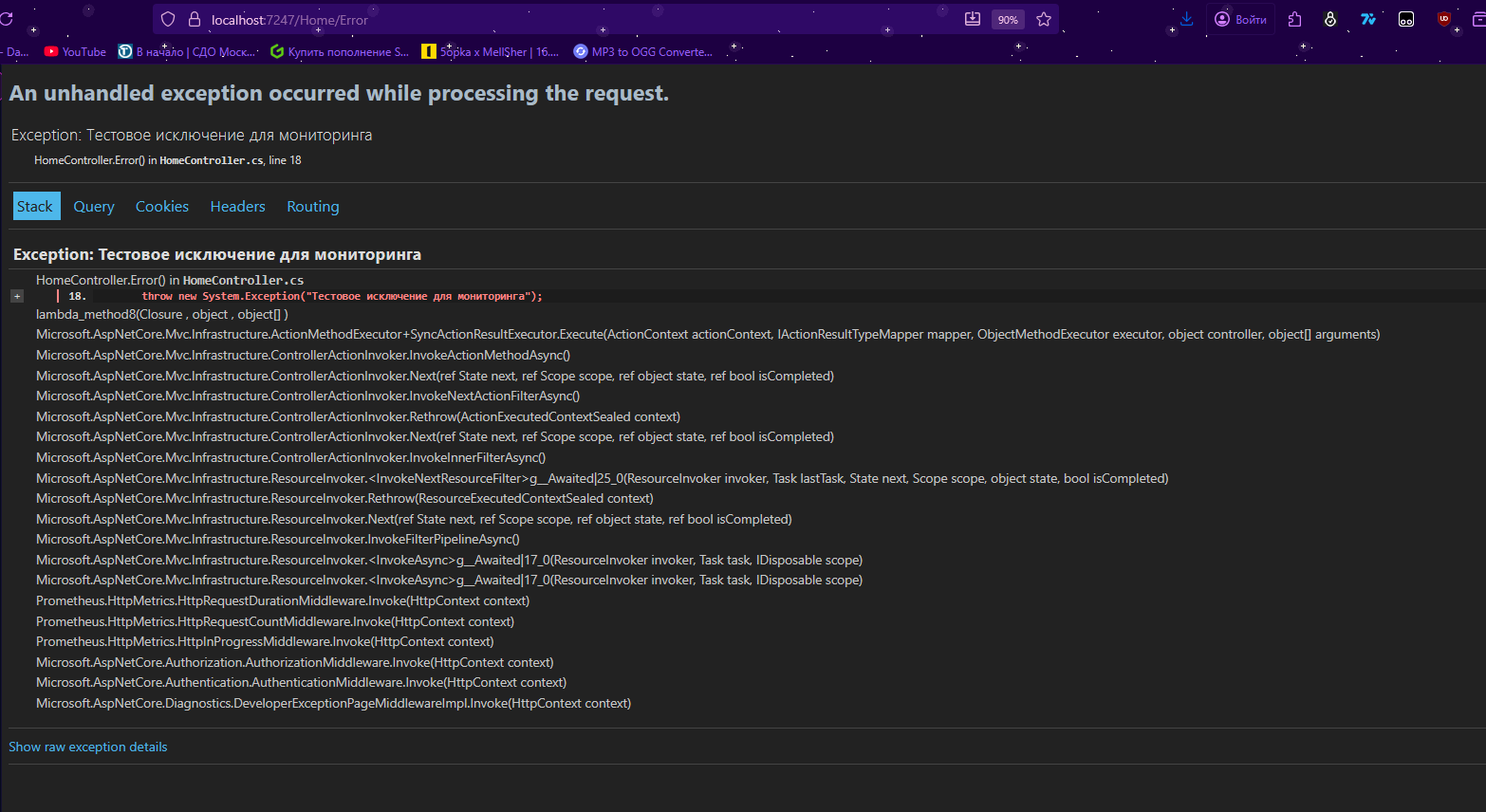
Вы можете добавлять пользовательские метрики через Prometheus.Metrics.CreateCounter или Gauge.

## **5. Проверка работы мониторинга**

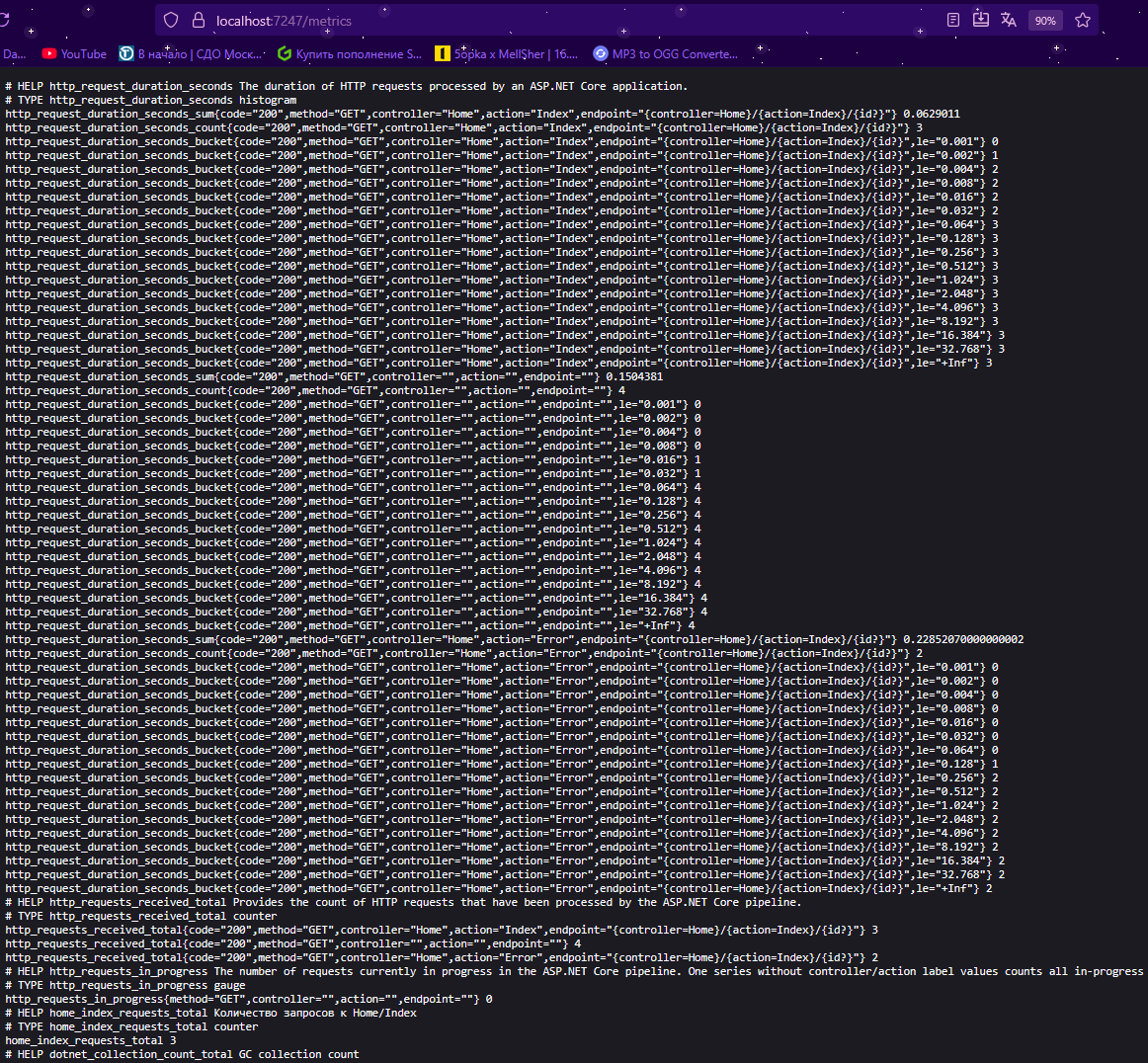
1. Запускаем приложение.
2. Открываем страницу /Home/Index → счетчик home\_index\_requests\_total увеличивается.



1. Переходим на /Home/Error → срабатывает исключение.



1. Переходим на /metrics → Prometheus собирает метрики:
   * home\_index\_requests\_total
   * стандартные HTTP метрики (время отклика, количество запросов, статус-коды).



1. Подключаем Grafana к Prometheus для визуализации метрик:

