

# GithubActions: How to create .NET 7 Web API Docker image and Upload it to AWS ECS Elastic Container Service

---

## 1. Create a .NET 7 Web API application in Visual Studio

---

We run Visual Studio 2022










We create a new project





## Visual Studio 2022

### Abrir recientes

#### Esta semana

	WebAPIdotNET6.sln	13/12/2023 22:17
	C:\AWS CodeCommit\mywebapidonet6repo\WebAPIdotNET6	
	WebAPIdotNET8.sln	13/12/2023 21:55
	C:\AWS CodeCommit\mywebapidonet8repo	
	WebAPIdotnet8.sln	13/12/2023 21:53
	C:\AWS CodeCommit\mywebapidonet8repo\WebAPIdotnet8	
	DeployAzureContainerInstance.sln	12/12/2023 23:31
	C:\...\DeployAzureContainerInstance	
	WebAPI.sln	12/12/2023 21:30
	C:\Azure SDK for .NET how to create an Azure Container Registry\WebAPI	
	HowToCreateAzureContainerRegistry.sln	12/12/2023 21:07
	C:\...\HowToCreateAzureContainerRegistry	
	WebAPI.sln	11/12/2023 19:13
	C:\AWS CodeCommit\mywebapidonet8repo\WebAPI	

### Tareas iniciales





- **Clonar un repositorio**  
Obtiene código desde un repositorio en línea, como GitHub o Azure DevOps.
- **Abrir un proyecto o una solución**  
Abre un archivo .sln o proyecto de Visual Studio local.
- **Abrir una carpeta local**  
Navegar y editar el código en cualquier carpeta
- **Crear un proyecto**  
Elija una plantilla de proyecto con la técnica scaffolding de código para comenzar.

[Continuar sin código →](#)


We select the projec template "ASP.NET Core Web API"

## Crear un proyecto


### Plantillas de proyecto recientes

-  ASP.NET Core Web API C#
-  Aplicación de consola C#
-  Azure Functions C#
-  Aplicación .NET MAUI Blazor C#

Todos los lenguajes ▼ Todas las plataformas ▼ Todos los tipos de proye... ▼


**Una plantilla de proyecto para crear una aplicación ASP.NET Core con contenido de Razor Pages de ASP.NET Core de ejemplo.**

C# Linux macOS Windows Nube Servicio Web

**ASP.NET Core Web API**  
Una plantilla de proyecto para crear una API web RESTful utilizando controladores ASP.NET Core o API mínimas, con soporte opcional para OpenAPI y autenticación.


C# Linux macOS Windows API Nube Servicio Web

Web API


**API web ASP.NET Core (native AOT)**  
Una plantilla de proyecto para crear una API web RESTful utilizando las API mínimas de ASP.NET Core publicadas como native AOT.

C# Linux macOS Windows API Nube Servicio Web

Web API

**Biblioteca de clases**  
Proyecto para crear una biblioteca de clases para .NET o .NET Standard

C# Android Linux macOS Windows Biblioteca

**ASP.NET Core vacío**  
Una plantilla de proyecto vacía para crear una aplicación ASP.NET Core. Esta plantilla no incluye ningún contenido.

[Atrás](#) [Siguiente](#)

We set the project name and location

## Configure su nuevo proyecto

ASP.NET Core Web API C# Linux macOS Windows API Nube Servicio Web Web API

Nombre del proyecto

WebApplication1

Ubicación

C:\AWS ECR dotNET7 WebAPI\

Nombre de la solución ⓘ

WebApplication1

☒ Colocar la solución y el proyecto en el mismo directorio

Proyecto se creará en "C:\AWS ECR dotNET7 WebAPI\WebApplication1\"

Atrás Siguiente

We select the project main options: .NET 7 framework, enable Docker, configure HTTPS, use OpenAPI and user controllers

## Información adicional

ASP.NET Core Web API C# Linux macOS Windows API Nube Servicio Web Web API

Framework ⓘ

.NET 7.0 (Soporte técnico de términos estándar)

Authentication de campo ⓘ

Ninguno

☒ Configurar para HTTPS ⓘ

☒ Habilitar Docker ⓘ

Sistema operativo de Docker ⓘ

Linux

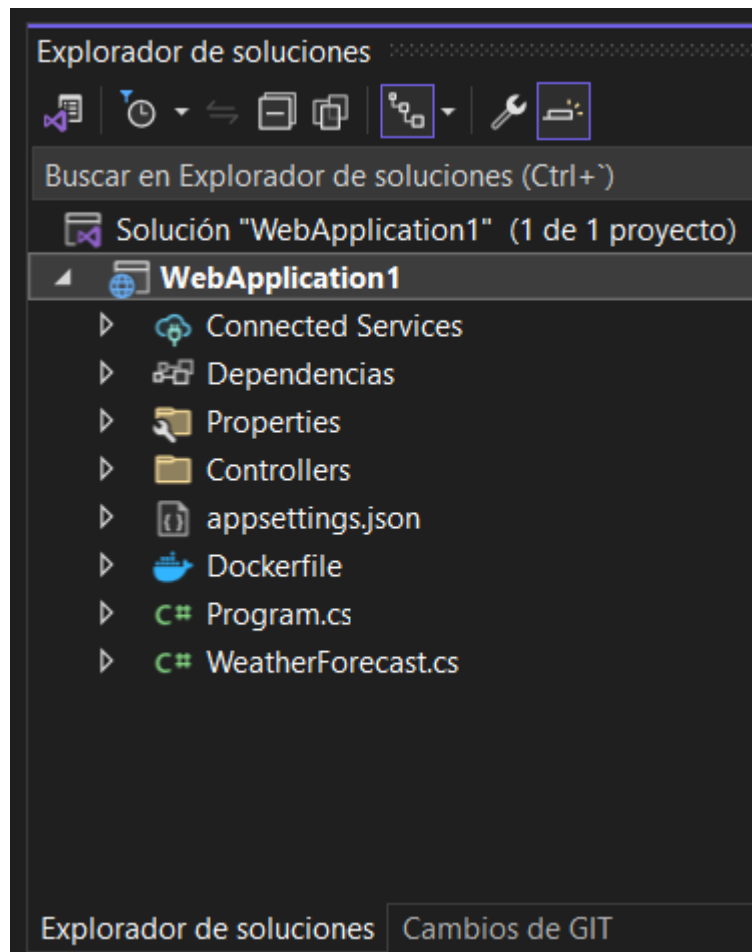
☒ Habilitar compatibilidad con OpenAPI ⓘ

☐ No usar instrucciones de nivel superior ⓘ

☒ Utilizar controladores ⓘ

Atrás Crear

## 2. Upload your .NET 7 Web API to a Github repo



## Crear un repositorio GIT

Enviar cambios a un nuevo repositorio remoto

GitHub

Azure DevOps

Otros

Repositorio remoto existente

Solo local

### Inicializar un repositorio GIT local

Ruta de acceso local [?](#) C:\AWS ECR dotNET7 WebAPI\WebApplication1

Plantilla .gitignore [?](#) Default (VisualStudio)

Plantilla de licencia [?](#) Ninguno

☒ Agregar un README.md [?](#)

### Crear un nuevo repositorio de GitHub

Cuenta [?](#) luisco (GitHub)

Propietario [?](#) luisco

Nombre del repositorio [?](#) GithubActions\_Create\_DockerImage\_Upload\_to\_AWS\_ECR\_dotNET7WebAP

Descripción [?](#) Escriba la descripción del repositorio de GitHub <Opcional>

☒ Repositorio privado [?](#)

[Vuelva a escribir sus credenciales](#)

[El nuevo repositorio se creará como GithubActions\\_Create\\_DockerImage\\_Upload\\_to\\_AWS\\_ECR\\_dotNET7WebAP.](#)

[Crear y enviar los cambios](#) [Cancelar](#)

### 3. Create a AWS ECR Public repo for storing the Docker image

Navigate to the AWS ECR service and create a new Public repo

eu-west-3.console.aws.amazon.com/ecr/public-registry/repositories?region=eu-west-3

Amazon Elastic Container Registry

Public registry

Repositories

Settings

ECR public gallery

Amazon ECS

Amazon EKS

Getting started

Documentation

### Public repositories

Repositories

Filter status

Repository name

URI


Created at

No repositories

No repositories were found

[Create repository](#)

We set the repository name, and the container compatible operating system and architecture

 **Services**  [Alt+S]

## Detail

**Repository name** [Info](#)

A namespace can be included with your repository name (e.g. namespace/repo-name).


public.ecr.aws/x6y4g2f4/ **dotnet8webapirepo**

17 out of 205 characters maximum (2 minimum). The name must start with a letter and can only contain lowercase letters, numbers, hyphens, underscores, periods and forward slashes.

**i** A default alias is associated with your public registry once your first public repository is created. The registry alias is displayed as a prefix to the repository name in the repository URI. A custom alias can be requested on the Registry settings page.

**Repository logo - optional** [Info](#)

Choose a local image file to use as the repository logo.

 **Upload file** **Reset**

The supported file format is PNG. The supported image dimensions for both height and width should be a minimum of 60 pixels and a maximum of 2048 pixels. The maximum file size is 500 KB.

**Short description - optional**

The short description is displayed in search results and on the repository detail page.

0 out of 255 characters maximum.

**Content types - optional** [Info](#)

Select the operating systems and system architectures that are compatible with the images in your repository.


**Operating systems**

- ☒ Linux
- ☒ Windows

**Architectures**

- ☒ ARM
- ☒ ARM 64
- ☒ x86
- ☒ x86-64

We press the **Create repository** button

 Services  [Alt+S]

### About - optional [Info](#) [View example](#)

Provide a detailed description of the repository. Identify what is included in the repository, any licensing details, or other relevant information.

*Describe this repository*

0 out of 10,240 characters maximum. Use GitHub Flavored Markdown format for the text. [Learn more](#)

Preview

### Usage - optional [Info](#) [View example](#)

Provide detailed information about how to use the images in the repository. This provides context, support information, and additional usage details for users of the repository.

*Usage information*


0 out of 10,240 characters maximum. Use GitHub Flavored Markdown format for the text. [Learn more](#)

Preview

Cancel

Create repository

See the new repo in the list

 Services  [Alt+S] Parts luisnewuser @ adminluiscooreniquez

Amazon Elastic Container Registry

▼ Private registry

Repositories

Settings

▼ Public registry

[Repositories](#)

Settings

ECR public gallery

Amazon ECS

Amazon EKS

Getting started

Documentation

Created public repository

dotnet8webapirepo has been successfully created in public registry

[Amazon ECR](#) > [Public Registry](#) > [Repositories](#)

Public repositories

Repositories (1)

dotnet8webapirepo

public.ecr.aws/x6y4g2f4/dotnet8webapirepo

December 15, 2023, 18:53:30 (UTC+01)

View push commands

Delete

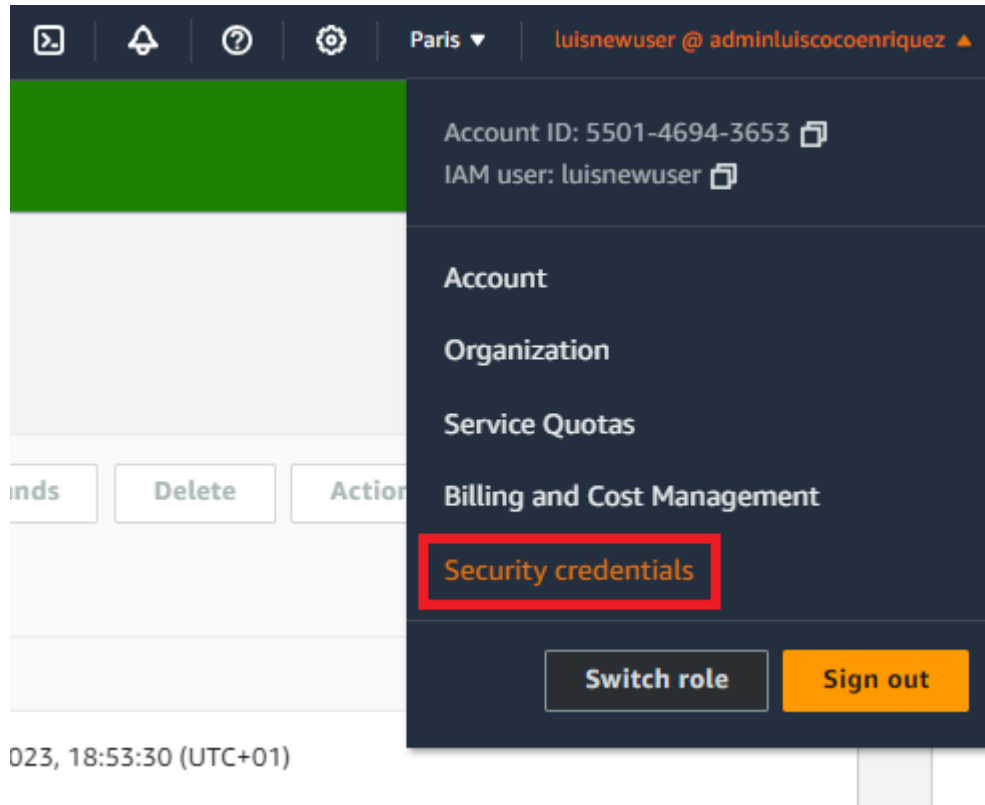
Actions

Create repository

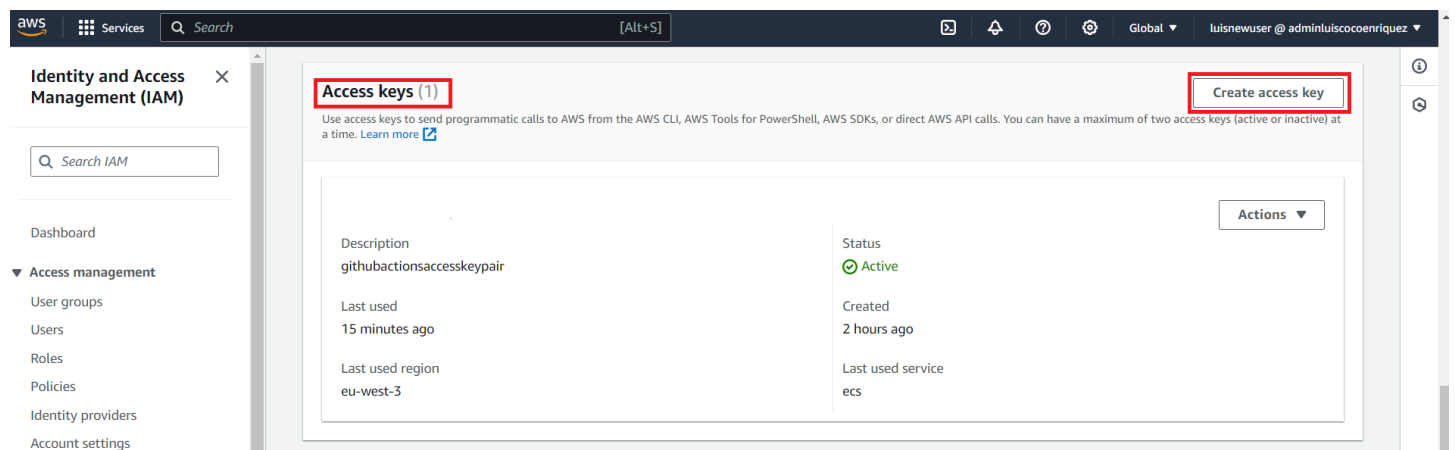
## 4. Create the Github Secrets to store the AWS\_ACCESS\_KEY\_ID and AWS\_SECRET\_ACCESS\_KEY

We start this section in **AWS** creating an access key and a secret key

We select the **Security Credentials** option in the top right menu

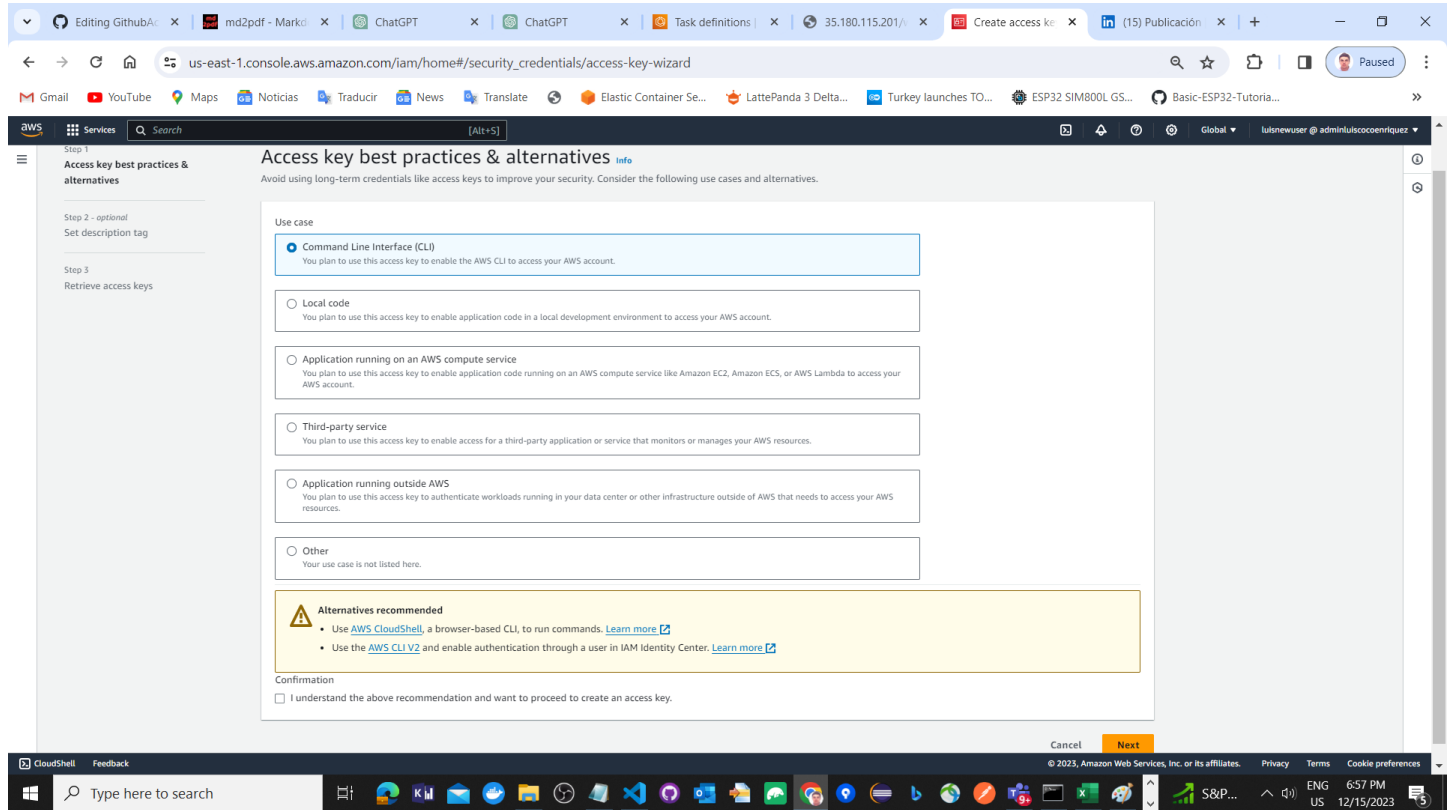


We click on the **Create access key** button

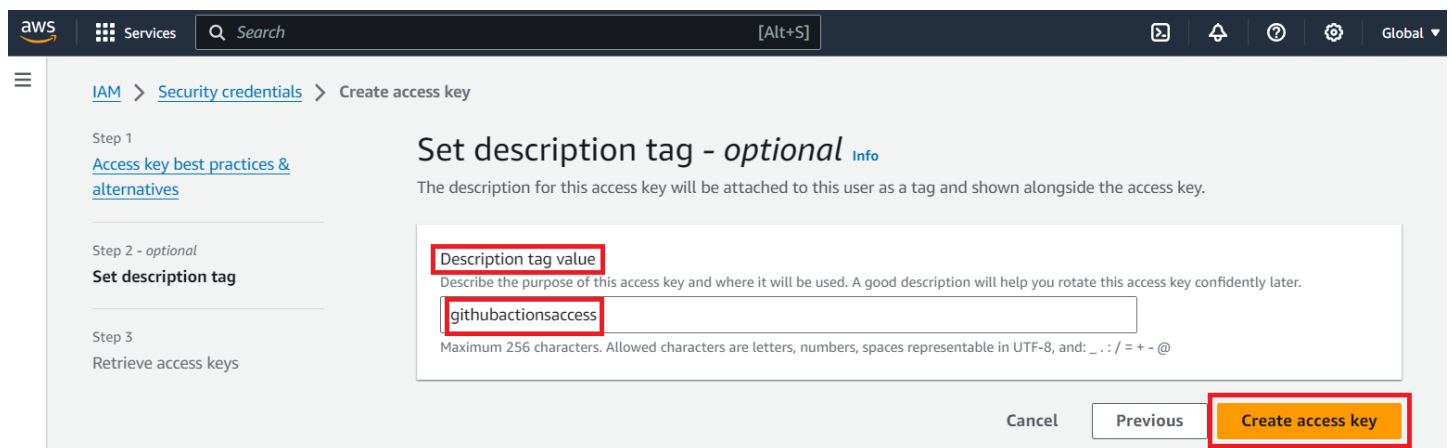


Select Use case **Command Line Interface CLI**, check "I understand the above recommendation and want to proceed to create an access key." and press Next button





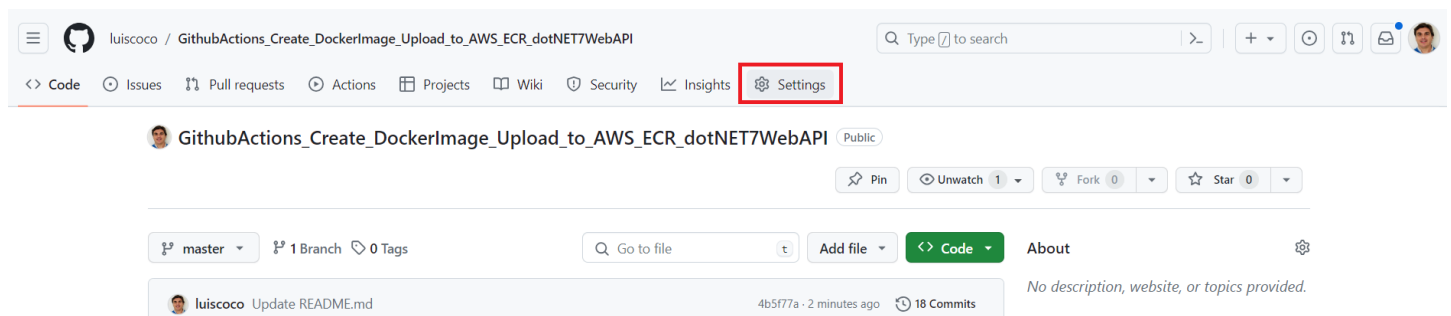
We input the description tag and press the **Create access key** button



We click on the **Download .csv** file and an Excel file will be downloaded to our laptop with the access key and secret key

Go to your **Github repo** and create two secrets for storing the access key and secret key

Press the **Settings** button



We select **Secrets and variables** and we store **AWS\_ACCESS\_KEY\_ID** and **AWS\_SECRET\_ACCESS\_KEY** in two github repository secrets

github.com/luisco/GithubActions\_Create\_DockerImage\_Upload\_to\_AWS\_ECR\_dotNET7WebAPI/settings

General

Repository name: GithubActions\_Create\_DockerImage\_ [Rename]

☐ Template repository

☐ Require contributors to sign off on web-based commits

Default branch: master [Edit]

Social preview

github.com/luisco/GithubActions\_Create\_DockerImage\_Upload\_to\_AWS\_ECR\_dotNET7WebAPI/settings/secrets/actions

Environment secrets

This repository has no environment secrets. [Manage environment secrets]

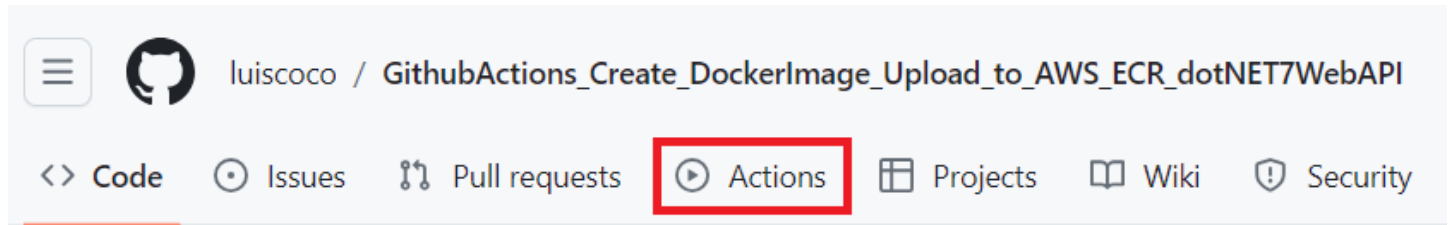
Repository secrets

Name	Last updated
AWS_ACCESS_KEY_ID	3 days ago
AWS_SECRET_ACCESS_KEY	3 days ago

[New repository secret]

## 5. Create the Github actions workflow

In the application Github repo click on the **Actions** button



We input the **main.yml** file for executing the github action workflow to create a .NET 7 Web API Docker image and upload it to AWS ECR

```
name: Build and Push Docker Image

on:
  push:
    branches: [ master ]

env:
  ECR_REGISTRY: public.ecr.aws/x6y4g2f4 # Your AWS ECR Registry
  IMAGE_NAME: dotnet8webapirepo # Replace with your image name
  AWS_REGION: us-east-1 # Replace with your AWS region

jobs:
  build_and_push:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2

      - name: Set up Docker Buildx
        uses: docker/setup-buildx-action@v1

      - name: Set up AWS CLI
        uses: aws-actions/configure-aws-credentials@v1
        with:
          aws-access-key-id: ${ secrets.AWS_ACCESS_KEY_ID }
          aws-secret-access-key: ${ secrets.AWS_SECRET_ACCESS_KEY }
          aws-region: ${ env.AWS_REGION }

      - name: Login to Amazon ECR
        run: |
          aws ecr-public get-login-password --region ${ env.AWS_REGION } | docker login --user
# New step to delete existing images
      - name: Delete existing images in ECR repository
        run: |
          aws ecr-public describe-images --repository-name ${ env.IMAGE_NAME } --region ${ en

      - name: Build, tag, and push image to Amazon ECR
        run: |
          docker build -t ${ env.ECR_REGISTRY }/${ env.IMAGE_NAME }:latest .
          docker push ${ env.ECR_REGISTRY }/${ env.IMAGE_NAME }:latest
```

We press "Commit changes..." button

## 6. How deploy to AWS ECS the Docker image stored in my ECR repo

---

We create a ECS Cluster:

```
aws ecs create-cluster --cluster-name myCluster
```

Create a new file task-definition.json and place it in a local folder: C:/AWS Task Definition/task-definition.json

This is the task-definition.json file source code

```
{
  "family": "myDotnetApp",
  "executionRoleArn": "arn:aws:iam::550146943653:role/ecsTaskExecutionRole",
  "networkMode": "awsvpc",
  "containerDefinitions": [
    {
      "name": "dotnet8webapi",
      "image": "public.ecr.aws/x6y4g2f4/dotnet8webapirepo:latest",
      "cpu": 1024,
      "memory": 3072,
      "essential": true,
      "portMappings": [
        {
          "containerPort": 80,
          "hostPort": 80
        }
      ],
      "environment": [
        {
          "name": "ASPNETCORE_ENVIRONMENT",
          "value": "Development"
        }
      ]
    }
  ],
  "requiresCompatibilities": [
    "FARGATE"
  ],
  "cpu": "1024",
  "memory": "3072"
}
```

We create a new Task Definition:

```
aws ecs register-task-definition --cli-input-json file://"C:/AWS Task Definition/task-definiti
```

We get my AWS Account default **VPC** name:

```
aws ec2 describe-vpcs ^
--filters "Name=isDefault,Values=true" ^
--query "Vpcs[].VpcId" ^
--output text
```

For example the VPC name output is: **vpc-07d51d92f73354c0b**

With that VPC name we can retrieve the attached **Subnets** names:

```
aws ec2 describe-subnets ^
--filters "Name=vpc-id,Values=vpc-07d51d92f73354c0b" ^
--query "Subnets[].{ID:SubnetId,Name:Tags[?Key=='Name']|[0].Value}"
```

For example the Subnet output is: **subnet-0df35048d0d30e90f**

We also get the **Security Group** name with this command:

```
aws ec2 describe-security-groups ^
--filters "Name=vpc-id,Values=vpc-07d51d92f73354c0b" "Name=group-name,Values=default" ^
--query "SecurityGroups[].{ID:GroupId,Name:GroupName}" ^
--output text
```

For example the Security Group output is: **sg-051b9197846af4fe0**

We create a new **Service** with the "aws ecs create-service" command

Do not forget to set the **subnetId** and the **securityGroupId**

```
aws ecs create-service ^
--cluster myCluster ^
--service-name myDotnetService ^
--task-definition myDotnetApp ^
--launch-type FARGATE ^
--desired-count 1 ^
--network-configuration "awsvpcConfiguration={subnets=[subnet-0df35048d0d30e90f],securityGro
```

For accessing the endpoint click on the Public IP address and add the controller name

<http://IPAddress/controlername>