LoadBalancer.DistributedAlgorithms2 Documentation

Welcome to the LoadBalancer.DistributedAlgorithms2 project! This is a fun and educational distributed system built with .NET 9, designed to explore load balancing, fault tolerance, and monitoring. Created on May 23, 2025, at 01:56 PM EEST, this project balances computational tasks across multiple worker nodes, caches results with Redis, and tracks performance with Prometheus and Grafana—all running in Docker containers. Whether you're a student, developer, or just curious, dive in and play around!

# Overview

This project simulates a distributed system where a load balancer distributes calculation requests to worker nodes. It’s powered by:

- A single .NET 9 codebase that runs as either a load balancer or worker (controlled by an environment variable).  
- Redis for caching results to boost performance.  
- Prometheus for collecting metrics and Grafana for visualizing them.  
  
The system is fault-tolerant—if a worker fails, the load balancer reroutes tasks—and it’s all containerized with Docker for easy setup.

# Features

- Load Balancing: Distributes tasks to three worker instances.  
- Fault Tolerance: Uses Polly for retry and circuit breaker logic to handle failures.  
- Caching: Stores results in Redis to avoid redundant calculations.  
- Monitoring: Tracks metrics with Prometheus and visualizes them in Grafana.  
- Dockerized: Runs everything in containers for portability.

# Getting Started

## Prerequisites

- Docker Desktop: Installed on Windows (with WSL2 backend enabled).  
- .NET 9 SDK (optional): Useful for local development, though Docker handles builds.  
- PowerShell or Command Prompt: For running commands.

## Installation

1. Clone the Repository:

git clone https://github.com/mohammedabdelaleem/LoadBalancer.DistributedAlgorithms2.git  
cd LoadBalancer.DistributedAlgorithms2

2. Build and Run:

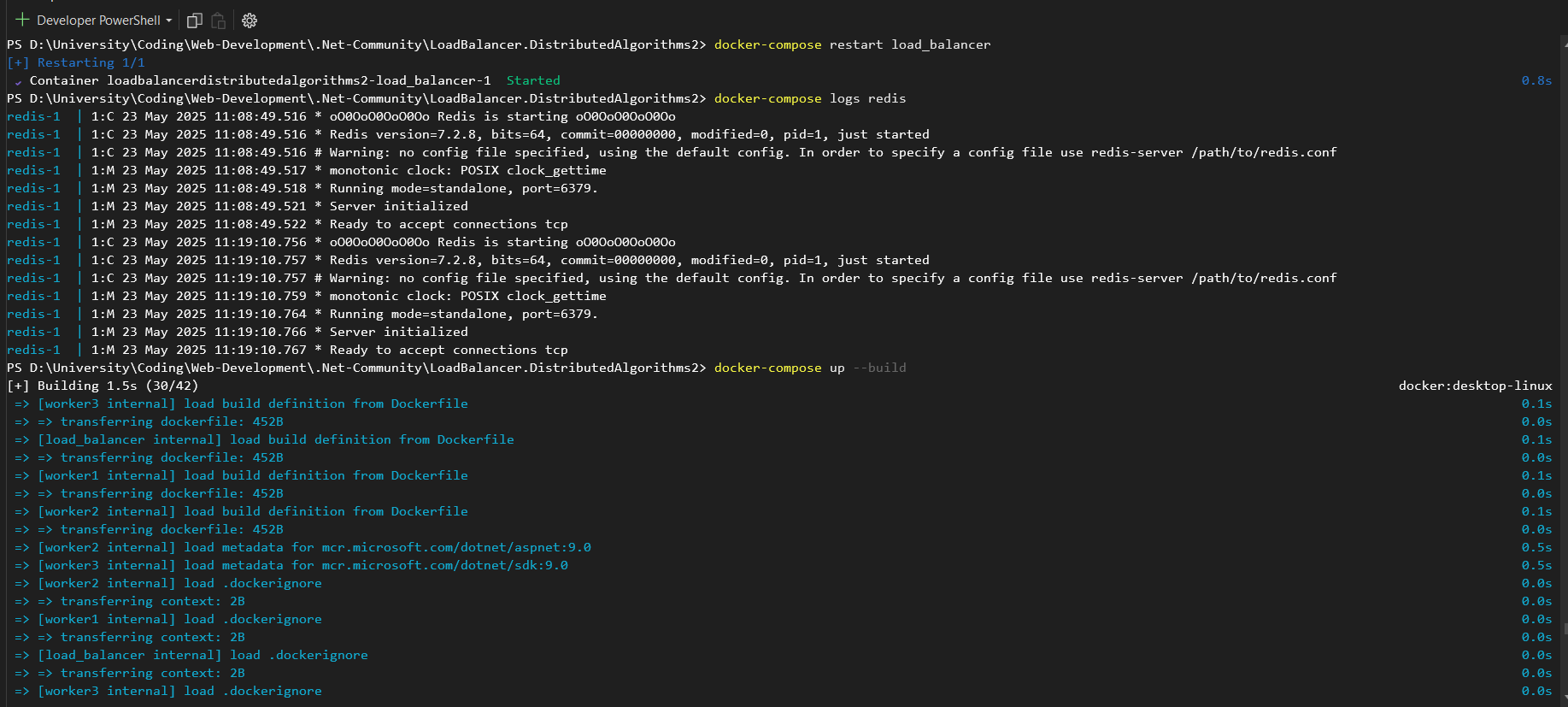
docker-compose down --rmi all  
docker system prune -a -f  
docker-compose up --build

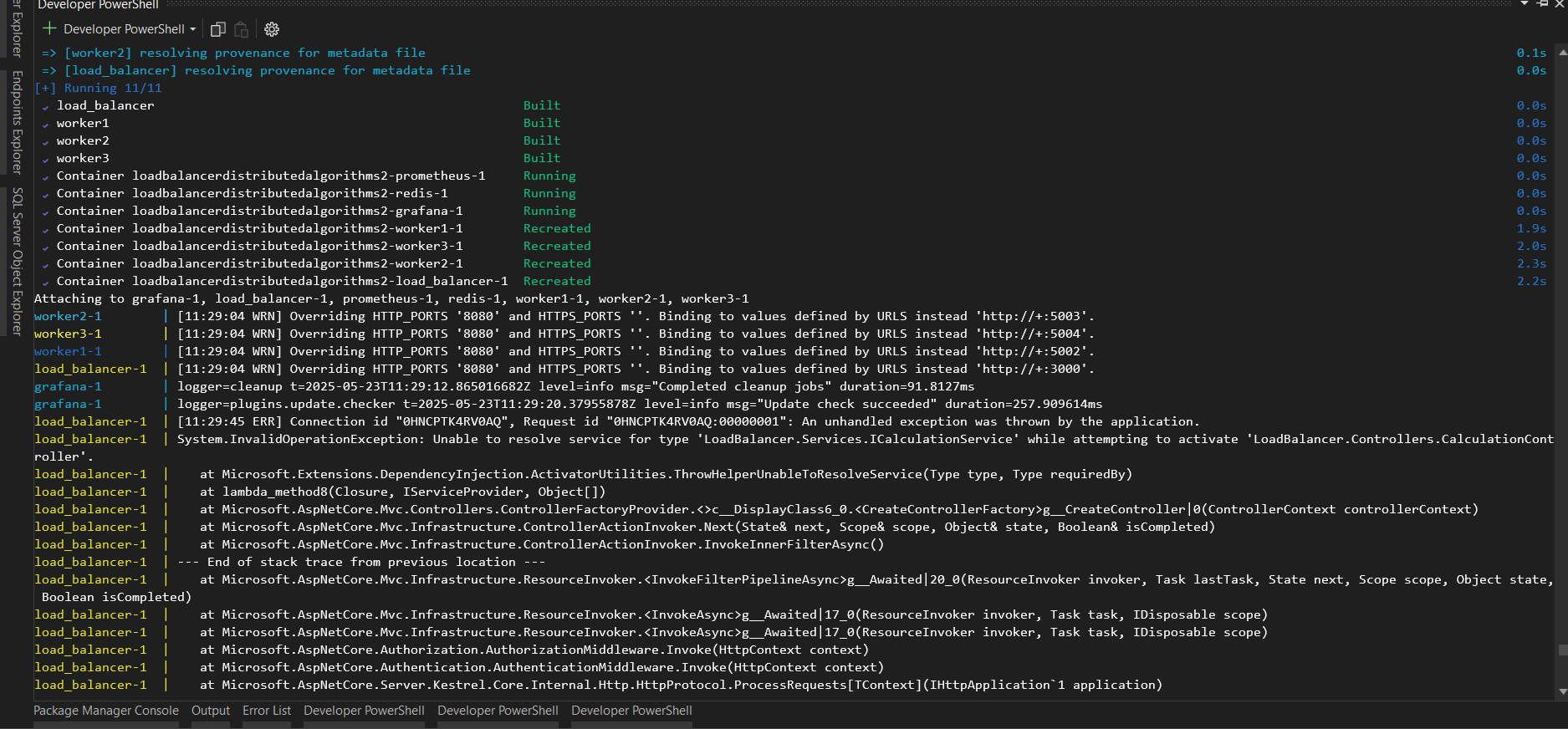
This pulls images (Redis 7.2, Prometheus, Grafana) and builds the app. First run might take a few minutes.

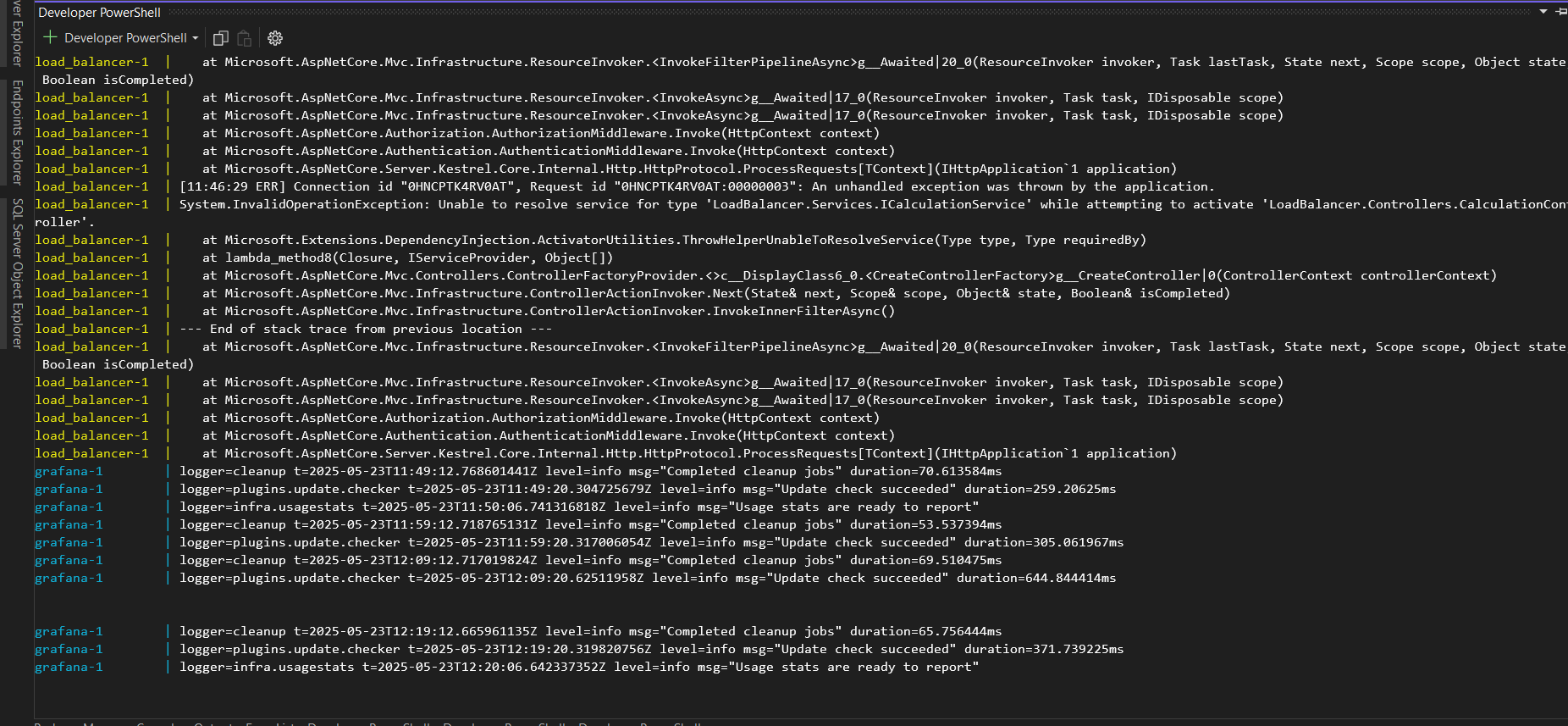
3. Verify:

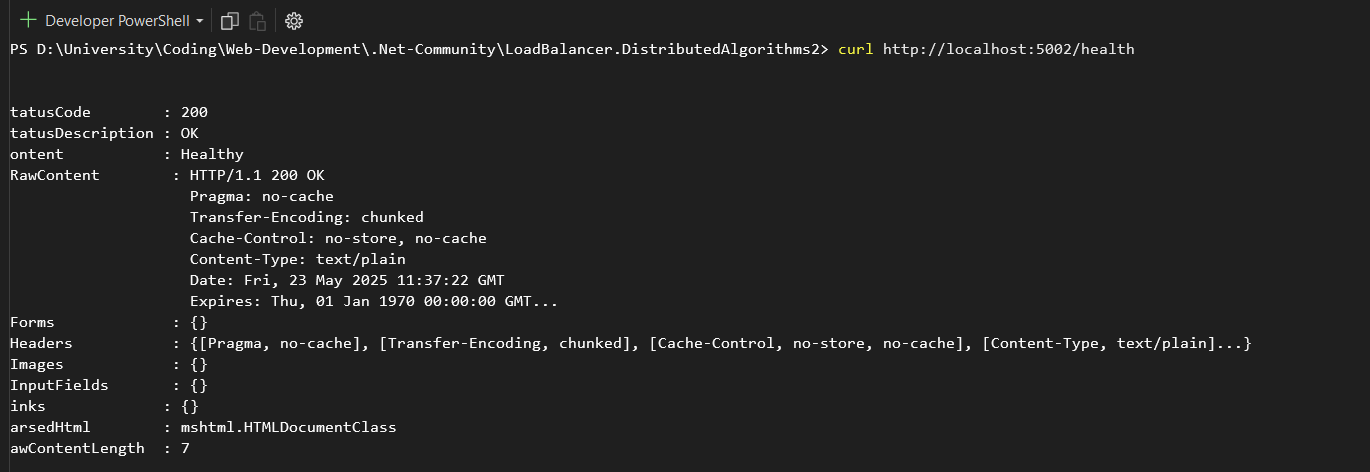
Check the terminal for logs. You’ll see Redis starting, followed by the app logs. If it fails, peek at the troubleshooting section below.

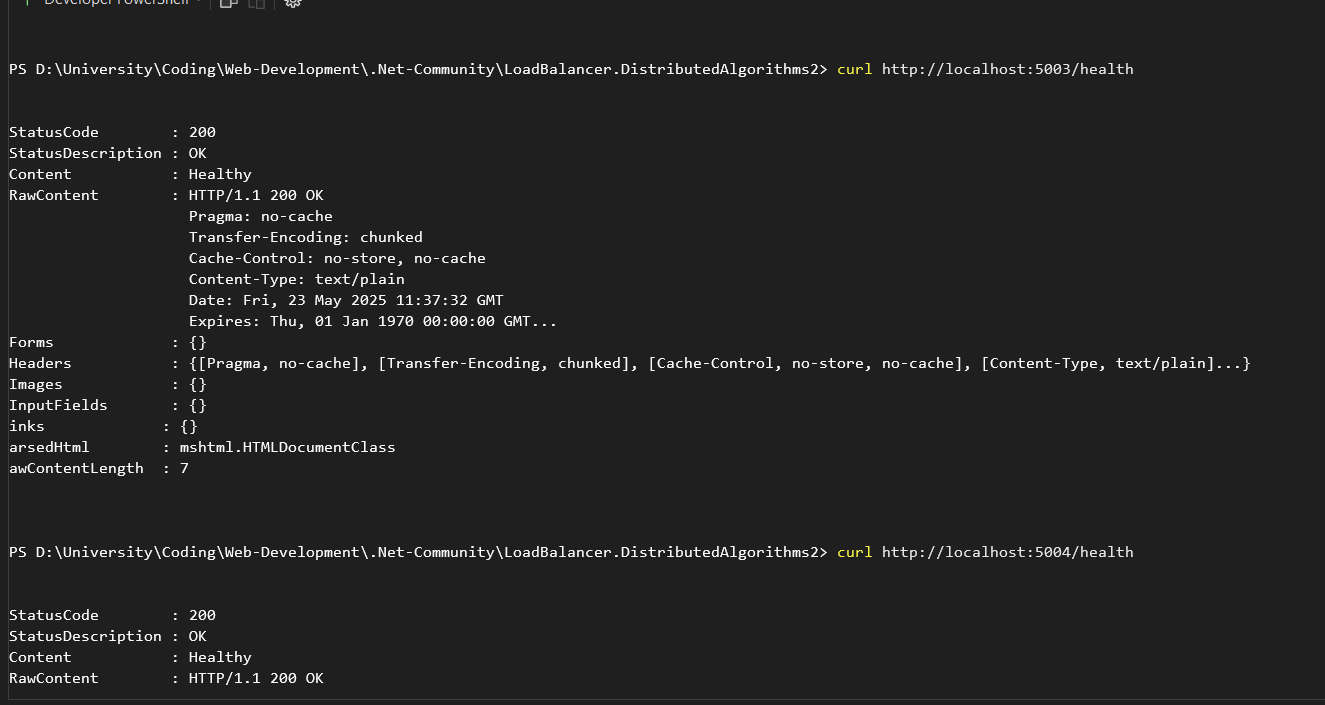
**Run Samples** :

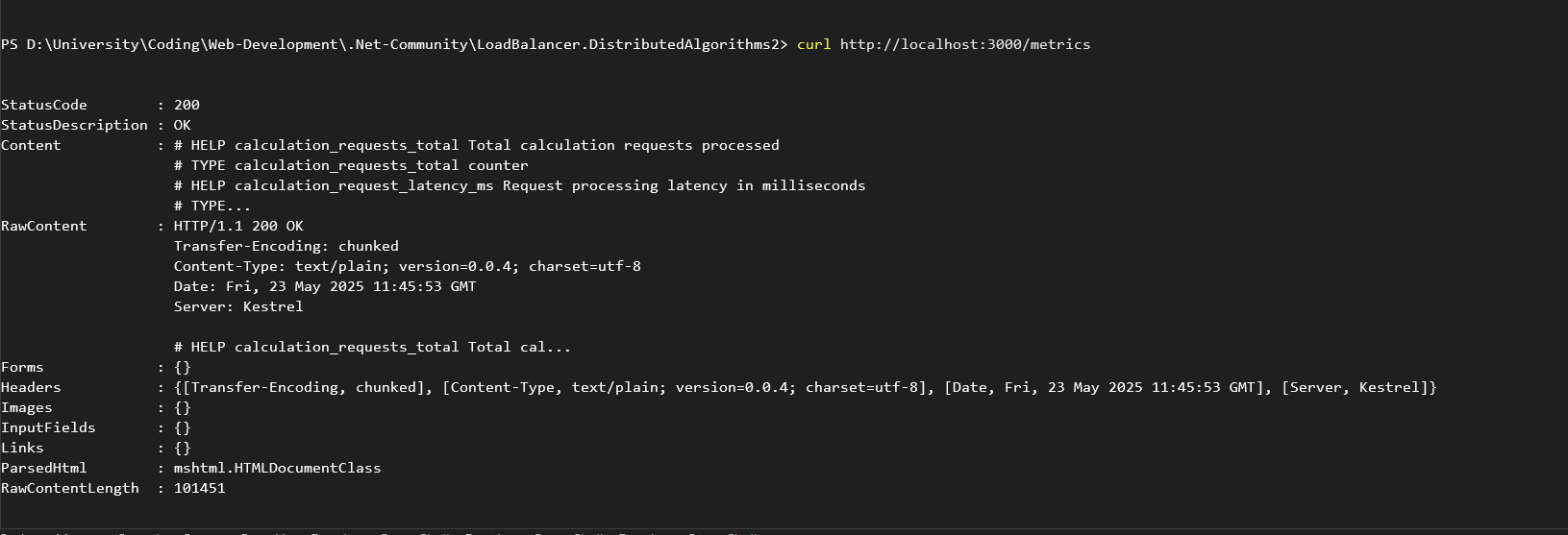












Name: Mohammed Abdelaleem Mostafa Abdeldaym  
Section: 5