

# TP9 - Unit Test

## 5- Desarrollo de Pruebas Unitarias sobre una aplicación de consola.

Preparamos el entorno

- Desde línea de comandos clonamos el proyecto MiSimpleApp, entramos a la carpeta y abrimos VS.Code

```
MINGW64:~/go/src/github.com/belenaguilarv/IngenieriaDeSoftware3/TP9 - Unit Test/MiSimpleApp
BELU@belenaguilarv MINGW64 ~/go/src/github.com/belenaguilarv/IngenieriaDeSoftware3/TP9 - Unit Test (main)
$ git clone https://github.com/ingsoft3ucc/MiSimpleApp.git
Cloning into 'MiSimpleApp'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.

BELU@belenaguilarv MINGW64 ~/go/src/github.com/belenaguilarv/IngenieriaDeSoftware3/TP9 - Unit Test (main)
$ cd MiSimpleApp

BELU@belenaguilarv MINGW64 ~/go/src/github.com/belenaguilarv/IngenieriaDeSoftware3/TP9 - Unit Test/MiSimpleApp (main)
$ code .

BELU@belenaguilarv MINGW64 ~/go/src/github.com/belenaguilarv/IngenieriaDeSoftware3/TP9 - Unit Test/MiSimpleApp (main)
```

- Revisamos el código
- Cerramos VS.Code
- Nos movemos una carpeta hacia arriba y creamos un nuevo proyecto de pruebas unitarias con NUnit:

```
BELU@belenaguilarv MINGW64 ~/go/src/github.com/belenaguilarv/IngenieriaDeSoftware3/TP9 - Unit Test (main)
$ dotnet new nunit -n MiSimpleAppTests
La plantilla "NUnit 3 Test Project" se creó correctamente.

Procesando acciones posteriores a la creación...
Restaurando C:\Users\BELU\go\src\github.com\belenaguilarv\IngenieriaDeSoftware3\TP9 - Unit Test\MiSimpleAppTests\MiSimpleAppTests.csproj:
  Determinando los proyectos que se van a restaurar...
  Se ha restaurado C:\Users\BELU\go\src\github.com\belenaguilarv\IngenieriaDeSoftware3\TP9 - Unit Test\MiSimpleAppTests\MiSimpleAppTests.csproj (en 21,87 sec).
Restauración realizada correctamente.
```

- Entramos a la carpeta del nuevo proyecto y agregamos los paquetes NUnit y NUnit.ConsoleRunner. Luego le agregamos al proyecto de pruebas una referencia

al proyecto que vamos a probar. Nos movemos una carpeta hacia arriba y lo vemos en VS.Code

## NUnit

```
BEU@belenaguilarv MINGW64 ~/go/src/github.com/belenaguilarv/IngenieriaDeSoftware3/TP9 - Unit Test (main)
$ cd MiSimpleAppTests

BEU@belenaguilarv MINGW64 ~/go/src/github.com/belenaguilarv/IngenieriaDeSoftware3/TP9 - Unit Test/MiSimpleAppTests (main)
$ dotnet add package NUnit
Determinando los proyectos que se van a restaurar...
Writing C:\Users\BEU\AppData\Local\Temp\tmp4A7.tmp
info : X.509 certificate chain validation will use the default trust store selected by .NET for code signing.
info : X.509 certificate chain validation will use the default trust store selected by .NET for timestamping.
info : Agregando PackageReference para el paquete "NUnit" al proyecto "C:\Users\BEU\go\src\github.com\belenaguilarv\IngenieriaDeSoftware3\TP9 - Unit Test\MiSimpleAppTests\MiSimpleAppTests.csproj".
info : GET https://api.nuget.org/v3/registration5-gz-semver2/nunit/index.json
info : OK https://api.nuget.org/v3/registration5-gz-semver2/nunit/index.json 918 ms
info : Restaurando paquetes para C:\Users\BEU\go\src\github.com\belenaguilarv\IngenieriaDeSoftware3\TP9 - Unit Test\MiSimpleAppTests\MiSimpleAppTests.csproj...
info : El paquete "NUnit" es compatible con todos los marcos de trabajo especificados del proyecto "C:\Users\BEU\go\src\github.com\belenaguilarv\IngenieriaDeSoftware3\TP9 - Unit Test\MiSimpleAppTests\MiSimpleAppTests.csproj".
info : Se actualizó PackageReference para la versión "3.13.3" del paquete "NUnit" en el archivo "C:\Users\BEU\go\src\github.com\belenaguilarv\IngenieriaDeSoftware3\TP9 - Unit Test\MiSimpleAppTests\MiSimpleAppTests.csproj".
info : El archivo de recursos no ha cambiado, así que se omitirá su escritura. Ruta de acceso: C:\Users\BEU\go\src\github.com\belenaguilarv\IngenieriaDeSoftware3\TP9 - Unit Test\MiSimpleAppTests\obj\project.assets.json
log : Se ha restaurado C:\Users\BEU\go\src\github.com\belenaguilarv\IngenieriaDeSoftware3\TP9 - Unit Test\MiSimpleAppTests\MiSimpleAppTests.csproj (en 200 ms).
```

## NUnit.ConsoleRunner

```
BEU@belenaguilarv MINGW64 ~/go/src/github.com/belenaguilarv/IngenieriaDeSoftware3/TP9 - Unit Test/MiSimpleAppTests (main)
$ dotnet add package NUnit.ConsoleRunner
Determinando los proyectos que se van a restaurar...
Writing C:\Users\BEU\AppData\Local\Temp\tmp15AF.tmp
info : X.509 certificate chain validation will use the default trust store selected by .NET for code signing.
info : X.509 certificate chain validation will use the default trust store selected by .NET for timestamping.
info : Agregando PackageReference para el paquete "NUnit.ConsoleRunner" al proyecto "C:\Users\BEU\go\src\github.com\belenaguilarv\IngenieriaDeSoftware3\TP9 - Unit Test\MiSimpleAppTests\MiSimpleAppTests.csproj".
info : GET https://api.nuget.org/v3/registration5-gz-semver2/nunit.consolerunner/index.json
info : OK https://api.nuget.org/v3/registration5-gz-semver2/nunit.consolerunner/index.json 925 ms
info : Restaurando paquetes para C:\Users\BEU\go\src\github.com\belenaguilarv\IngenieriaDeSoftware3\TP9 - Unit Test\MiSimpleAppTests\MiSimpleAppTests.csproj...
info : GET https://api.nuget.org/v3-flatcontainer/nunit.consolerunner/index.json
info : OK https://api.nuget.org/v3-flatcontainer/nunit.consolerunner/index.json 836 ms
info : GET https://api.nuget.org/v3-flatcontainer/nunit.consolerunner/3.16.3/nunit.consolerunner.3.16.3.nupkg
info : OK https://api.nuget.org/v3-flatcontainer/nunit.consolerunner/3.16.3/nunit.consolerunner.3.16.3.nupkg 28 ms
info : Se instaló NUnit.ConsoleRunner 3.16.3 de https://api.nuget.org/v3/index.json con el hash de contenido bhc4f0bVdS70xLau1Pu2dMuUWK6hLj5fWLWmUnvbyEA9w41sZNe6Xb3vPkcp+50rIqggmtcRbhX+o5hoz62Jg==.
info : El paquete "NUnit.ConsoleRunner" es compatible con todos los marcos de trabajo especificados del proyecto "C:\Users\BEU\go\src\github.com\belenaguilarv\IngenieriaDeSoftware3\TP9 - Unit Test\MiSimpleAppTests\MiSimpleAppTests.csproj".
info : Se agregó PackageReference para la versión "3.16.3" del paquete "NUnit.ConsoleRunner" al archivo "C:\Users\BEU\go\src\github.com\belenaguilarv\IngenieriaDeSoftware3\TP9 - Unit Test\MiSimpleAppTests\MiSimpleAppTests.csproj".
info : Generación de archivo MSBuild C:\Users\BEU\go\src\github.com\belenaguilarv\IngenieriaDeSoftware3\TP9 - Unit Test\MiSimpleAppTests\obj\MiSimpleAppTests.csproj.nuget.g.props.
info : Escribiendo el archivo de recursos en el disco. Ruta de acceso: C:\Users\BEU\go\src\github.com\belenaguilarv\IngenieriaDeSoftware3\TP9 - Unit Test\MiSimpleAppTests\obj\project.assets.json
log : Se ha restaurado C:\Users\BEU\go\src\github.com\belenaguilarv\IngenieriaDeSoftware3\TP9 - Unit Test\MiSimpleAppTests\MiSimpleAppTests.csproj (en 2,88 sec).
```

```

BELU@belenaguilarv MINGW64 ~/go/src/github.com/belenaguilarv/IngenieriaDeSoftware3/TP9 - Unit Test/MiSimpleAppTests (main)
$ dotnet add reference ../MiSimpleApp/MiSimpleApp.csproj
Se ha agregado la referencia "..\MiSimpleApp\MiSimpleApp.csproj" al proyecto.

BELU@belenaguilarv MINGW64 ~/go/src/github.com/belenaguilarv/IngenieriaDeSoftware3/TP9 - Unit Test/MiSimpleAppTests (main)
$ cd ..

BELU@belenaguilarv MINGW64 ~/go/src/github.com/belenaguilarv/IngenieriaDeSoftware3/TP9 - Unit Test (main)
$ code .

BELU@belenaguilarv MINGW64 ~/go/src/github.com/belenaguilarv/IngenieriaDeSoftware3/TP9 - Unit Test (main)
$

```



Siempre hay 2 proyectos: El objeto de prueba y El de prueba

Creamos nuestros Tests:

- Dado que el método **CanBeCancelledBy** de la clase **Reservation** tiene una lógica con 3 caminos posibles, debemos probar esos 3 caminos:
- Modificamos nuestro archivo **UnitTest1.cs** del proyecto **MiSimpleAppTests**

```

1 namespace MiSimpleAppTests;
2
3 [TestFixture]
4 public class Tests
5 {
6     [SetUp]
7     public void Setup() ...
8
9
10
11     [Test]
12     public void CanBeCancelledBy_AdminCancelling_ReturnsTrue() ...
13
14
15
16     [Test]
17     public void CanBeCancelledBy_SameUserCancelling_ReturnsTrue() ...
18
19
20
21     [Test]
22     public void CanBeCancelledBy_AnotherUserCancelling_ReturnsFalse() ...
23
24
25

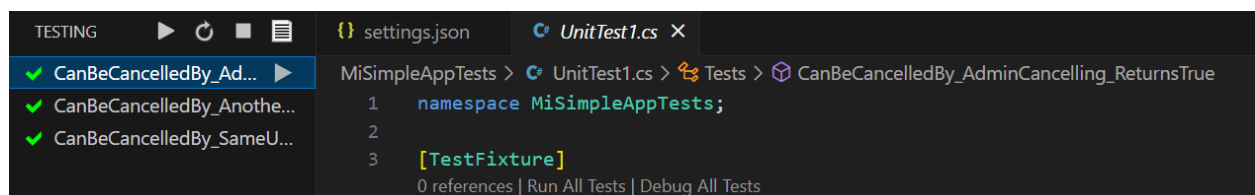
```

TP9 - Unit Test

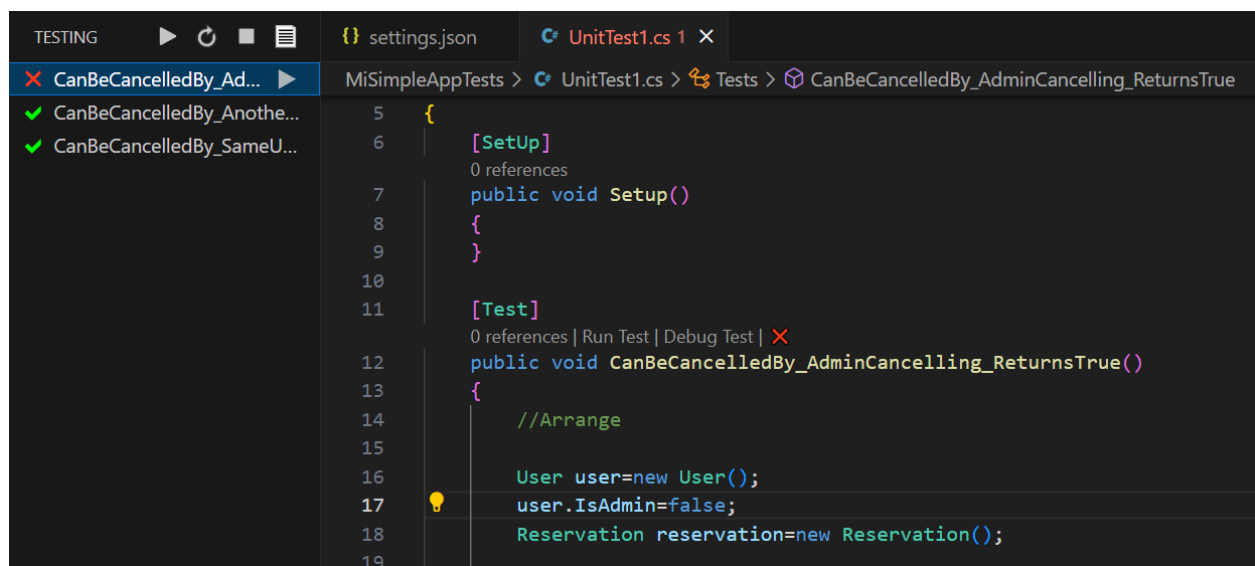
Las pruebas se centran en verificar el comportamiento del método **CanBeCancelledBy** bajo diferentes circunstancias. Aquí tienes una breve descripción de cada prueba:

1. `CanBeCancelledBy_AdminCancelling_ReturnsTrue` : Esta prueba verifica que si un usuario es un administrador (con la propiedad `IsAdmin` establecida en `true`), el método `CanBeCancelledBy` debe devolver `true`.
2. `CanBeCancelledBy_SameUserCancelling_ReturnsTrue` : Esta prueba verifica que si el usuario que realizó la reserva (`MadeBy`) es el mismo usuario que intenta cancelarla, el método `CanBeCancelledBy` debe devolver `true`.
3. `CanBeCancelledBy_AnotherUserCancelling_ReturnsFalse` : Esta prueba verifica que si el usuario que intenta cancelar la reserva no es el mismo que la hizo (`MadeBy`), el método `CanBeCancelledBy` debe devolver `false`.

Ejecutamos los test



- Modificamos la lógica de nuestro código bajo prueba haciendo que devuelva false la línea 17 (recordar guardar el archivo):



Ejecutamos nuestros tests desde la línea de comandos. Nos posicionamos en el directorio de nuestro proyecto de pruebas y ejecutamos el comando `dotnet test`

```
C:\ MINGW64:/c/Users/BELU/go/src/github.com/belenaguilarv/IngenieriaDeSoftware3/TP9 - Unit Test/MiSimpleAppTests
BELU@belenaguilarv MINGW64 ~/go/src/github.com/belenaguilarv/IngenieriaDeSoftware3/TP9 - Unit Test/MiSimpleAppTests (main)
$ dotnet test
Determinando los proyectos que se van a restaurar...
Se ha restaurado C:\Users\BELU\go\src\github.com\belenaguilarv\IngenieriaDeSoftware3\TP9 - Unit Test\MiSimpleApp\MiSimpleApp.csproj (en 176 ms).
Se ha restaurado C:\Users\BELU\go\src\github.com\belenaguilarv\IngenieriaDeSoftware3\TP9 - Unit Test\MiSimpleAppTests\MiSimpleAppTests.csproj (en 340 ms).
C:\Users\BELU\go\src\github.com\belenaguilarv\IngenieriaDeSoftware3\TP9 - Unit Test\MiSimpleApp\Clases.cs(4,21): warning CS8618: El elemento propiedad "MadeBy" que no acepta valores NULL debe contener un valor distinto de NULL al salir de l constructor. Considere la posibilidad de declarar el elemento propiedad como que admite un valor NULL. [C:\Users\BELU\go\src\github.com\belenaguilarv\IngenieriaDeSoftware3\TP9 - Unit Test\MiSimpleApp\MiSimpleApp.csproj]
MiSimpleApp -> C:\Users\BELU\go\src\github.com\belenaguilarv\IngenieriaDeSoftware3\TP9 - Unit Test\MiSimpleApp\bin\Debug\net7.0\MiSimpleApp.dll
MiSimpleAppTests -> C:\Users\BELU\go\src\github.com\belenaguilarv\IngenieriaDeSoftware3\TP9 - Unit Test\MiSimpleAppTests\bin\Debug\net7.0\MiSimpleAppTests.dll
Serie de pruebas para C:\Users\BELU\go\src\github.com\belenaguilarv\IngenieriaDeSoftware3\TP9 - Unit Test\MiSimpleAppTests\bin\Debug\net7.0\MiSimpleAppTests.dll (.NETCoreApp,Version=v7.0)
Herramienta de línea de comandos de ejecución de pruebas de Microsoft(R), versión 17.7.0-preview-23364-03+bc17bb9693cfc4778ded51aa0ab7f1065433f989 (x64)
Copyright (c) Microsoft Corporation. Todos los derechos reservados.

Iniciando la ejecución de pruebas, espere...
1 archivos de prueba en total coincidieron con el patrón especificado.

Correctas! - Con error:      0, Superado:      3, Omitido:      0, Total:      3, Duración: 613 ms - MiSimpleAppTests.dll (net7.0)
BELU@belenaguilarv MINGW64 ~/go/src/github.com/belenaguilarv/IngenieriaDeSoftware3/TP9 - Unit Test/MiSimpleAppTests (main)
$
```

## 6- Desarrollo de Pruebas Unitarias sobre una WebAPI:

```
MINGW64/c/Users/BELU/go/src/github.com/belenaguiarv/IngenieriaDeSoftware3/TP9 - Unit Test
BELU@belenaguiarv MINGW64 ~/go/src/github.com/belenaguiarv/IngenieriaDeSoftware3/TP9 - Unit Test (main)
$ git clone https://github.com/ingsoft3ucc/SimpleWebAPI.git
Cloning into 'SimpleWebAPI'...
remote: Enumerating objects: 150, done.
remote: Counting objects: 100% (150/150), done.
Receiving objects: 27% (41/150)% (113/113), done. 38/150)
Receiving oal 150 (delta 60), reused 37 (delta 11), pack-reused 0R
Receiving objects: 100% (150/150), 28.40 KiB | 1.09 MiB/s, done.
Resolving deltas: 100% (60/60), done.

BELU@belenaguiarv MINGW64 ~/go/src/github.com/belenaguiarv/IngenieriaDeSoftware3/TP9 - Unit Test (main)
$ dotnet new nunit -n SimpleWebAPITests
La plantilla "NUnit 3 Test Project" se creó correctamente.

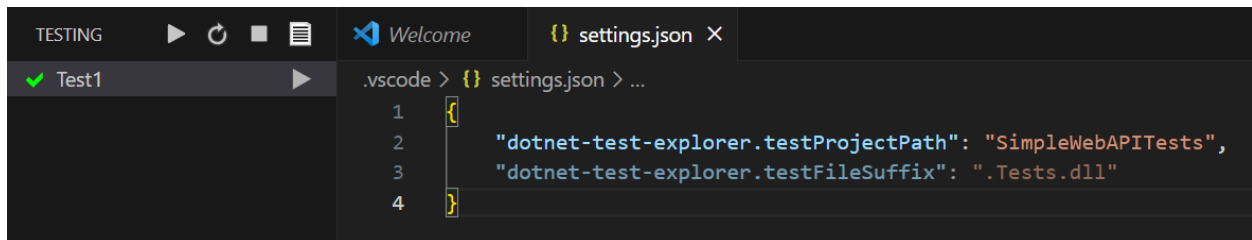
Procesando acciones posteriores a la creación...
Restaurando C:\Users\BELU\go\src\github.com\belenaguiarv\IngenieriaDeSoftware3\TP9 - Unit Test\SimpleWebAPITests\SimpleWebAPITest
s.csproj:
  Determinando los proyectos que se van a restaurar...
  Se ha restaurado C:\Users\BELU\go\src\github.com\belenaguiarv\IngenieriaDeSoftware3\TP9 - Unit Test\SimpleWebAPITest
s\SimpleWebAPITests.csproj (en 353 ms).
Restauración realizada correctamente.

BELU@belenaguiarv MINGW64 ~/go/src/github.com/belenaguiarv/IngenieriaDeSoftware3/TP9 - Unit Test (main)
$ cd SimpleWebAPITests

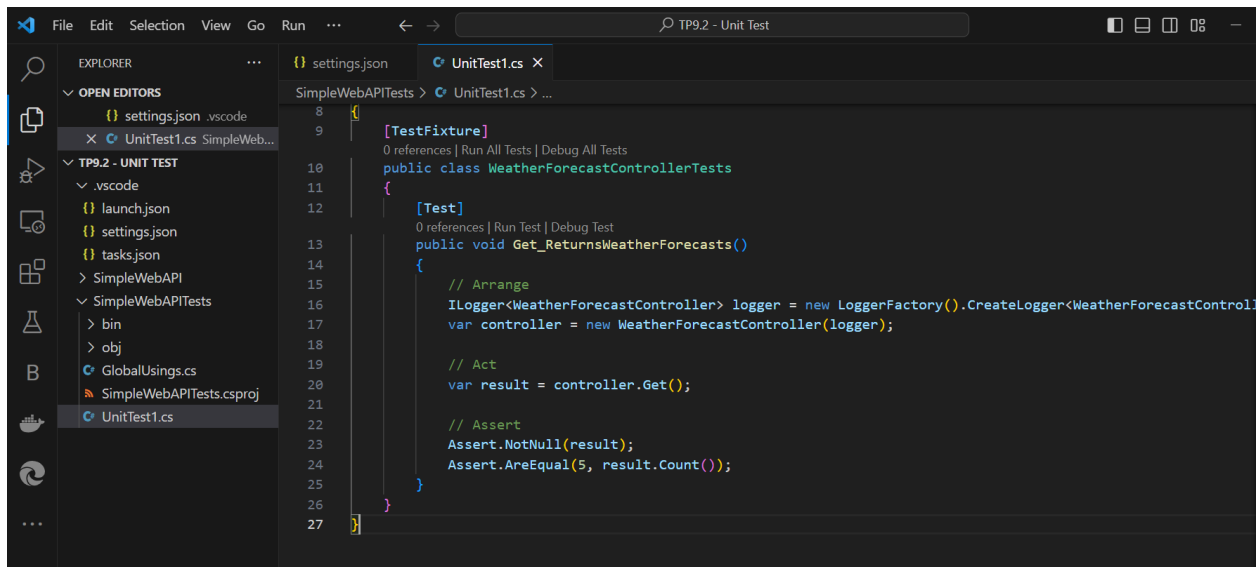
BELU@belenaguiarv MINGW64 ~/go/src/github.com/belenaguiarv/IngenieriaDeSoftware3/TP9 - Unit Test/SimpleWebAPITests (main)
$ dotnet add reference ../SimpleWebAPI/SimpleWebAPI/SimpleWebAPI.csproj
Se ha agregado la referencia "..\SimpleWebAPI\SimpleWebAPI\SimpleWebAPI.csproj" al proyecto.

BELU@belenaguiarv MINGW64 ~/go/src/github.com/belenaguiarv/IngenieriaDeSoftware3/TP9 - Unit Test/SimpleWebAPITests (main)
$ cd ..

BELU@belenaguiarv MINGW64 ~/go/src/github.com/belenaguiarv/IngenieriaDeSoftware3/TP9 - Unit Test (main)
$ code .
```



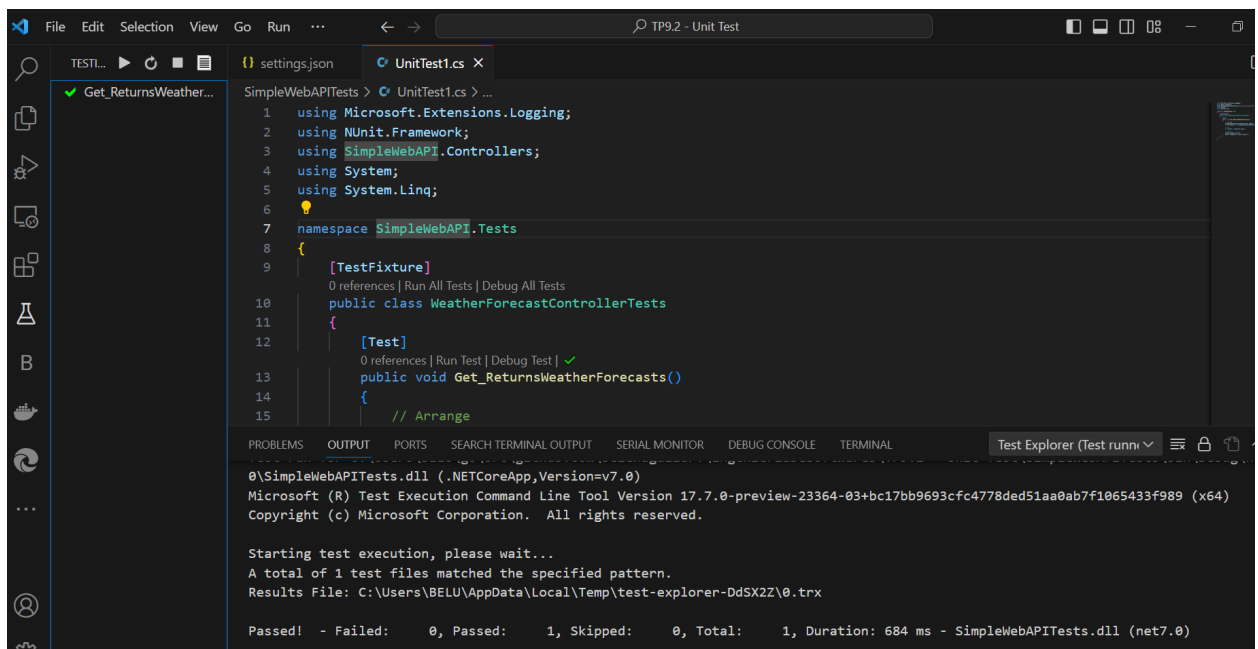
Creamos tests:



TP9.2 - Unit Test

Esta prueba unitaria se usa para verificar si el método `Get` del `WeatherForecastController` devuelve un resultado no nulo y si ese resultado contiene exactamente 5 elementos.

Ejecutamos el test



## 7- Familiarizarse con algunos conceptos de Mock

## 8- Utilizando Moq

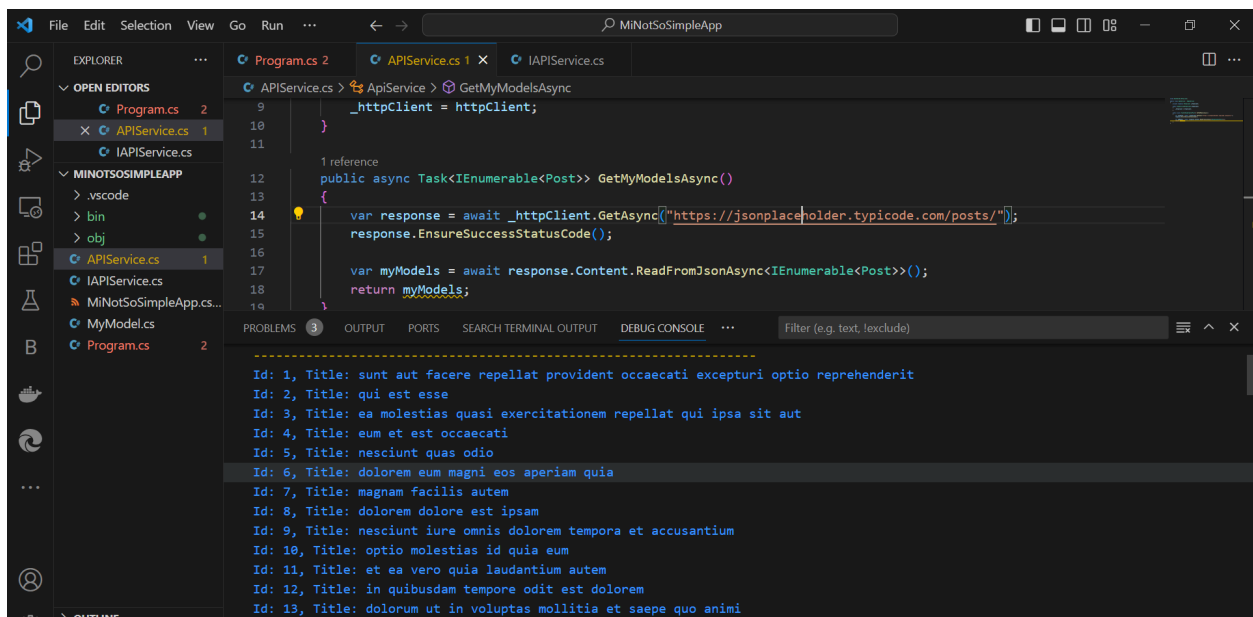
- Clonamos una app de consola en .NET Core que hace uso de un servicio externo (una llamada a una API Rest) y la abrimos en VS.Code

```
BELU@belenaguilarv MINGW64 ~/go/src/github.com/belenaguilarv/IngenieriaDeSoftware3/TP9 - Unit Test (main)
$ git clone https://github.com/ingsoft3succ/MiNotSoSimpleApp.git
Cloning into 'MiNotSoSimpleApp'...
remote: Enumerating objects: 22, done.
remote: Counting objects: 100% (22/22), done.
remote: Compressing objects: 100% (21/21), done.
remote: Total 22 (delta 6), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (22/22), 5.19 KiB | 1.30 MiB/s, done.
Resolving deltas: 100% (6/6), done.

BELU@belenaguilarv MINGW64 ~/go/src/github.com/belenaguilarv/IngenieriaDeSoftware3/TP9 - Unit Test (main)
$ cd MiNotSoSimpleApp

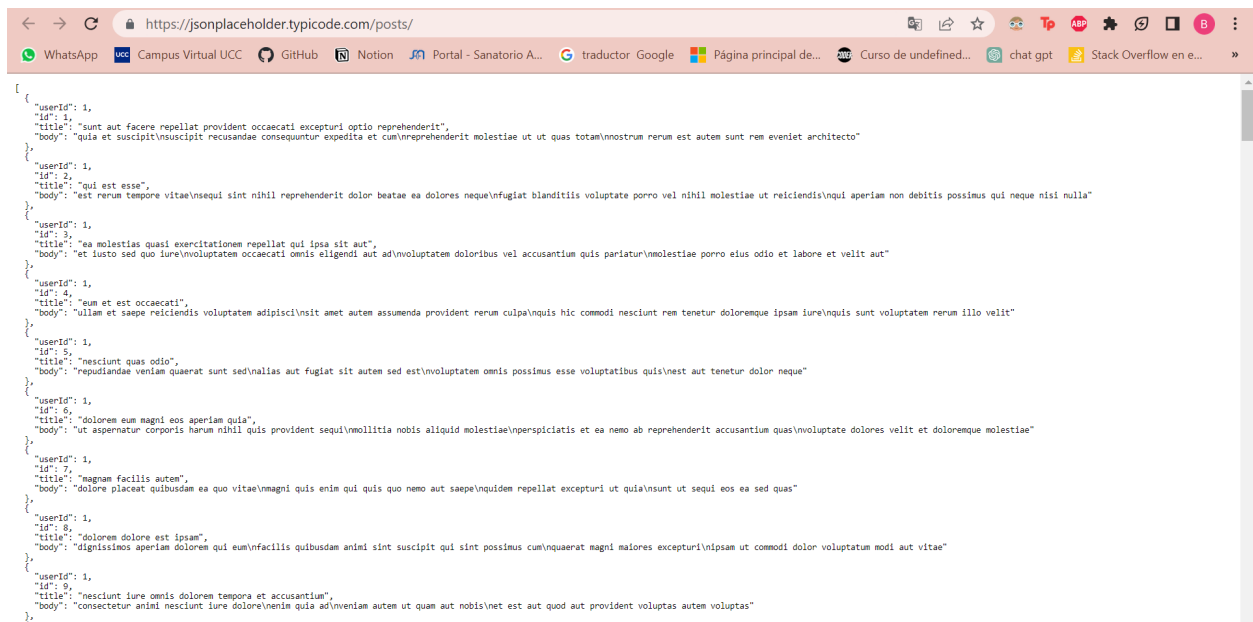
BELU@belenaguilarv MINGW64 ~/go/src/github.com/belenaguilarv/IngenieriaDeSoftware3/TP9 - Unit Test/MiNotSoSimpleApp (main)
$ code .
```

- Ejecutamos la app y vemos como nos devuelve 100 items desde la API:

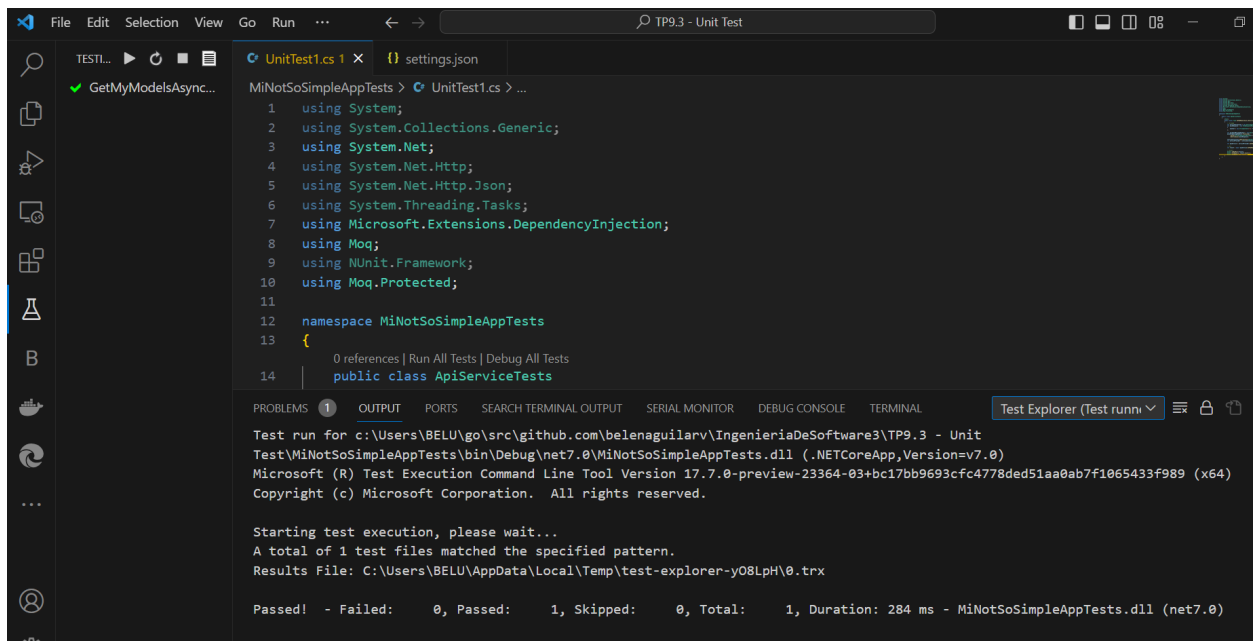


```
Id: 1, Title: sunt aut facere repellat provident occaecati excepturi optio reprehenderit
Id: 2, Title: qui est esse
Id: 3, Title: ea molestias quasi exercitationem repellat qui ipsa sit aut
Id: 4, Title: eum et est occaecati
Id: 5, Title: nesciunt quas odio
Id: 6, Title: dolorem eum magni eos aperiam quia
Id: 7, Title: magnam facillis autem
Id: 8, Title: dolorem dolore est ipsam
Id: 9, Title: nesciunt iure omnis dolorem tempora et accusantium
Id: 10, Title: optio molestias id quia eum
Id: 11, Title: et ea vero quia laudantium autem
Id: 12, Title: in quibusdam tempore odit est dolorem
Id: 13, Title: dolorum ut in voluptas mollitia et saepe quo animi
```





- Cerramos VS.Code y creamos el proyecto de NUnit, escribimos el test



TP9.3 - Unit Test

Este es un conjunto de pruebas unitarias escritas en C# utilizando el framework NUnit y Moq. Estas pruebas se centran en probar el método `GetMyModelsAsync` de una clase llamada `ApiService`.

Estas pruebas están diseñadas para verificar si el método `GetMyModelsAsync` se comporta correctamente al hacer una solicitud HTTP simulada y si devuelve los resultados esperados.

El uso de Moq permite simular el comportamiento del cliente HTTP, lo que hace que estas pruebas sean aisladas y no dependan de un servicio web real.