Challenge Nubimetrics

Selección .NET SR.

# Versionado del Documento

|  |  |  |
| --- | --- | --- |
| Autor | Cambios | Fecha |
| Aranguez Jonatan | Creación | 27/01/2022 |

# Ejercicio N-1 (Países y Búsqueda)

*Detalles técnicos*

La estructura de la solución está basada en los principios Onion Architecture (Inversión de dependencia) ya que considero que impulsa el orden y las buenas prácticas.

La solución cuenta con 4 proyectos:

* **Nubimetrics\_Challenge.WebApi**
* **Nubimetrics\_Challenge.Services**
* **Nubimetrics\_Challenge.EntityFrameworkCore**
* **Nubimetrics\_Challenge.Core**

En el orden listado, los proyectos actúan resolviendo las dependencias de forma inversa, siguiendo los lineamientos de Onion Architecture.

**Nubimetrics\_Challenge.WebApi:**Es el entry point del proyecto, donde ingresan los request y son derivados a la capa de negocio para ser procesados. Además, contiene todas las configuraciones iniciales del sistema, separadas en extensiones, como ser registro de servicios para inyección de dependencia, registro de repositorios, configuración de swagger y conexión a la base de datos, entre otras cosas.

**Nubimetrics\_Challenge.Services:**Contiene toda la lógica de negocio. Es la capa que se encarga de mapear los dto’s con las entidades de base de datos y viceversa, consumir los repositorios y conectar con servicios externos.

**Nubimetrics\_Challenge.EntityFrameworkCore:**Capa dedicada a la conexión con la base de datos. En ella se registran los repositorios, archivos de migraciones y context.

**Nubimetrics\_Challenge.Core:**Este es el proyecto central, el núcleo del sistema. Este proyecto particular no tiene dependencias con las capas superiores, pero es quien contiene la información necesaria para que las demás funcionen. En este caso, solo contiene los modelos de entidades de la base de datos.

Además de lo antes nombrado, podrán encontrar que utilizo “genéricos” de manera frecuente. Esto se debe a que considero una buena práctica centralizar comportamientos que pueden ser comunes dependiendo de las necesidades de la aplicación. En este caso, utilicé los genéricos para establecer un comportamiento común para operaciones de Alta, Baja, Modificación y Lectura en un controller base, servicio base, y repositorio base abstractos.

*Ejecución y pruebas*

Previo a ejecutar el proyecto, se debe inicializar la base de datos utilizando el script llamado “Nubimetrics script.sql” para poder contar con la tabla y sus registros.

El proyecto inicia con el portal de Swagger para poder realizar las pruebas necesarias con los endpoints y verificar su correcto funcionamiento.

En el archivo de configuración “**appsettings.json**” se encuentran las configuraciones correspondientes a los código de países desautorizados para acceder al sistema, y la url base correspondiente a la api de MercadoLibre.

# Ejercicio N-2 (Currency Conversions)

*Detalles técnicos*

Al igual que en el ejercicio 1, la estructura de la solución está basada en los principios Onion Architecture.

Ésta solución cuenta con solo 2 proyectos:

* **Nubimetrics\_ChallengeCurrencies.WebApi**
* **Nubimetrics\_ChallengeCurrencies.Services**

En el orden listado, los proyectos actúan resolviendo las dependencias de forma inversa, siguiendo los lineamientos de Onion Architecture.

**Nubimetrics\_ChallengeCurrencies.WebApi:**Es el entry point del proyecto, donde ingresan los request y son derivados a la capa de negocio para ser procesados. Además, contiene todas las configuraciones iniciales del sistema. También contienen las extensiones de la clase Startup para evitar el desorden y facilitar las configuraciones por herramienta.

**Nubimetrics\_ChallengeCurrencies.Services:**Capa orientada a la lógica de negocio, donde se efectúan las tareas más pesadas y genera conexiones con Apis externas, base de datos, etc.

*Ejecución y pruebas*

Tanto como en el proyecto del primer ejercicio, ésta api inicia con el portal de Swagger para poder realizar las pruebas necesarias con los endpoints y verificar su correcto funcionamiento.

En el archivo de configuración “**appsettings.json**” se encuentran las configuraciones correspondientes a la ruta donde serán almacenados con archivos CSV y JSON generados por el sistema, y la url base correspondiente a la api de MercadoLibre.