Documentación para la Ejecución y Test de la solución ImperialSniffer en Windows.

| **Revisión** | **Fecha** | **Descripción de Cambios** | **Autor(s)** | **Página** |
| --- | --- | --- | --- | --- |
| 1.0R0 | 12/02/2021 | Inicio del documento | Pablo Vasquez | Todas |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Contents

[1 Código del Proyecto 2](#_Toc64482708)

[1.1 Repositorio del proyecto 2](#_Toc64482709)

[1.2 Directorios del Proyecto 2](#_Toc64482710)

[Detalle del árbol de directorios 2](#_Toc64482711)

[1.3 Requisitos para compilación y debug del proyecto en Windows: 2](#_Toc64482712)

[2 Ejecución y prueba del Código. 2](#_Toc64482713)

[2.1 Ejercico LEVEL 1. 2](#_Toc64482714)

[2.2 Ejercicio LEVEL 2 y 3 3](#_Toc64482715)

# Código del Proyecto

## Repositorio del proyecto

El Código del proyecto ImperialSniffer se encuentra subido a GitHub en un repositorio público, accesible mediante el link:

<https://github.com/pmvasquez89/melichallenge.git>

## Directorios del Proyecto

### Detalle del árbol de directorios

#### Solution

* LEVEL 1: La solución en si ImperialSniffer que resuelve los métodos, GetLocatios y GetMessage.
* LEVEL 2 y 3: La API ImperialSniffer con los servicios topsecret y topsecret\_split solicitados. Dentro de este LEVEL existe también el proyecto de test unitarios ImperialSnifferTest.

#### Test

* POSTMAN : archivos con request para la prueba de los servicio de la API desde el cliente POSTMAN.

## Requisitos para compilación y debug del proyecto en Windows:

-Visual Studio 2019

-Frameworks: ASP.NET core Framework, Entity Framework

-Packages Nuget : Nlog, Swagger, Nunit.

# Ejecución y prueba del Código.

## Ejercico LEVEL 1.

Descargar el código de la carpeta LEVEL 1.

Una vez descargado el código y configurado Visual Studio con los requisitos anteriores, abrir el proyecto desde el navegador de Visual y compilar (Build). Y luego para ejecutarlo en Modo Debug.

Este programa abrirá una línea de comandos donde podemos ingresar los parámetros para probar el funcionamiento de la lógica de detección de posición y mensajes.

## Ejercicio LEVEL 2 y 3

Descargar el código de la carpeta LEVEL 2.

Una vez descargado el código y configurado Visual Studio con los requisitos anteriores, abrir el proyecto desde el navegador de Visual y compilar (Build). Y luego para ejecutarlo en Modo Debug podemos utilizar la Herramienta ISS Express que abrirá el navegador por defecto y permitirá realizar Request a la API.

Luego podemos ejecutar los test mediante botón derecho en en el proyecto ImperialSnifferTest + Run Test.

Se generaron Test unitarios de los Métodos GetLocation y GetMessage y luego test de integración con el uso de la base de datos del ejercicio topsecret\_split, para funcionamiento correcto y distintas posibles fallas(cabe aclarar que se podrían agregar mas, pero por cuestiones de tiempo se generaron solo algunos). Se incluyen comentarios en los Test.

Para este ejercicio se incluye además el proyecto de POSTMAN con los Request y la configuración necesarios para probar ambos servicios (topsecret y topsecret\_split).

Descargar el archivo “Fuego de Quasar Prod V5.postman\_collection.json” desde la carpeta **Test** en GitHub e importar desde POSTMAN.

Este proyecto de POSTMAN facilita los request que se solicitan en el ejercicio de MELI pero además tiene cargados Request GET al servicio topsecret\_split, que si bien no era un requerimiento, sirve para pruebas de la Base de Datos.