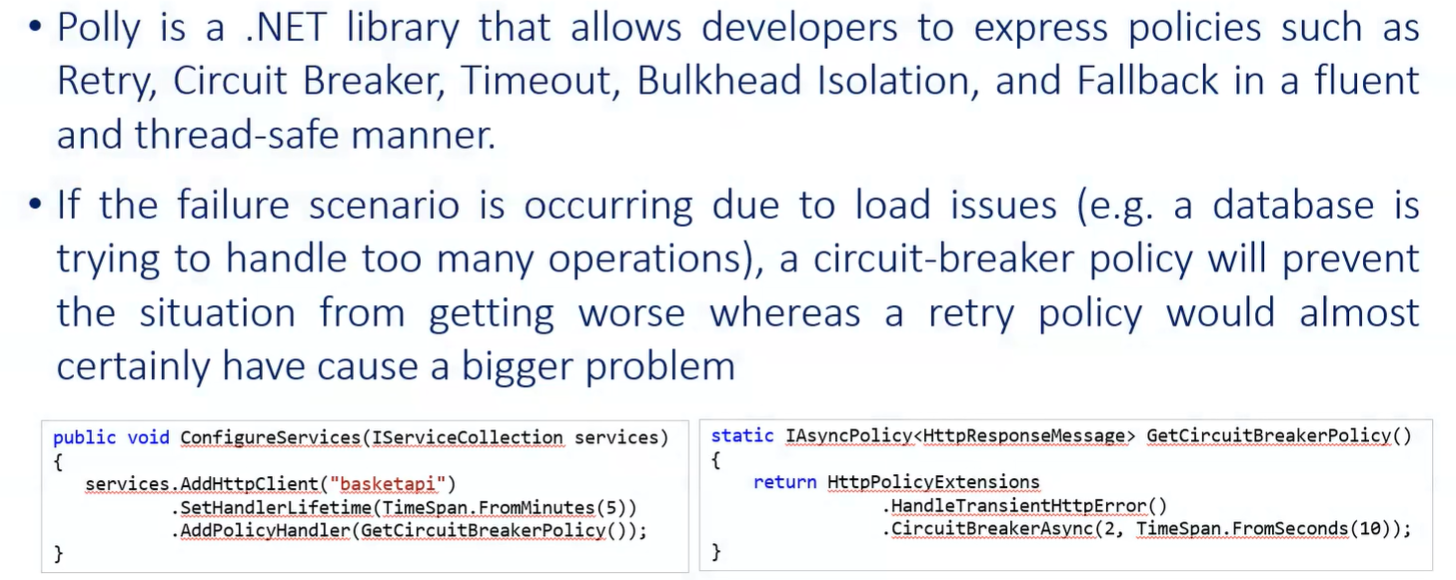
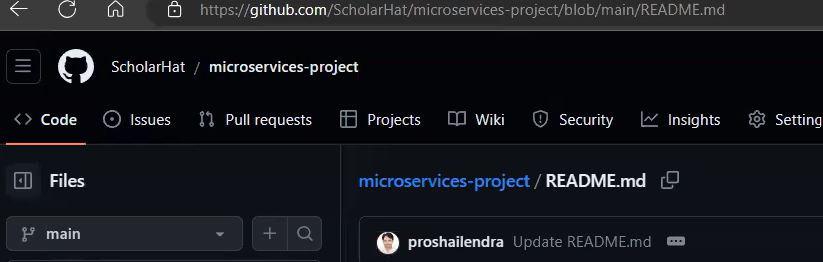
[**1. SECTION – 1 : MircroService Circuit Breaker and Retry Pattern.** 1](#_Toc173436187)

[**2. SECTION – 1.1 : Logging Service with ELK** 4](#_Toc173436188)

# **1. SECTION – 1 : MircroService Circuit Breaker and Retry Pattern.**

1. We use the same solution i.e **Cart and Payment Service -Saga Pattern**.



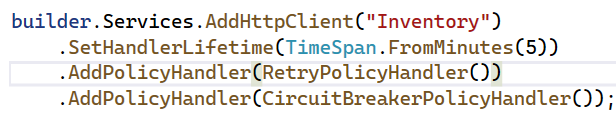


1. Create a blank solution name it as MicroservicesPolly.
2. Add a ASP.NET Core Web API project name as InventoryService inside MicroservicesPolly solution.
3. Create InventoryController and define the action method inside InventoryService project.
4. Add a ASP.NET Core Web API project name as OrderService inside MicroservicesPolly solution.
5. Create OrderController inside OrderService project.
6. Install the below NuGet Package for the OrderService.

A screenshot of a computer

Description automatically generated

1. Refer the Microsoft official github for Polly for documentation.(<https://github.com/App-vNext/Polly>)
2. Define Polly policy under OrderService of Program.cs file.

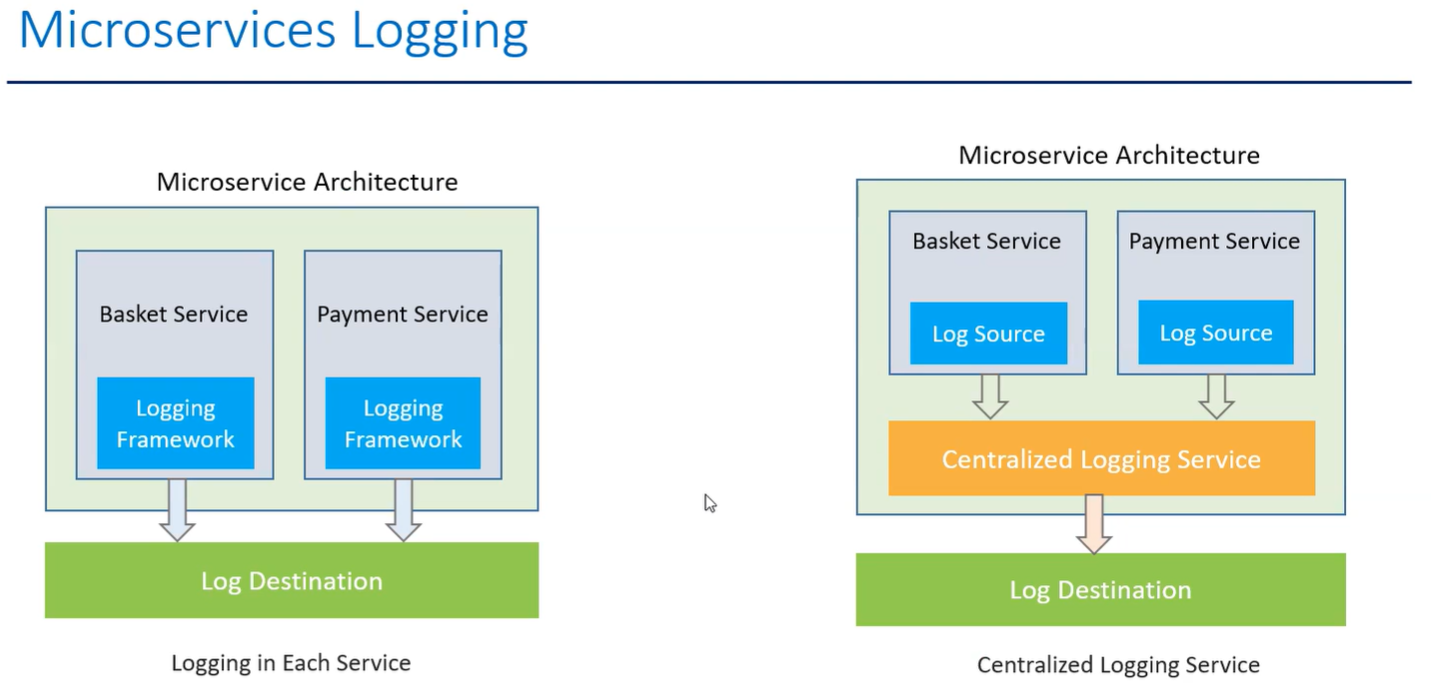


A screen shot of a computer program

Description automatically generated

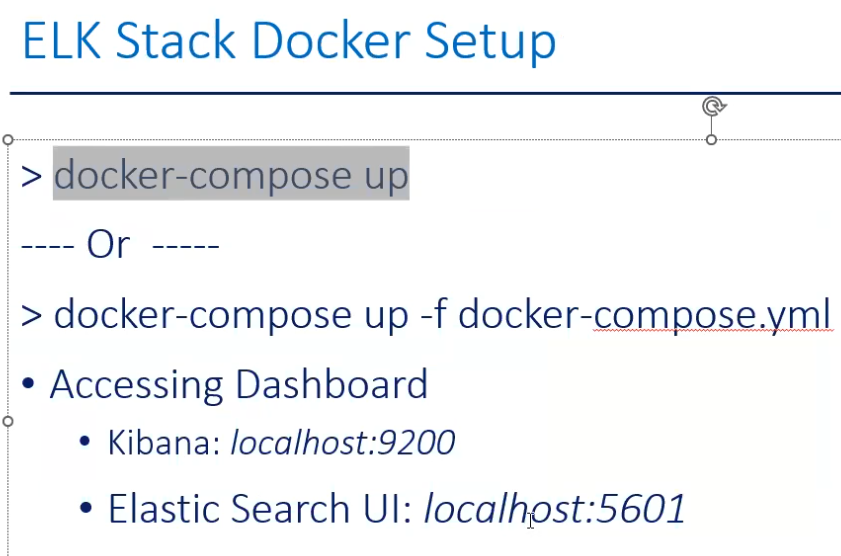
1. Add Order Model inside OrderService.
2. Add the ApiAddress configuration in OrderService of appsettings.json file.
3. Setup 2 projects as start up project and run the applications.
4. Make a call for OrderService. It gives the below response. (Please check with OrderService call and note down the observation.)



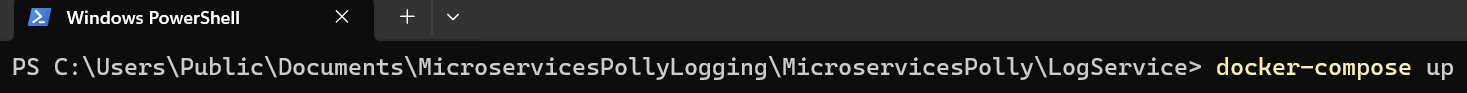


# **2. SECTION – 1.1 : Logging Service with ELK**

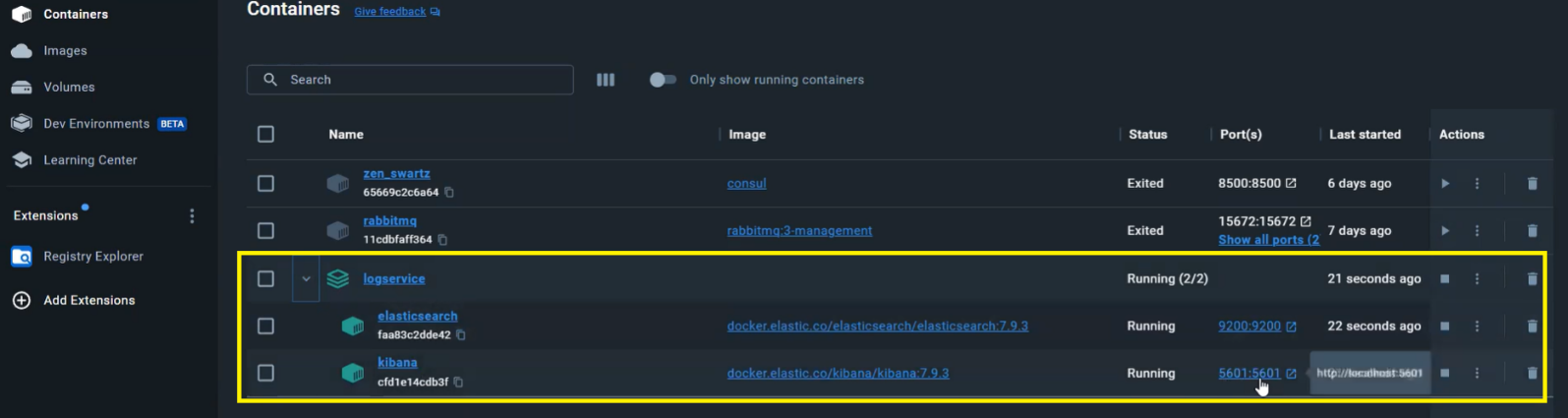
1. Create a ASP.NET Core Web API project name it as LogService.
2. Create a LogController and Model inside the LogService project.
3. Make a docker-compose.yml file for ELK.



1. Run the docker command in command prompt.



* docker-compose up -f docker-compose.yml
* **Note:** If the docker compose file is in another directory, then give the respective directory path, else it will not work.



Define the ElasticSearch and Serilog configuration in LogService project of appsettings.json file.

{

"Serilog": {

"Using": [],

"MinimumLevel": {

"Default": "Error"

}

},

"AllowedHosts": "\*",

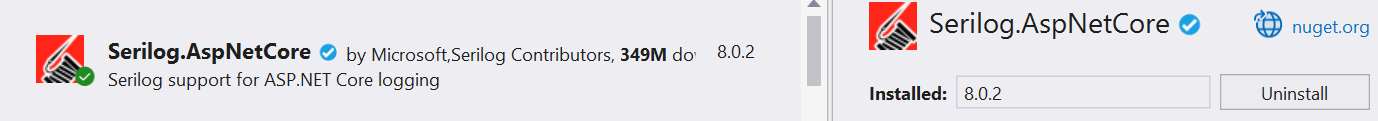
"ElasticSearch": {

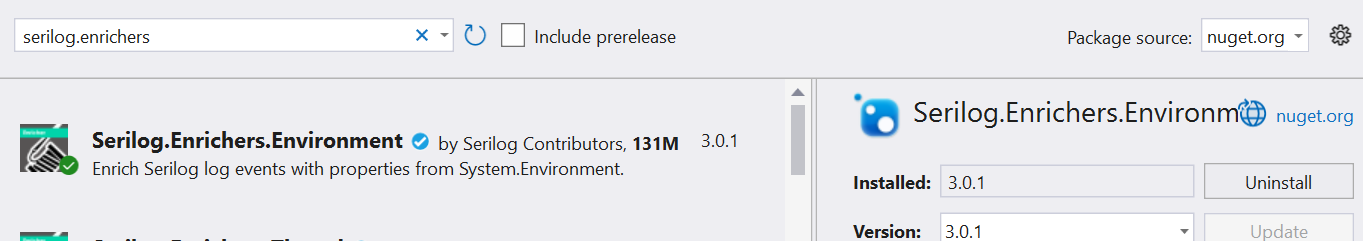
"Uri": "http://localhost:9200"

}

}

1. Add the below Serilog NuGet Packages.

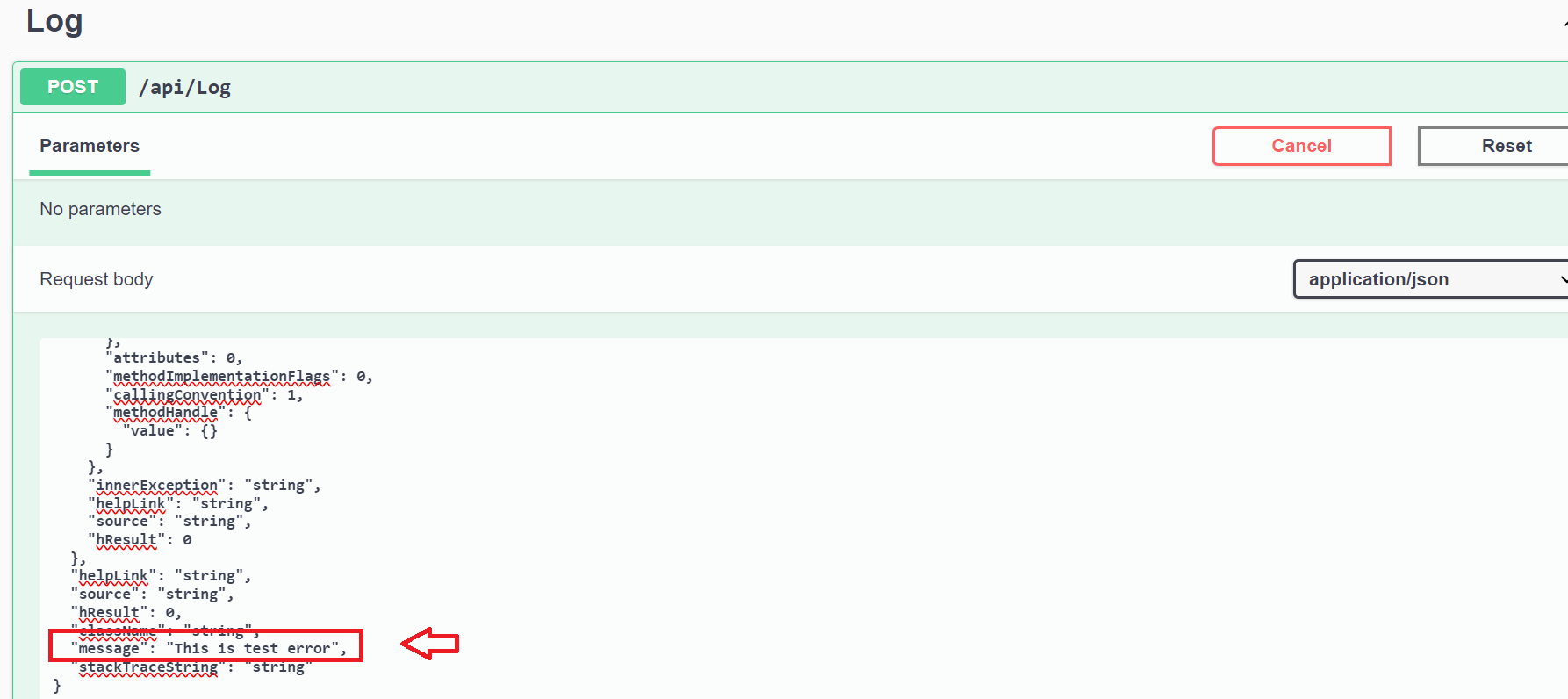


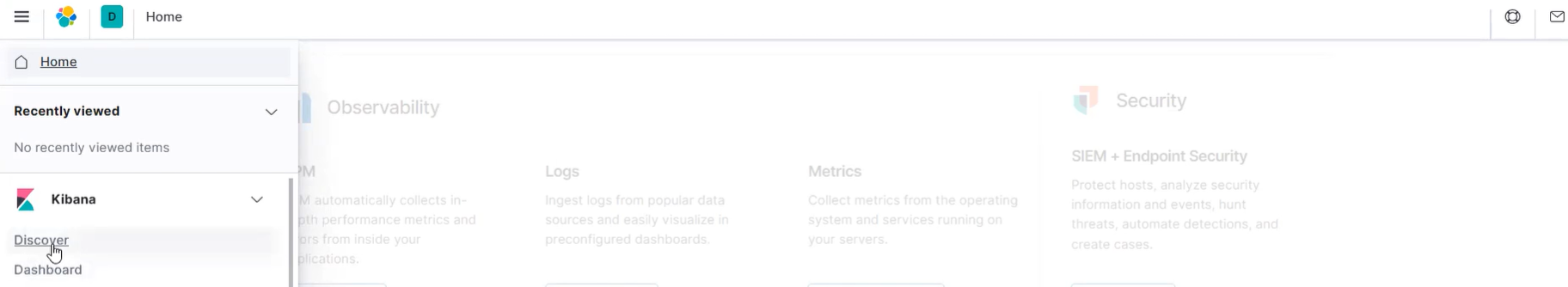


A screenshot of a computer

Description automatically generated

1. Add the Serilog configuration in Program.cs file of the LogService project.
2. Add the configuration of LogService project port (ApiAddress) into OrderService project of appsettings.json file.
3. Register the Log api address as HttpClient into Program.cs file of OrderService project.





1. Post the request through OrderService and check the error log in kibana.

**NOTE :** Here we use docker-compose.yml file but in (**Cart and Payment Service - Saga Pattern**) we use ELK, and we run the docker-compose.yml file by using **ELK > docker-elk** Project.

If Docker container is not working in this project then refer and run the docker container using the **ELK > docker-elk** Project.