# Mi primer API con .NetCore

## Requerimientos

- Estar en un sistema *Linux* (en una distribución basada en *Debian*, *Ubuntu* o *Fedora*).
- Tener *.NetCore* instalado en el sistema, en su última version LTS (al momento la elaboración de esta guia es la versión 3.1).
- Tener *Git* instalado en el sistema.
- Un editor de texto o IDE.
- *Postman* o cualquier herramienta para probar APIs
- Tener instalado *Docker*.

  Para esta guia se utilizara Ubuntu con VSCode

## Configurando entorno de trabajo

Primero necesitamos crear nuestro directorio donde estará nuestro proyecto. Una vez dentro del nuevo directorio, se debe crear repositorio correspondiente:

```
mkdir MiPrimerApi
cd MiPrimerApi
git init
```

Luego procederemos a crear nuestro archivo *.gitignore* y lo llenaremos con las reglas necesarias. Además, si así se desea, se puede crear el archivo *README*.

> Sugerencia: Para esta guía se utilizaran las reglas especificadas en este enlace, pero el/la estudiante es libre ocupar las reglas que el considere.

```
touch .gitignore
touch README.md
```

Ahora haremos commit.

> El/la estudiante es libre de escoger si hacer commit por archivo individual o agregar los dos archivos en un mismo commit.

```
git add README.md
git commit -m "Agregado README"
git add .gitignore
git commit -m "Agregado archivo .gitignore"
```

## Creando el proyecto

Comprobamos que tengamos la versión 3.1 de *.NetCore* instalada y procedemos a crear nuestro proyecto.

```
dotnet --version
dotnet new webapi
```

Hacemos commit para declarar que hemos creado nuestro proyecto.

```
git add .
git commit -m "Creacion del proyecto"
```

> A partir de aquí el/la estudiante es libre de como manejar sus commits

Corremos nuestro proyecto recien creado, este debería correr en la siguiente dirección https://localhost:5001/WeatherForecast/

```
dotnet run
```

> Si se quiere modificar el puerto en donde se se ejecutará la aplicacion al ocupar el comando **dotnet run**, se debe modificar el archivo *MiPrimeraApi/Properties/launchSettings.json*.

## Trabajando con el proyecto

En la raíz de nuestro directorio se hará un directorio con el nombre *Models* y dentro de este se creará la clase *Articulo.cs*, esta clase contiene lo siguiente:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;

namespace MiPrimeraApi.Models
{
    public class Articulo
    {
        public int Id { set; get; }
        public string Nombre { set; get; }
        public string Descripcion { set; get; }
        public double Precio { set; get; }
        public DateTime FechaRegistro { set; get; }
    }
}
```

Una vez se creó el modelo, en la carpeta de *Controllers* agregar el archivo *ArticuloController*. En este archivo agregar lo siguiente

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using MiPrimeraApi.Models;

namespace MiPrimeraApi.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class ArticuloController : ControllerBase
    {
        List<Articulo> articulos { set; get; }

        public ArticuloController()
        {
            articulos = new List<Articulo>()
            {
                new Articulo { Id = 1, Nombre = "Laptop", Descripcion = "Laptop HP", Pre
                new Articulo { Id = 2, Nombre = "Impresora", Descripcion = "Impresora Ep
                new Articulo { Id = 3, Nombre = "Monito", Descripcion = "Monitor ASUS",
                new Articulo { Id = 4, Nombre = "Cable USB", Descripcion = "Cable USB Ge
            };
        }

        // GET api/articulo
        [HttpGet]
        [Route("")]
        public IActionResult Obtener()
        {
            return Ok(articulos);
        }
    }
}
```

Después de eso, ejecutar el proyecto y probarlo, ingresando al siguiente enlace: https://localhost:5001/api/articulo, esto nos debería dar algo similiar a lo siguiente:

```
[
    {
        "id":1,
        "nombre":"Laptop",
        "descripcion":"Laptop HP",
        "precio":15000,
```

```
        "fechaRegistro":"2020-03-14T00:29:47.2435536-06:00"
    },
    {
        "id":2,
        "nombre":"Impresora",
        "descripcion":"Impresora Epson",
        "precio":8700,
        "fechaRegistro":"2020-03-14T00:29:47.2436054-06:00"
    },
    {
        "id":3,
        "nombre":"Monito",
        "descripcion":"Monitor ASUS",
        "precio":1600,
        "fechaRegistro":"2020-03-14T00:29:47.2436069-06:00"
    },
    {
        "id":4,
        "nombre":"Cable USB",
        "descripcion":"Cable USB Generico",
        "precio":193,
        "fechaRegistro":"2020-03-14T00:29:47.2436073-06:00"
    }
]
```


## Creando nuestra primer imagen

En el proyecto, crear un archivo *Dockerfile* y agregar las siguiente lineas

```
# Esto va a crear la imagen del SDK de Microsoft

FROM mcr.microsoft.com/dotnet/core/sdk:3.1 AS build-env
WORKDIR /app

# Esto va a copiar el archivo csproj e instala/restaura las dependencias (Via gestor de

COPY *.csproj ./
RUN dotnet restore

# Esto copia los archivos del proyecto y crea el lanzamiento (release)

COPY . ./
RUN dotnet publish -c Release -o out
# Si este último comando se ejecuta directamente desde la consola, creará una carpeta co
```

```
# Genera nuestra imagen

FROM mcr.microsoft.com/dotnet/core/aspnet:3.1
WORKDIR /app
EXPOSE 80
COPY --from=build-env /app/out .
ENTRYPOINT [ "dotnet", "MiPrimeraApi.dll" ]
```

Despues de eso, crearemos nuestro archivo *.dockerignore*. El cual, similar al archivo *.gitignore*, le dirá a Docker que ignore esos archivos. El contenido de este archivo es el siguiente.

```
bin/
obj/
```

Una vez tengamos nuestros archivos, ejecutamos el siguiente comando:

```
sudo docker build -t <TuDockerHubID>/<NombreDelProyecto>:<Version> .
```

*Nota*: El *TuDockerHubID* y *NombreDelProyecto* deben ir en minusculas.

En mi caso ejecutaría el siguiente comando

```
sudo docker build -t saulenriquemr/miprimeraapi:1.0 .
```

*Nota*: Si no se especifica una versión, *Docker* por defecto asignará el tag *latest*, y si no se pone DockerHubID ni nombre del proyecto, *Docker* le pondra un ID único.

Ahora, si ejecutamos el siguiente comando, podremos ver que ya existe nuestra imagen

```
sudo docker images
```

## Corriendo nuestra imagen

Ya hemos construido nuestra imagen de *Docker*, ahora el siguiente paso es correr la imagen.

Ejecutar el siguiente comando:

```
sudo docker run -p <PuertoDeNuestraMaquinaDondeSeEjecutara>:<PuertoExpuestoEnNuestroDock
```

En mi caso quedaría algo así:

```
sudo docker run -p 8080:80 saulenriquemr/miprimeraapi:1.0
```

Ahora si accedemos a la siguiente direccion http://localhost:8080/api/articulo, nos debería de dar el *json* que nos regreso la primera vez.

## Publicando mi imagen en DockerHub

Primero debemos iniciar sesion en *Docker* con el siguiente comando

```
sudo docker login
```

Luego debemos ejecutar el siguiente comando:

```
sudo docker push <TuDockerHubID>/<NombreDelProyecto>:<VersionSiSeEspecifico>
```

En mi caso sería así:

```
sudo docker push saulenriquemr/miprimeraapi:1.0
```

Una vez terminado, se puede revisar el repositorio en *DockerHub*.