

Project Documentation

Introduction

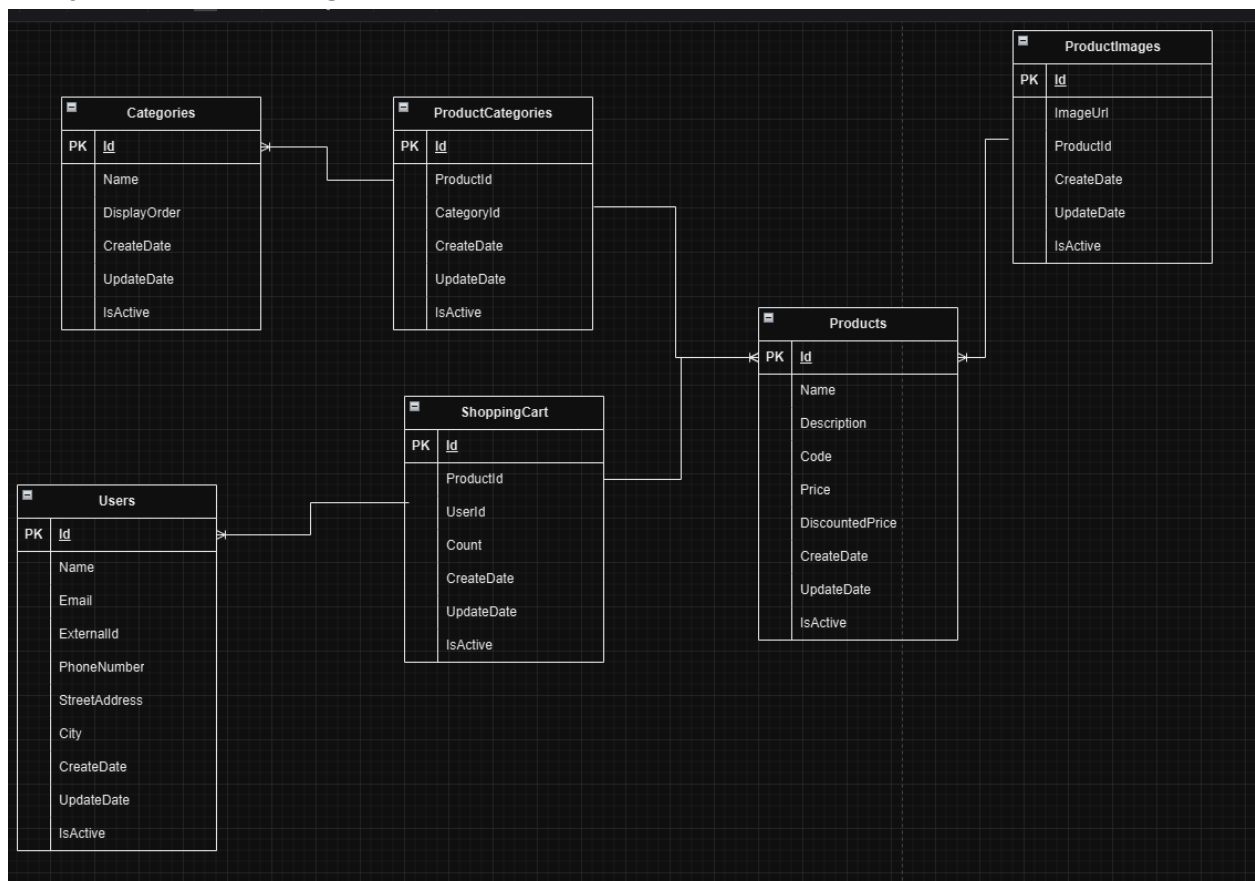
An E-Commerce web application using C# ASP .Net Core and Microservice architecture with Docker and Kubernetes

Prerequisites

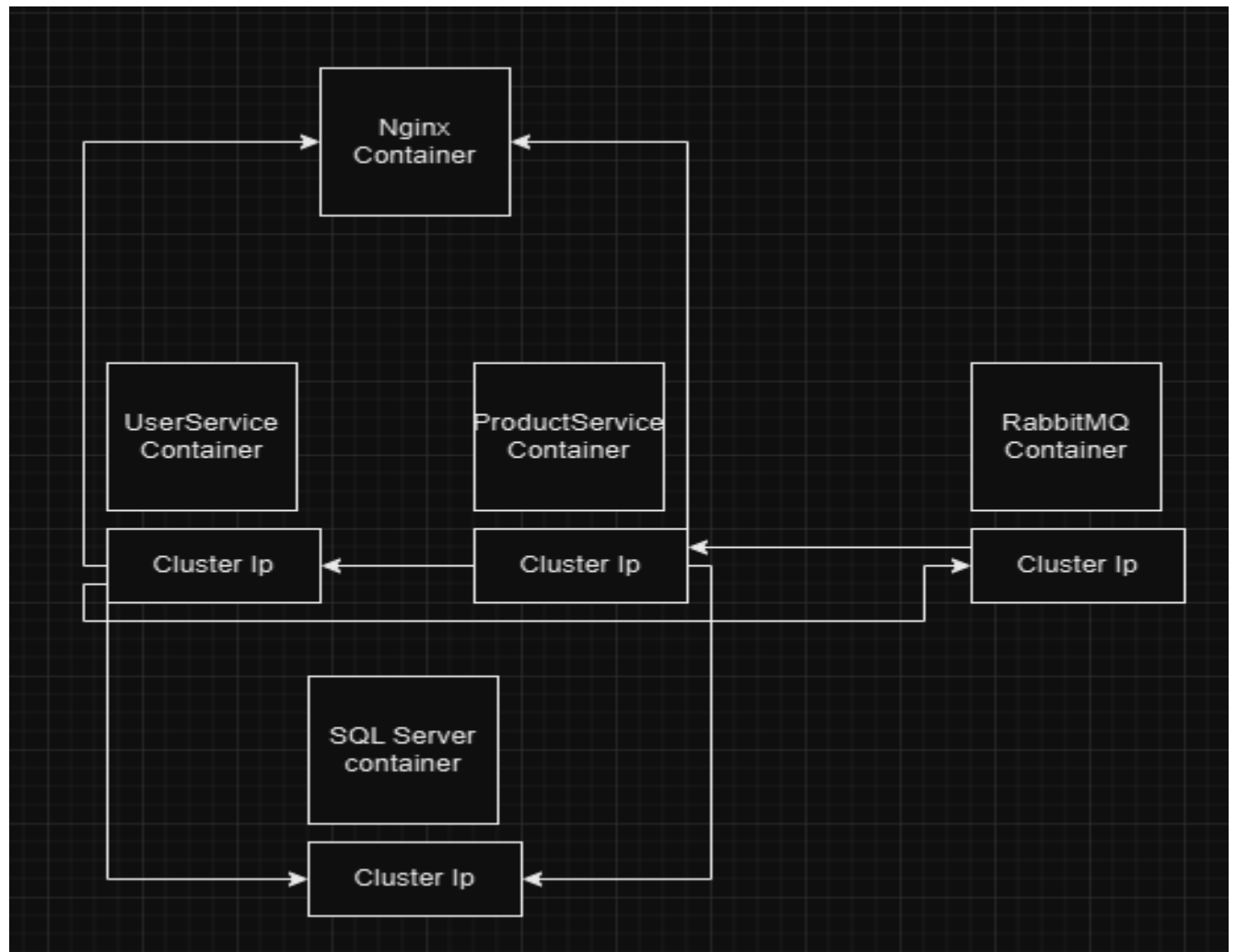
- .NET SDK
- Docker
- Kubernetes (kubectl)
- NGINX Ingress Controller
- SQL Server (MSSQL)
- Postman

Architecture Overview

Entity Relationship Diagram



Microservice Architecture



Setup Instructions

Local Development

1. Clone the Repositories

URLs:

<https://github.com/sbarrettoarcany/MicroServiceECommerce-UserService.git>

<https://github.com/sbarrettoarcany/MicroServiceEcommerce-ProductService.git>

<https://github.com/sbarrettoarcany/MicroserviceEcommerce-K8S.git>

```
bash
```

```
git clone <repository-url>
```

```
cd <repository-directory>
```

2. Install Dependencies

```
bash
```

```
dotnet restore
```

3. Run the Application Locally

```
bash
```

```
dotnet run
```

Ex.

```
kubectl apply -f ecommerceproductservice-depl.yaml
```

```
kubectl apply -f ecommerceuserservice-depl.yaml
```

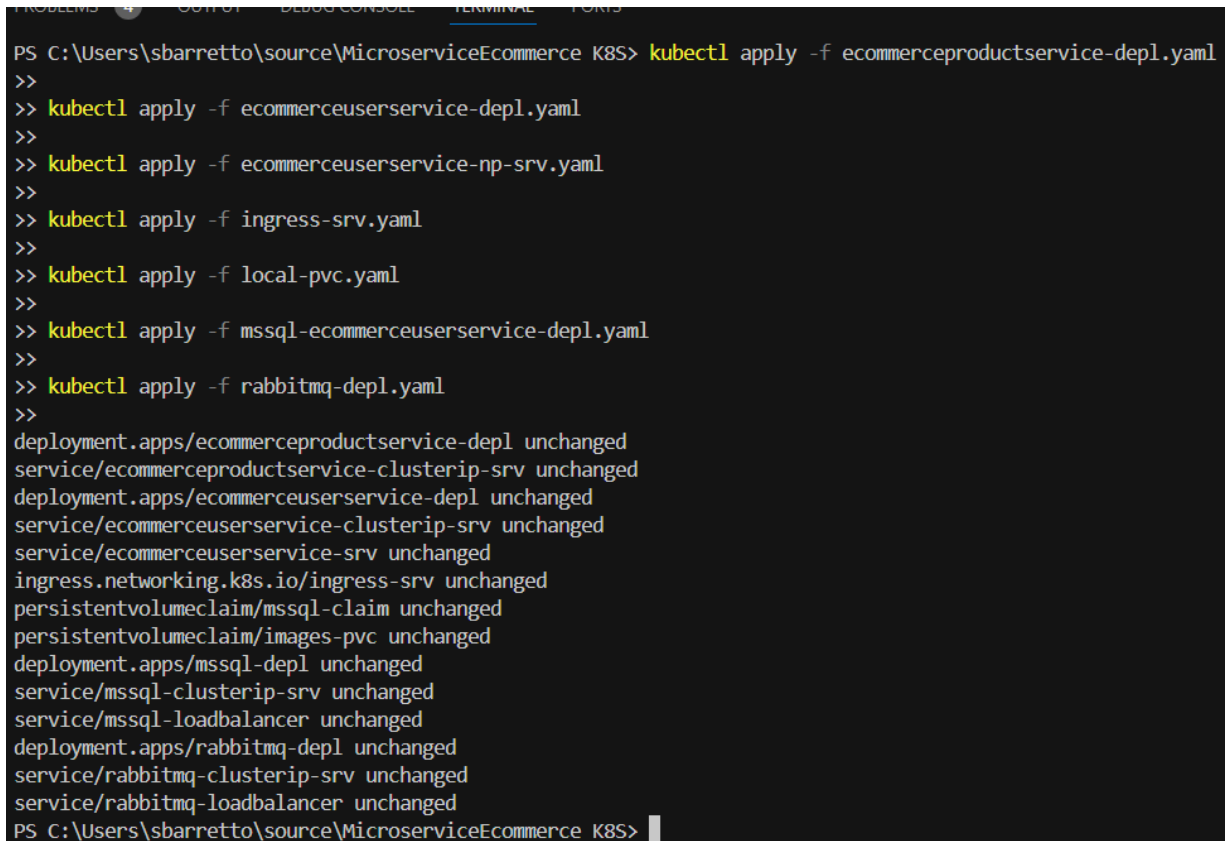
```
kubectl apply -f ecommerceuserservice-np-srv.yaml
```

```
kubectl apply -f ingress-srv.yaml
```

```
kubectl apply -f local-pvc.yaml
```

```
kubectl apply -f mssql-ecommerceuserservice-depl.yaml
```

```
kubectl apply -f rabbitmq-depl.yaml
```



```
PS C:\Users\sbarretto\source\MicroserviceEcommerce K8S> kubectl apply -f ecommerceproductservice-depl.yaml
>>
>> kubectl apply -f ecommerceuserservice-depl.yaml
>>
>> kubectl apply -f ecommerceuserservice-np-srv.yaml
>>
>> kubectl apply -f ingress-srv.yaml
>>
>> kubectl apply -f local-pvc.yaml
>>
>> kubectl apply -f mssql-ecommerceuserservice-depl.yaml
>>
>> kubectl apply -f rabbitmq-depl.yaml
>>
deployment.apps/eccommerceproductservice-depl unchanged
service/eccommerceproductservice-clusterip-srv unchanged
deployment.apps/eccommerceuserservice-depl unchanged
service/eccommerceuserservice-clusterip-srv unchanged
service/eccommerceuserservice-srv unchanged
ingress.networking.k8s.io/ingress-srv unchanged
persistentvolumeclaim/mssql-claim unchanged
persistentvolumeclaim/images-pvc unchanged
deployment.apps/mssql-depl unchanged
service/mssql-clusterip-srv unchanged
service/mssql-loadbalancer unchanged
deployment.apps/rabbitmq-depl unchanged
service/rabbitmq-clusterip-srv unchanged
service/rabbitmq-loadbalancer unchanged
PS C:\Users\sbarretto\source\MicroserviceEcommerce K8S>
```

2. Verify Deployment

```
kubectl get pods
```

```
kubectl get services
```

```

service/rabbitmq-loadbalancer unchanged
PS C:\Users\sbarretto\source\MicroserviceEcommerce K8S> kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
ecommerceproductservice-depl-df887d5d5-xk7gg  1/1     Running   8 (136m ago)  61d
ecommerceuserservice-depl-67d8c5d74c-vb7n7    1/1     Running   15 (136m ago)  67d
mssql-depl-5879d88f8-chkw8                 1/1     Running   14 (137m ago)  75d
rabbitmq-depl-5955868749-zshsb              1/1     Running   12 (137m ago)  73d
PS C:\Users\sbarretto\source\MicroserviceEcommerce K8S> kubectl get services
NAME                                TYPE        CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
ecommerceproductservice-clusterip-srv  ClusterIP   10.96.57.169    <none>           8080/TCP         80d
ecommerceuserservice-clusterip-srv     ClusterIP   10.102.28.150   <none>           8080/TCP         81d
ecommerceuserservice-srv               NodePort    10.106.98.156   <none>           8080:30131/TCP   81d
kubernetes                             ClusterIP   10.96.0.1       <none>           443/TCP          88d
mssql-clusterip-srv                   ClusterIP   10.103.76.1     <none>           1433/TCP         81d
mssql-loadbalancer                    LoadBalancer 10.111.246.203  localhost        1433:32207/TCP   81d
rabbitmq-clusterip-srv                ClusterIP   10.109.65.7     <none>           15672/TCP,5672/TCP 73d
rabbitmq-loadbalancer                  LoadBalancer 10.100.34.10    localhost        15672:30596/TCP,5672:31761/TCP 73d
PS C:\Users\sbarretto\source\MicroserviceEcommerce K8S> kubectl get deployments
NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
ecommerceproductservice-depl       1/1     1             1           80d
ecommerceuserservice-depl          1/1     1             1           80d
mssql-depl                         1/1     1             1           81d
rabbitmq-depl                      1/1     1             1           73d
PS C:\Users\sbarretto\source\MicroserviceEcommerce K8S>

```

NGINX Ingress Configuration

1. Install NGINX Ingress Controller

Go to NGINX github repo

<https://github.com/kubernetes/ingress-nginx/blob/main/docs/deploy/index.md#quick-start>

> look for the YAML manifest URL like this one

```

If you don't have Helm or if you prefer to use a YAML manifest, you can run the following command instead:

kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v1.12.0-beta.0/deploy/static/provider/cloud/deploy.yaml

```

> copy and apply locally

2. Check created NGINX container

“Kubectl get pods --namespace=ingress-nginx”

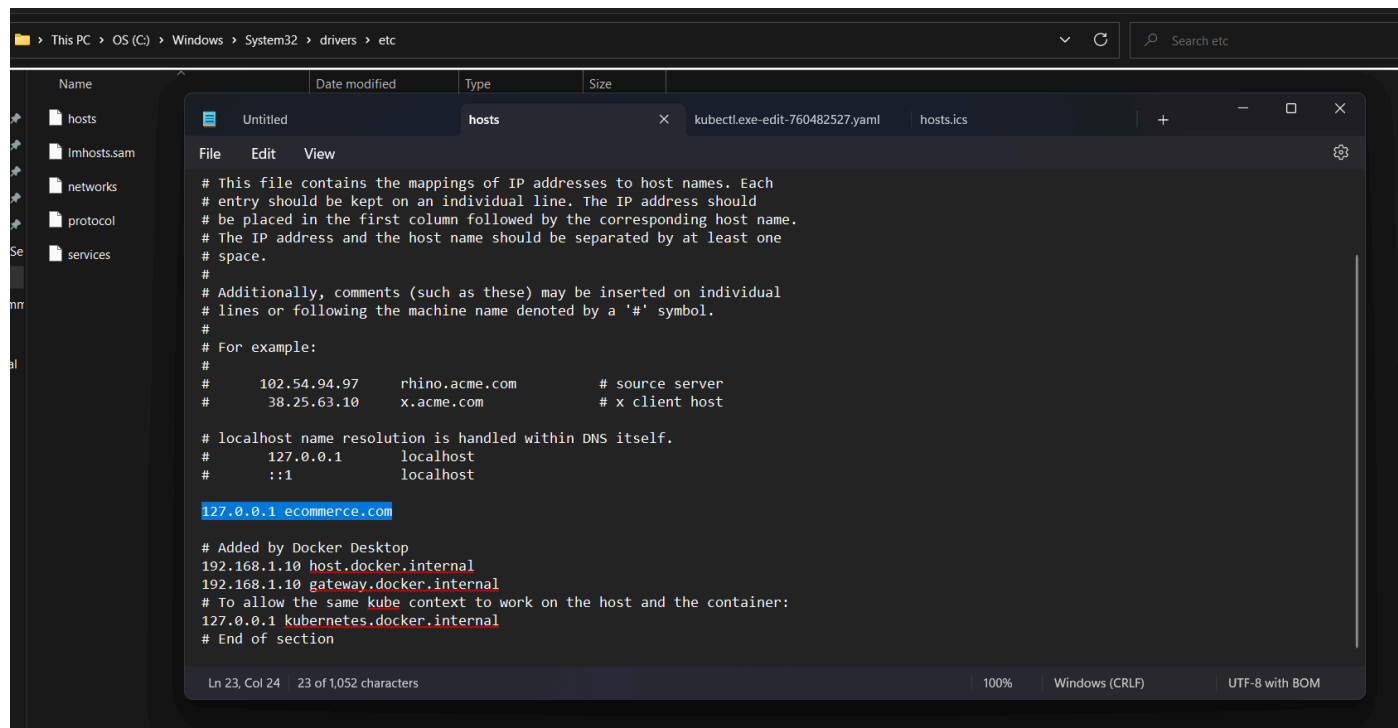
```

PS C:\Users\sbarretto\source\MicroserviceEcommerce K8S> kubectl get namespace
NAME                STATUS   AGE
default             Active   7d22h
ingress-nginx       Active   159m
kube-node-lease     Active   7d22h
kube-public         Active   7d22h
kube-system         Active   7d22h
PS C:\Users\sbarretto\source\MicroserviceEcommerce K8S> kubectl get pods --namespace=ingress-nginx
NAME                                READY   STATUS    RESTARTS   AGE
ingress-nginx-admission-create-tdkrf 0/1     Completed 0           155m
ingress-nginx-admission-patch-4qctt   0/1     Completed 0           155m
ingress-nginx-controller-7d4db76476-zvtqv 1/1     Running   0           155m
PS C:\Users\sbarretto\source\MicroserviceEcommerce K8S>

```

3. Access the Application

Go to System32>drivers>etc and add a line for ecommerce URL



RabbitMQ Access

Go to ecommerce.com:15672 (port setup in rabbitmq-depl.yaml) > use “guest” as username and password

