How to deploy .NET8 CRUD WebAPI Azure CosmosDB Microservice to Azure Container Instance (ACI)

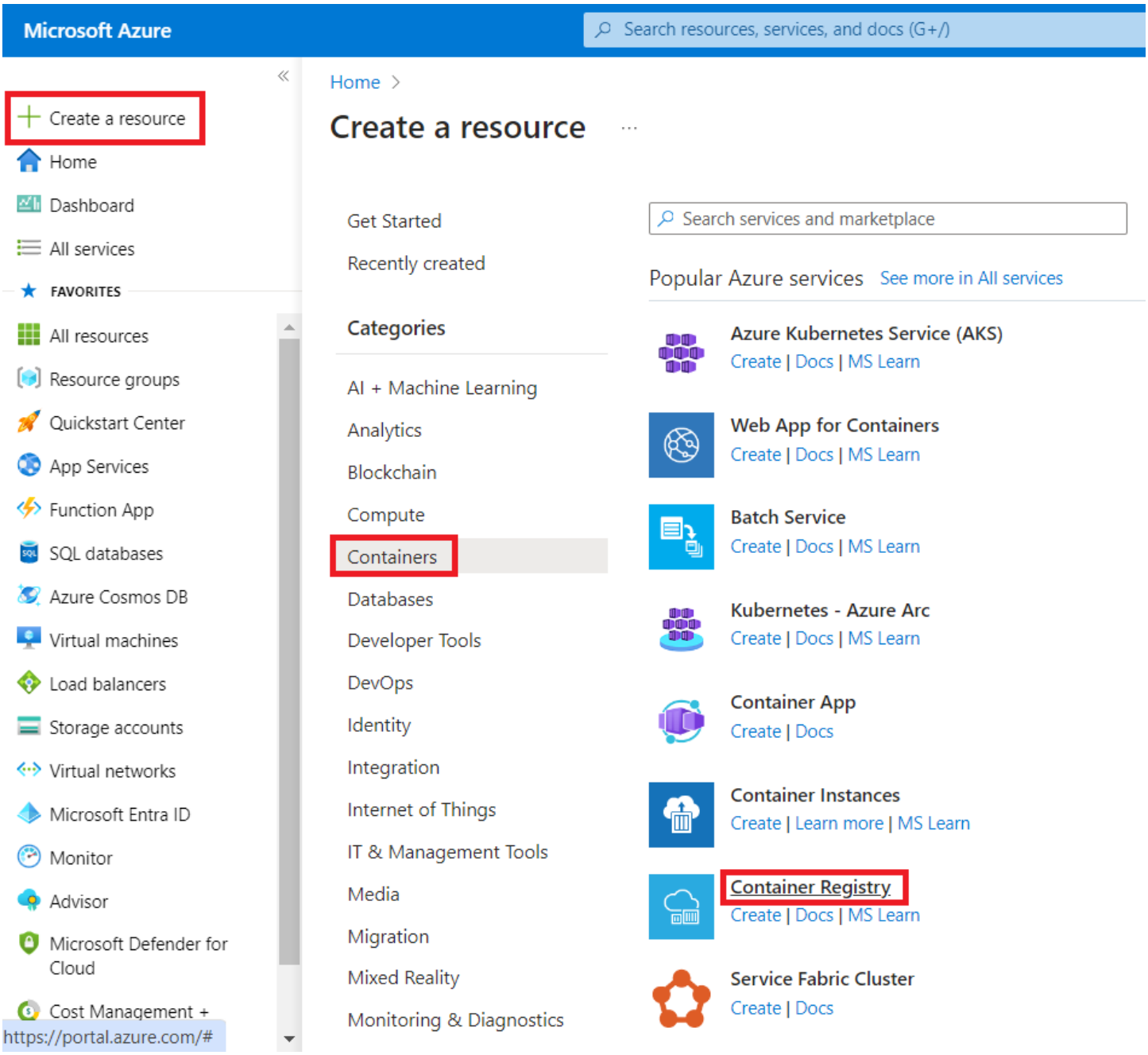
0. Prerequisite create .NET8 CRUD WebAPI connected to Azure CosmosDB

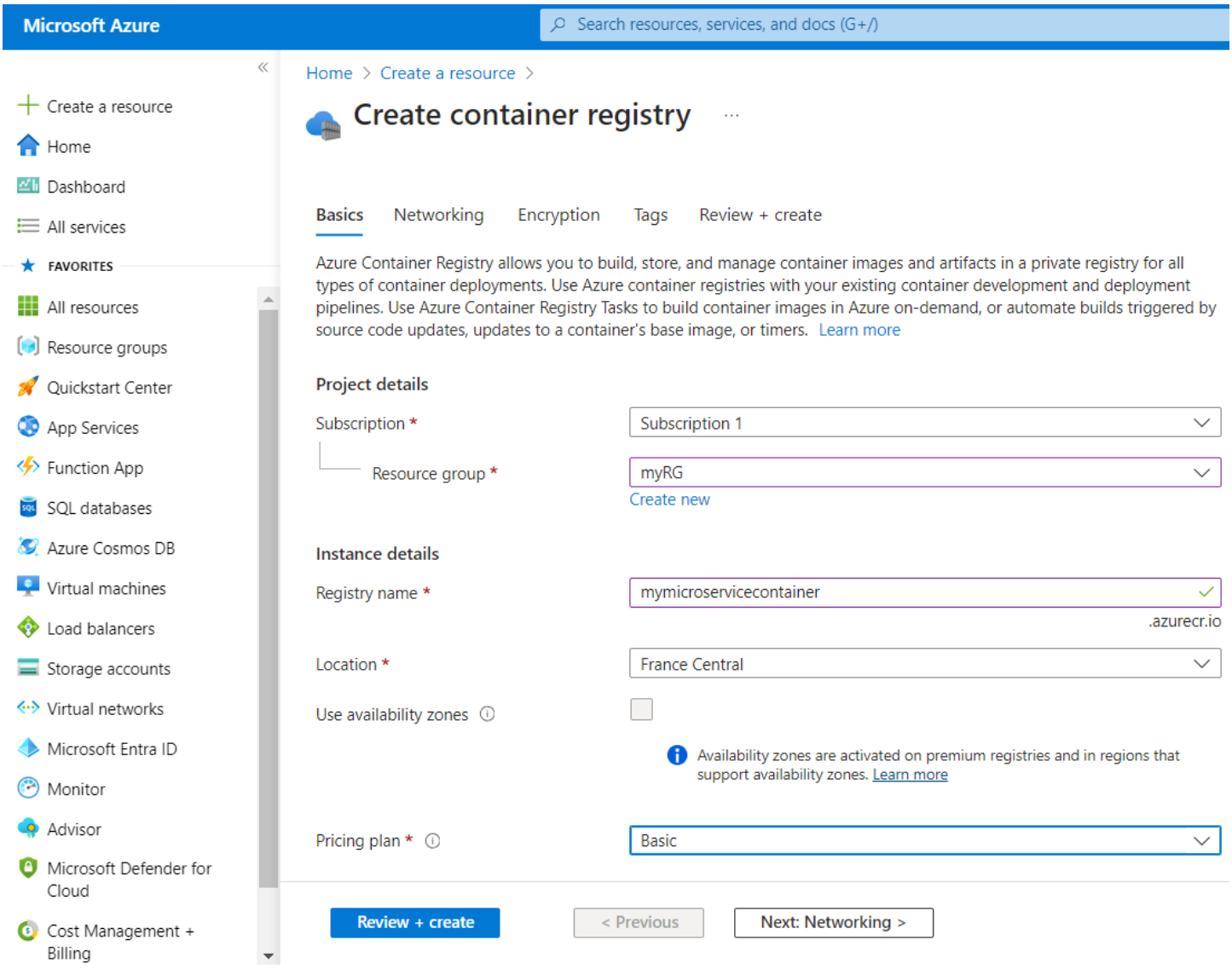
See the code in this repo: https://github.com/luiscoco/MicroServices_dotNET8_CRUD_WebAPI-CosmosDB-deployed_to_Azure_Container_Instance

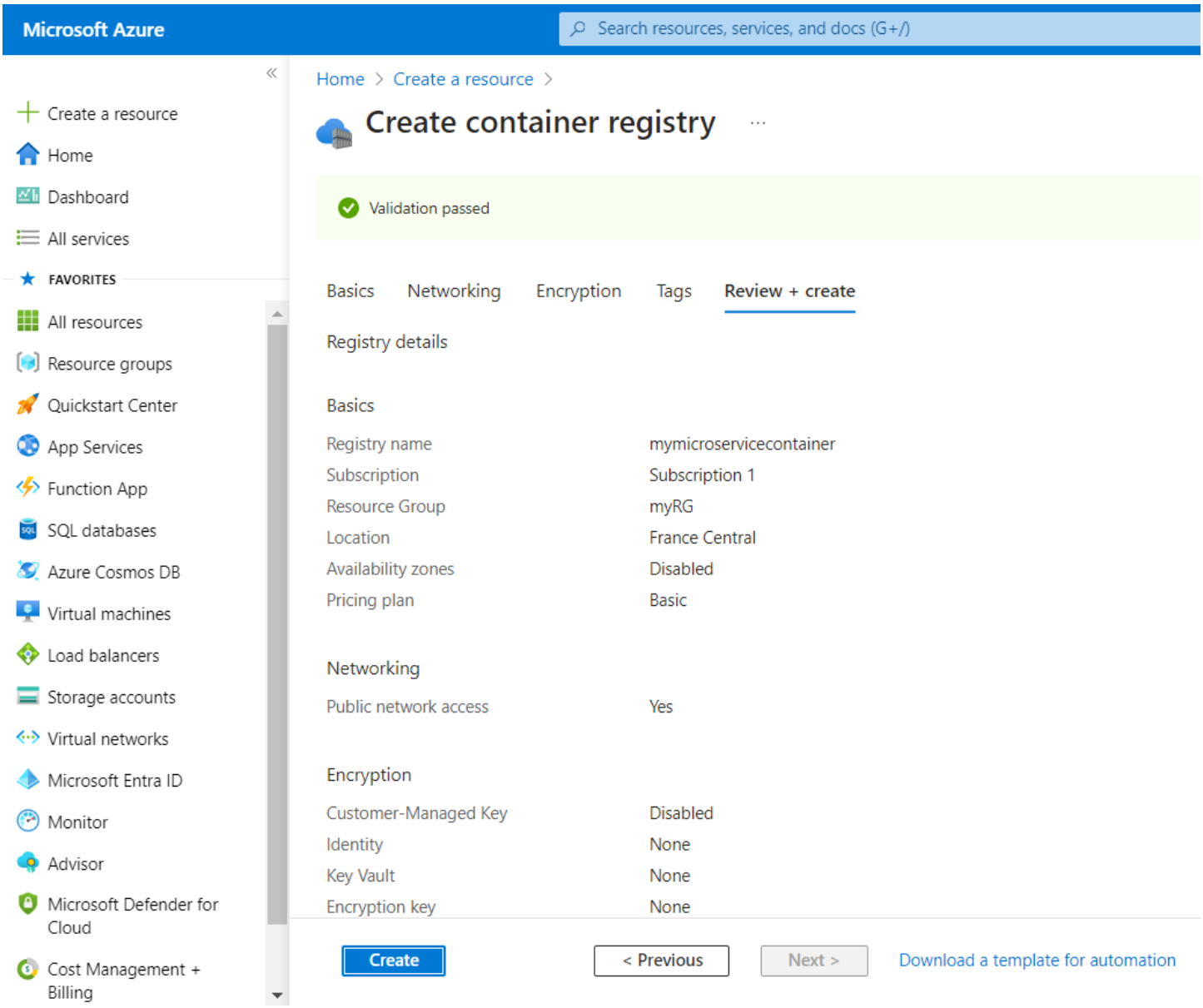
Also see this repo: https://github.com/luiscoco/MicroServices_dotNET8_CRUD_WebAPI-AzureCosmosDB

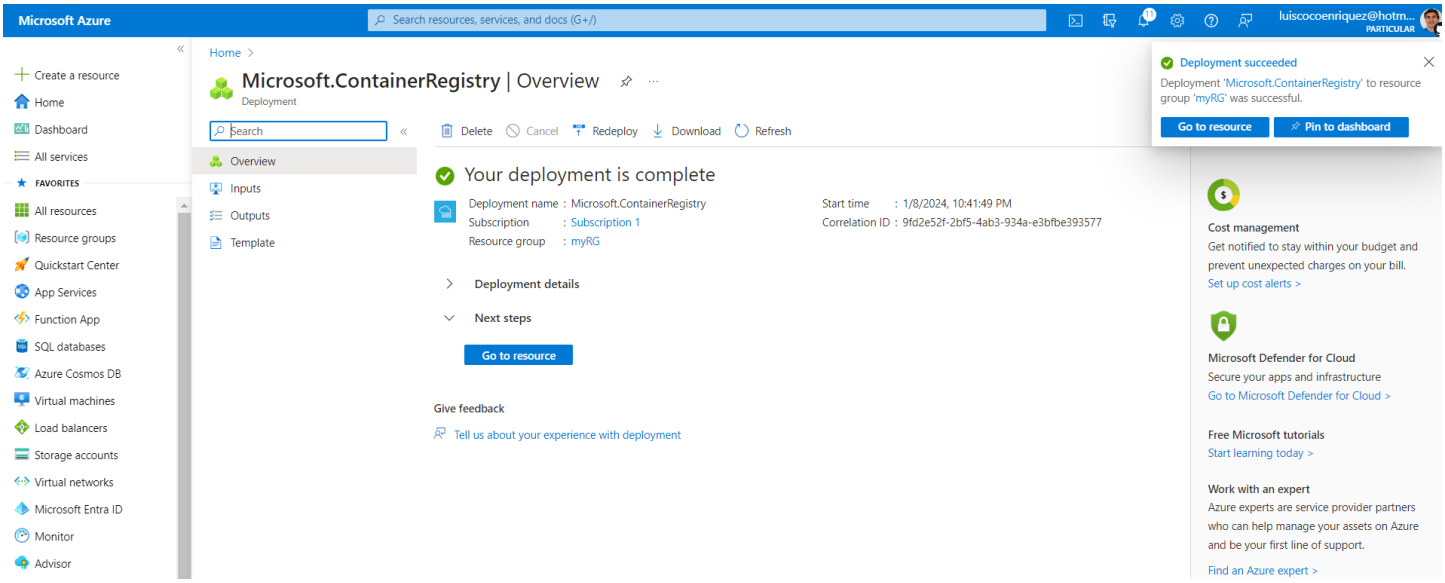
1. Create an Azure Container Registry

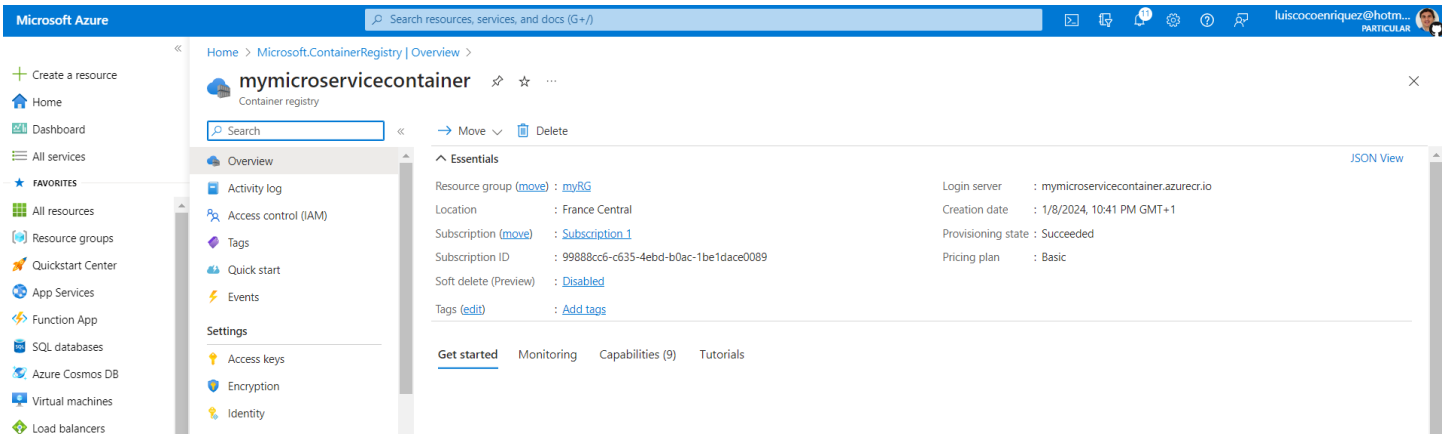
We create Azure Container Registry service for uploading the .NET CRUD WebAPI docker image


Microsoft Azure
Search resources, services, and docs (G+/)
Create a resource
Home
Dashboard
All services
FAVORITES
All resources
Resource groups
Quickstart Center
App Services
Function App
SQL databases
Azure Cosmos DB
Virtual machines
Load balancers
Storage accounts
Virtual networks
Microsoft Entra ID
Monitor
Advisor
Microsoft Defender for Cloud
Cost Management +
https://portal.azure.com/#
Home
Create a resource
Get Started
Recently created
Categories
AI + Machine Learning
Analytics
Blockchain
Compute
Containers
Databases
Developer Tools
DevOps
Identity
Integration
Internet of Things
IT & Management Tools
Media
Migration
Mixed Reality
Monitoring & Diagnostics
Search services and marketplace
Popular Azure services
See more in All services
Azure Kubernetes Service (AKS)
Create | Docs | MS Learn
Web App for Containers
Create | Docs | MS Learn
Batch Service
Create | Docs | MS Learn
Kubernetes - Azure Arc
Create | Docs | MS Learn
Container App
Create | Docs
Container Instances
Create | Learn more | MS Learn
Container Registry
Create | Docs | MS Learn
Service Fabric Cluster
Create | Docs


Microsoft Azure
Search resources, services, and docs (G+/)
Create a resource
Home
Dashboard
All services
FAVORITES
All resources
Resource groups
Quickstart Center
App Services
Function App
SQL databases
Azure Cosmos DB
Virtual machines
Load balancers
Storage accounts
Virtual networks
Microsoft Entra ID
Monitor
Advisor
Microsoft Defender for Cloud
Cost Management + Billing
Home > Create a resource >
Create container registry
Basics
Networking
Encryption
Tags
Review + create
Azure Container Registry allows you to build, store, and manage container images and artifacts in a private registry for all types of container deployments. Use Azure container registries with your existing container development and deployment pipelines. Use Azure Container Registry Tasks to build container images in Azure on-demand, or automate builds triggered by source code updates, updates to a container's base image, or timers. Learn more
Project details
Subscription *
Subscription 1
Resource group *
myRG
Create new
Instance details
Registry name *
mymicroservicecontainer
.azurecr.io
Location *
France Central
Use availability zones
Availability zones are activated on premium registries and in regions that support availability zones. Learn more
Pricing plan *
Basic
Review + create
< Previous
Next: Networking >

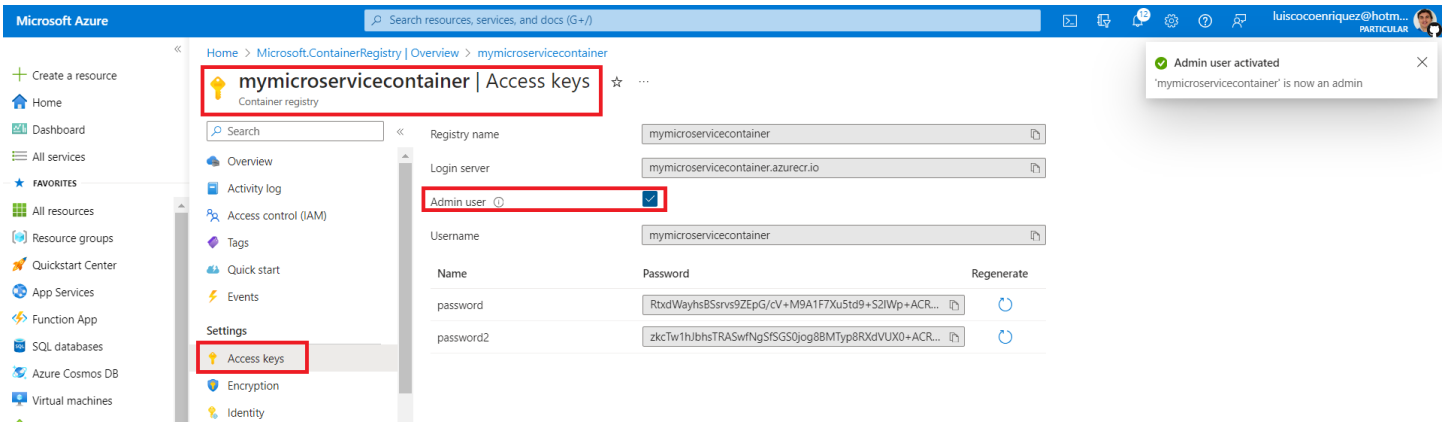

Microsoft Azure
Search resources, services, and docs (G+/)
Create a resource
Home
Dashboard
All services
FAVORITES
All resources
Resource groups
Quickstart Center
App Services
Function App
SQL databases
Azure Cosmos DB
Virtual machines
Load balancers
Storage accounts
Virtual networks
Microsoft Entra ID
Monitor
Advisor
Microsoft Defender for Cloud
Cost Management + Billing
Home > Create a resource >
Create container registry
Validation passed
Basics
Networking
Encryption
Tags
Review + create
Registry details
Basics
Registry name mymicroservicecontainer
Subscription Subscription 1
Resource Group myRG
Location France Central
Availability zones Disabled
Pricing plan Basic
Networking
Public network access Yes
Encryption
Customer-Managed Key Disabled
Identity None
Key Vault None
Encryption key None
Create
< Previous
Next >
Download a template for automation


Microsoft Azure
Search resources, services, and docs (G+/)
Home >
Microsoft.ContainerRegistry | Overview
Deployment
Search
Overview
Inputs
Outputs
Template
Delete
Cancel
Redeploy
Download
Refresh
Your deployment is complete
Deployment name : Microsoft.ContainerRegistry
Subscription : Subscription 1
Resource group : myRG
Start time : 1/8/2024, 10:41:49 PM
Correlation ID : 9fd2e52f-2bf5-4ab3-934a-e3bfbe393577
Deployment details
Next steps
Go to resource
Give feedback
Tell us about your experience with deployment
Deployment succeeded
Deployment 'Microsoft.ContainerRegistry' to resource group 'myRG' was successful.
Go to resource
Pin to dashboard
Cost management
Get notified to stay within your budget and prevent unexpected charges on your bill.
Set up cost alerts >
Microsoft Defender for Cloud
Secure your apps and infrastructure
Go to Microsoft Defender for Cloud >
Free Microsoft tutorials
Start learning today >
Work with an expert
Azure experts are service provider partners who can help manage your assets on Azure and be your first line of support.
Find an Azure expert >



2. Set the Admin User

We can enable the Admin User in the Azure Portal



Or we can enable the Admin User programmatically with Azure CLI

```
az acr update --name mymicroservicecontainer --resource-group myRG --admin-enabled true
```

Log in to Azure ACR

```
az acr login --name mymicroservicecontainer
```

3. Create a Dockerfile

With Visual Studio 2022 Community Edition we can automatically create the Dockerfile.

After creating automatically the Dockerfile we **expose the port 80**

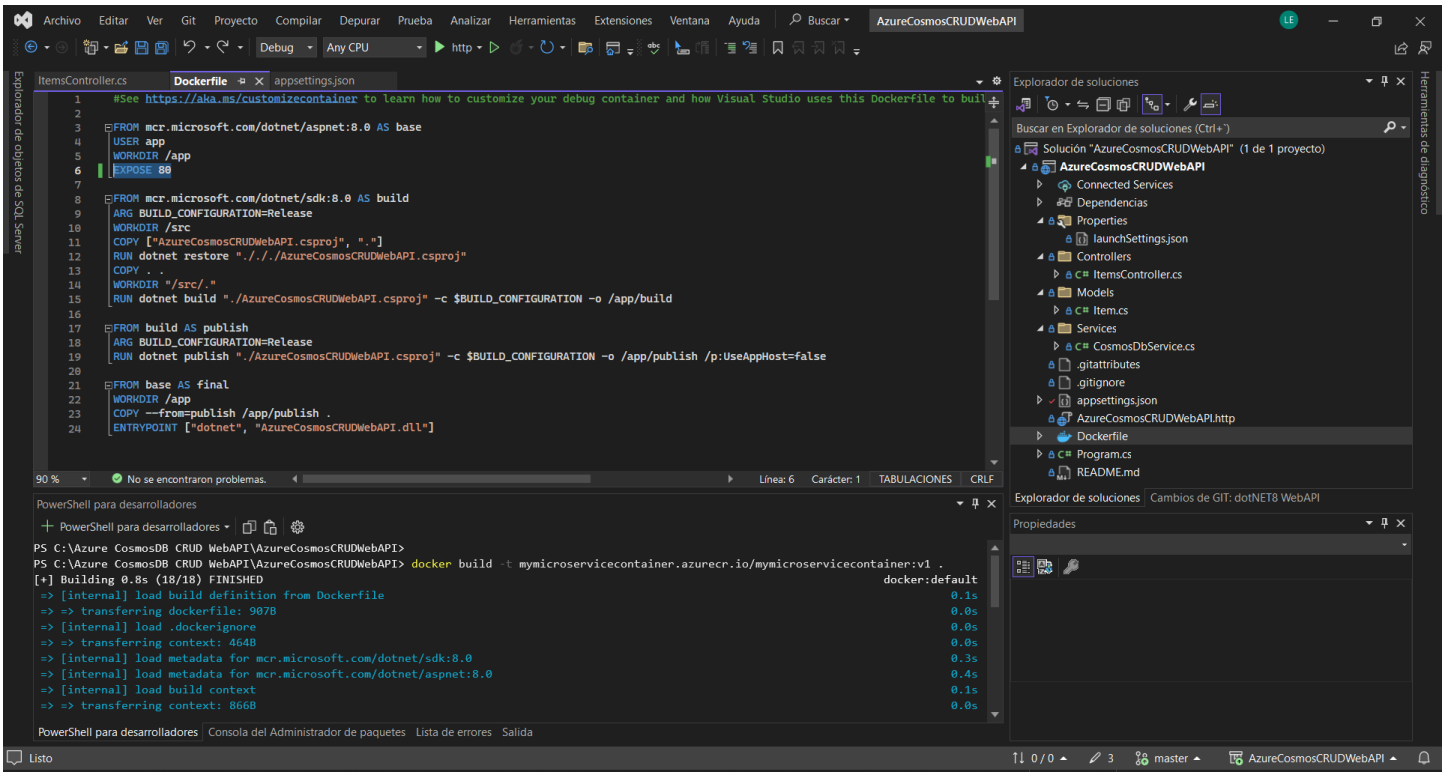
This is the Dockerfile source code

```
#See https://aka.ms/customizecontainer to learn how to customize your debug container and how

FROM mcr.microsoft.com/dotnet/aspnet:8.0 AS base
```

```
USER app
WORKDIR /app
EXPOSE 80

FROM mcr.microsoft.com/dotnet/sdk:8.0 AS build
ARG BUILD_CONFIGURATION=Release
WORKDIR /src
COPY ["AzureCosmosCRUDWebAPI.csproj", "."]
RUN dotnet restore "./././AzureCosmosCRUDWebAPI.csproj"
COPY . .
WORKDIR "/src/."
RUN dotnet build "./AzureCosmosCRUDWebAPI.csproj" -c $BUILD_CONFIGURATION -o /app/build

FROM build AS publish
ARG BUILD_CONFIGURATION=Release
RUN dotnet publish "./AzureCosmosCRUDWebAPI.csproj" -c $BUILD_CONFIGURATION -o /app/publish /p

FROM base AS final
WORKDIR /app
COPY --from=publish /app/publish .
ENTRYPOINT ["dotnet", "AzureCosmosCRUDWebAPI.dll"]
```

4. Create the Docker image

```
docker build -t mymicroservicecontainer.azurecr.io/mymicroservicecontainer:v1 .
```



5. Push the Docker image

Log in to Azure ACR

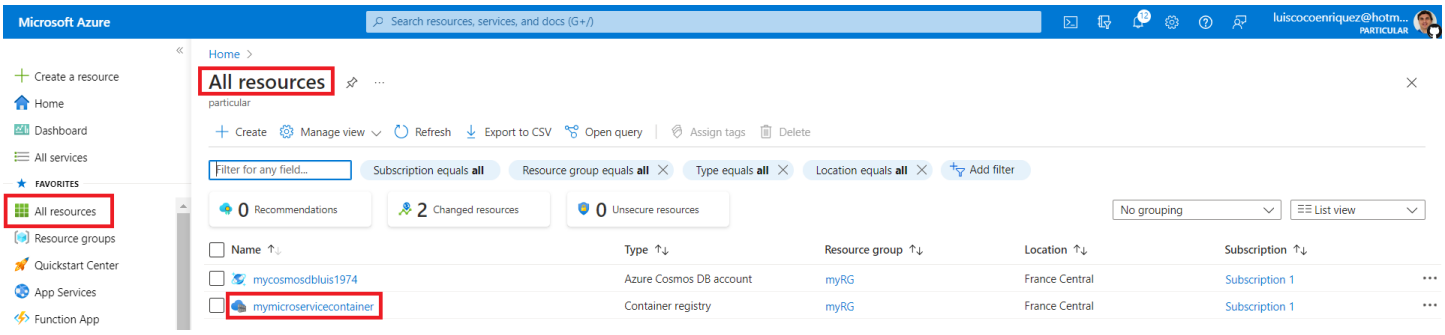
```
az acr login --name mymicroservicecontainer
```

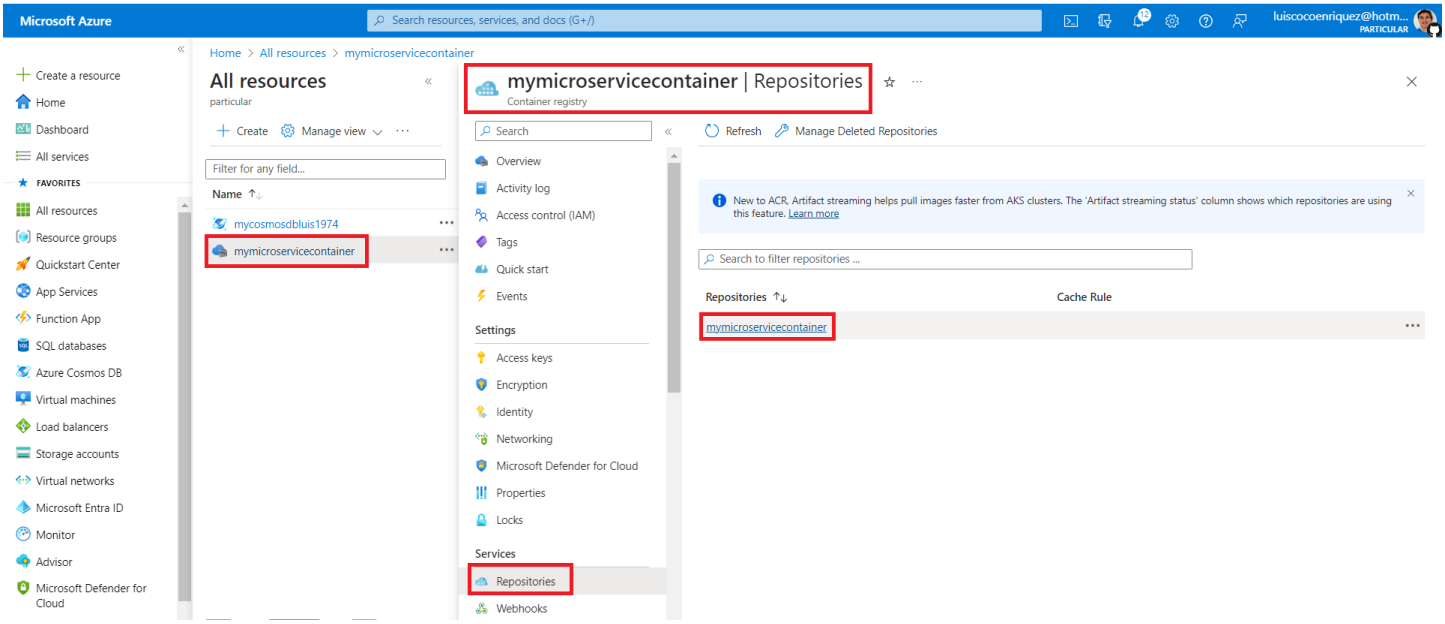
And we push the docker image to Azure ACR

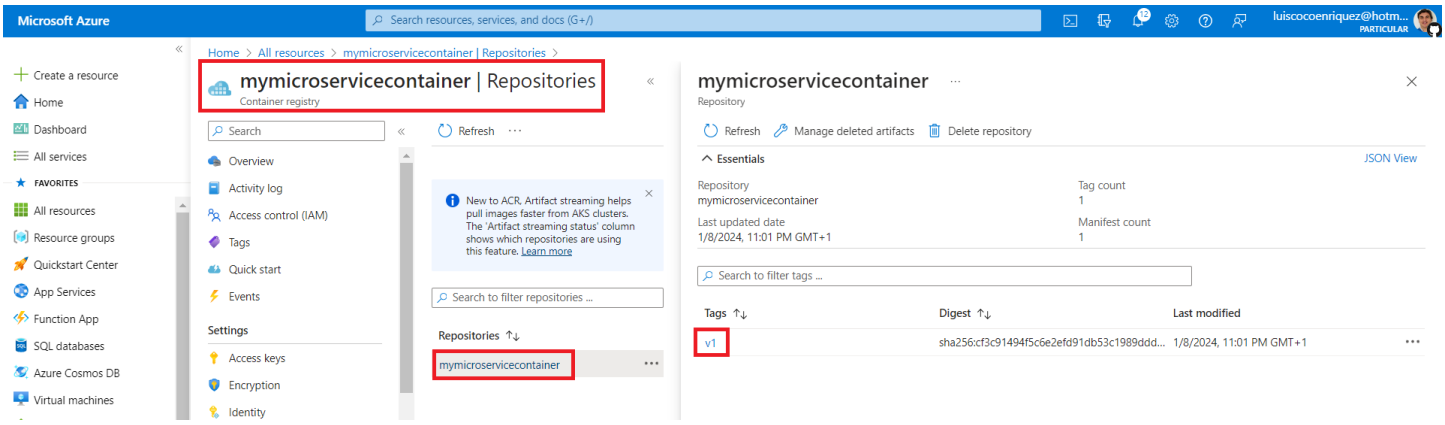
```
docker push mymicroservicecontainer.azurecr.io/mymicroservicecontainer:v1
```

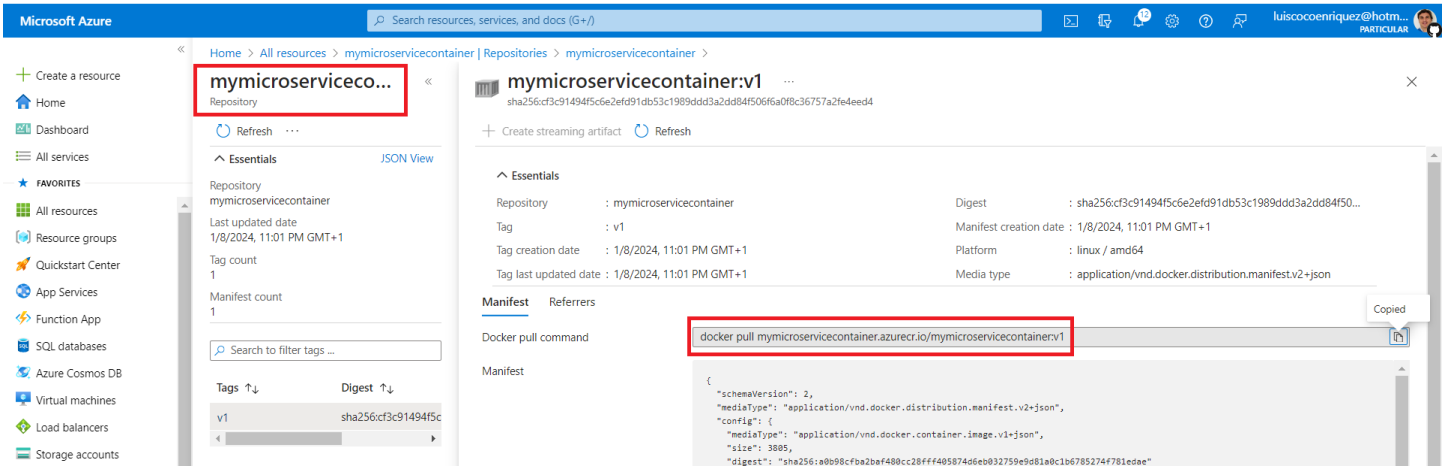
6. Verify the Docker image in Azure ACR

We navigate to the Azure ACR









We first pull the Azure Docker image to our local laptop with this command

```
docker pull mymicroservicecontainer.azurecr.io/mymicroservicecontainer:v1
```

And then we run the docker image

```
docker run -p 80:8080 mymicroservicecontainer.azurecr.io/mymicroservicecontainer:v1
```

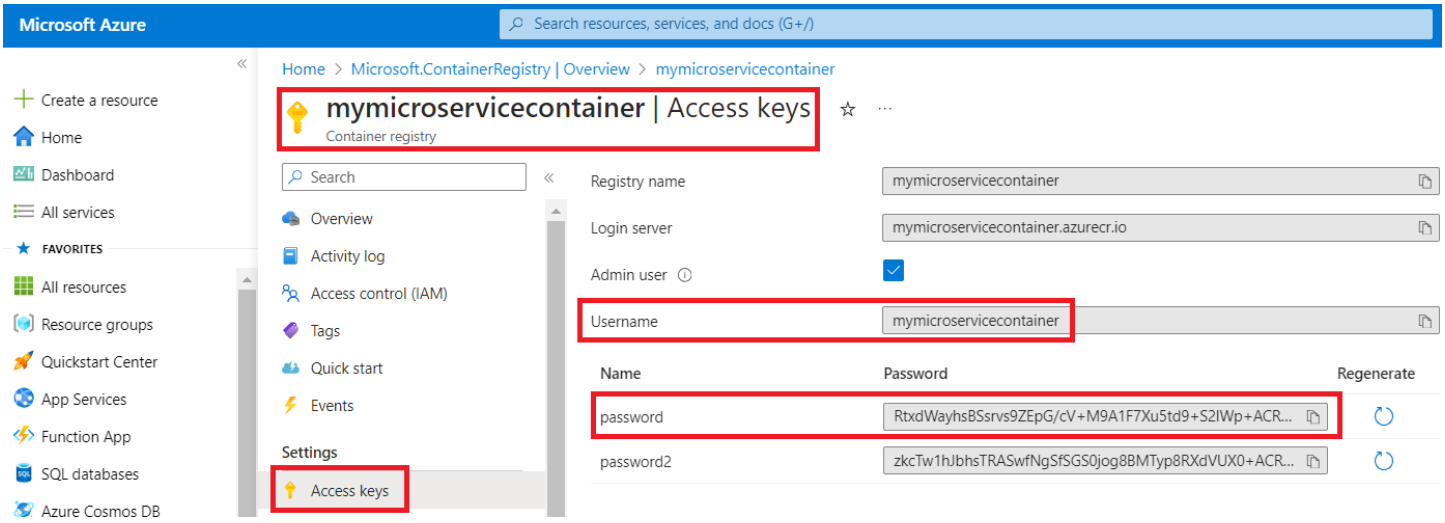
We can access the application endpoint

http://localhost/api/family


```
[
    {
        "id": "1",
        "partitionKey": "pk001",
        "lastName": "Smith",
        "parents": [
            {
                "familyName": "Smith",
                "firstName": "John"
            },
            {
                "familyName": "Smith",
                "firstName": "Jane"
            }
        ],
        "children": [
            {
                "familyName": "Smith",
                "firstName": "Emma",
                "gender": "Female",
                "grade": 5,
                "pets": [
                    {
                        "givenName": "Buddy"
                    }
                ]
            },
            {
                "familyName": "Smith",
                "firstName": "Mike",
                "gender": "Male",
                "grade": 8,
                "pets": []
            }
        ],
        "address": {
            "state": "California",
            "county": "Orange",
            "city": "Irvine"
        },
        "isRegistered": true
    }
]
```

7. Create the Azure Container Instance (ACI)

We copy the ACR username and password:



ACR username: mymicroservicecontainer

ACR password: RtxdWayhsBSsrvs9ZEpG/cV+M9A1F7Xu5td9+S2lWp+ACRDRt6Dk

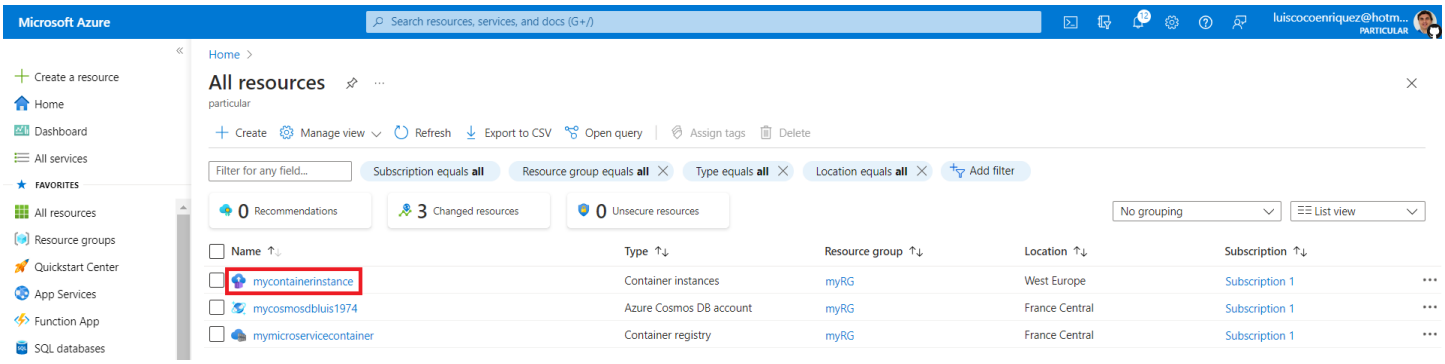
```
az container create --resource-group myRG --name mycontainerinstance --image mymicroservicecon
```

Also we can input the command in multiline

```
az container create --resource-group myRG ^
--name mycontainerinstance ^
--image mymicroservicecontainer.azurecr.io/mymicroservicecontainer:v1 ^
--cpu 1 ^
--memory 1.5 ^
--registry-login-server mymicroservicecontainer.azurecr.io ^
--registry-username mymicroservicecontainer ^
--registry-password RtxdWayhsBSsrvs9ZEpG/cV+M9A1F7Xu5td9+S2lWp+ACRDRt6Dk ^
--dns-name-label mymicroservicedns007 ^
--ports 8080 ^
--location westeurope
```

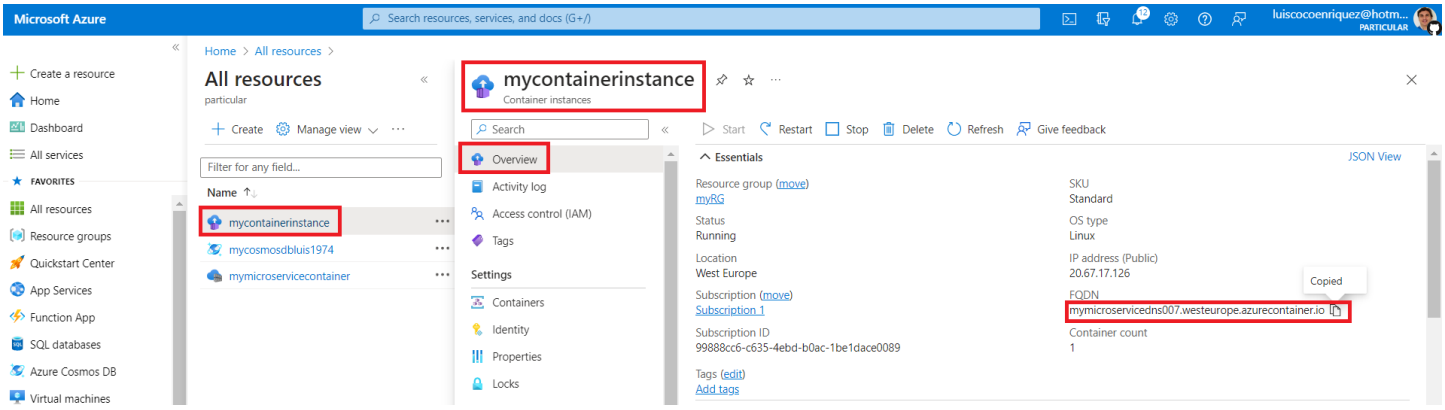
8. Verify the application running in the Azure Container Instance (ACI)

We navigate to the Azure ACI service



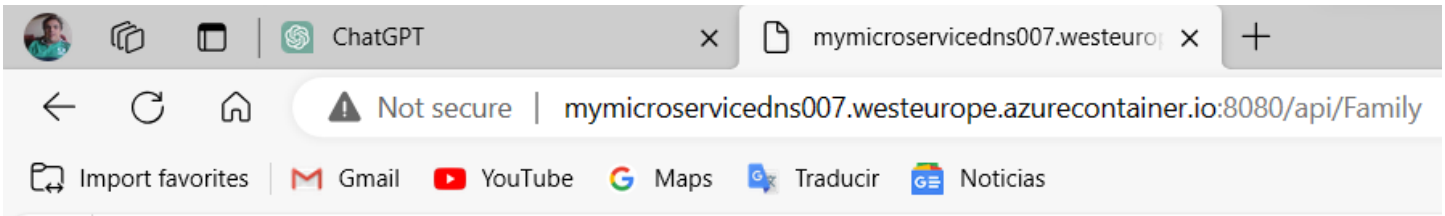
We press in the Azure ACI link

We copy the FQDN



We input the Azure ACI endpoint in the internet web browser

http://mymicroservicedns007.westeurope.azurecontainer.io:8080/api/Family




```
[
    {
        "id": "1",
        "partitionKey": "pk001",
        "lastName": "Smith",
        "parents": [
            {
                "familyName": "Smith",
                "firstName": "John"
            },
            {
                "familyName": "Smith",
                "firstName": "Jane"
            }
        ],
        "children": [
            {
                "familyName": "Smith",
                "firstName": "Emma",
                "gender": "Female",
                "grade": 5,
                "pets": [
                    {
                        "givenName": "Buddy"
                    }
                ]
            },
            {
                "familyName": "Smith",
                "firstName": "Mike",
                "gender": "Male",
                "grade": 8,
                "pets": []
            }
        ],
        "address": {
            "state": "California",
            "county": "Orange",
            "city": "Irvine"
        },
        "isRegistered": true
    }
]
```