**Prueba Desarrollador Full Stack .NET Senior / Robinson Urquijo**

**Angular CRUD**

La URL por defecto es: <http://localhost:XXXX/evalcrud/index>

Para listar los registros y desde allí hacer los respectivas validaciones.

**Proyecto WebApi Amadeus NetCore 5.0**

**Componentes**:

* Swagger: Interfaz Visual para documentar y probar el API
* ADO.Net: Componente para conectarse a la Base de datos. SQL Server
* .NET Core: Framework de Desarrollo 5.0.3 o Superior.
* Base de datos SQL Server

**Aplicación principal (Capa de presentación)**

**AmadeusApi:** Contiene los controladores que intervienen con el desarrollo del API.

Se crean 1 controlador

TravelController

En donde se realiza aplicación de inyección de dependencias para el uso de interfaces dentro de la referenciación de las capas, según la arquitectura de las capas. Adicional se desarrollaron los métodos:

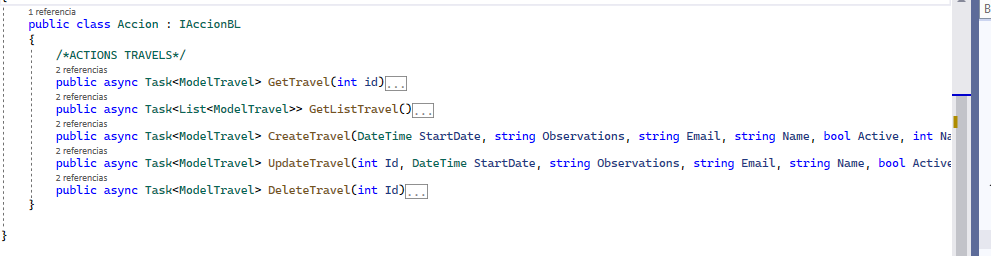
a) Travels  
b) Travel  
c) Create  
d) Update  
e) Delete

Se construyeron 4 bibliotecas de clases (Capas), adicionales al proyecto del API, el desarrollo esta creado bajo arquitectura MVC

* **Amadeus.BL:** Contiene los métodos que contemplan las funcionalidades del negocio, es intermediaria para interactuar con la capa de acceso a datos.
* **Amadeus.DAL:** Contiene los métodos, que se conectaran a la base de datos, se debe trabajar a través de procedimientos almacenados para mayor rendimiento del motor de base de datos y la aplicación, utilizando el componente ADO.NET.
* **Amadeus.Entities:** Contiene las entidades que intervienen en el proyecto.
* **Amadeus.Utility:** Contiene todos los componentes externos , que no hacen parte del proyecto , pero que si se necesitan para funcionar , como lo son dll externas , o utilitarios externos.

**Amadeus.BL** , la cual va a heredar la acción de la interfaz.

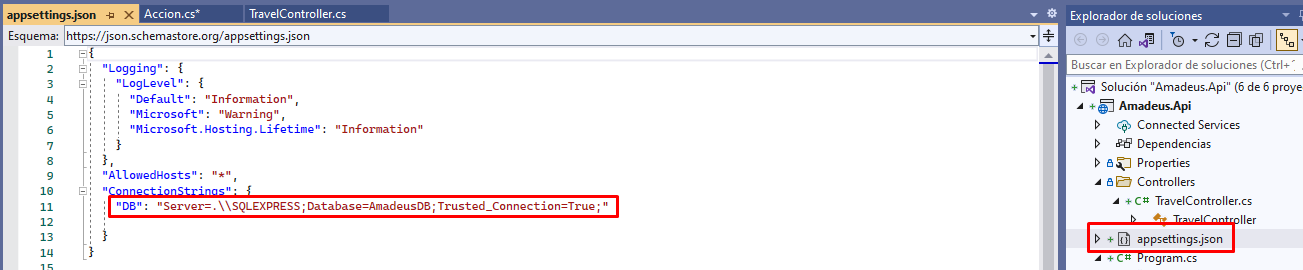
En la **interfaz** como tarea se referencia a la entidad que contiene los datos:



**Amadeus.Utility** se crea la clase **ApiConnectionStrings** la cual mapea las cadenas de conexión que este creadas en appsettings.json y usa el método Environment.GetEnvironmentVariable el cual se configura de manera, que se pueda trabajar en cualquier entorno.

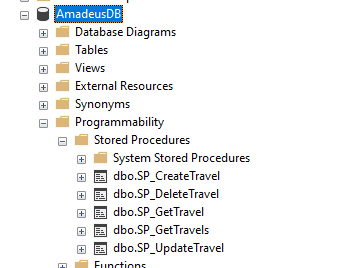
**Amadeus.DAL** Para realizar la conexión a la base de datos y pueda utilizar los procedimientos almacenados utilizando ADO.NET, se procede a crear en la capa Amadeus.DALen la clase el objeto de la cadena de conexión anteriormente creada, por medio de métodos asíncronos se configura y se traen los métodos mediante el using con componentes SqlClient y Data para traer el tipo de procedimiento almacenado.

**Al ejecutar los Script de Creación de la Base de datos deberá cambiar el apuntamiento y la forma de realizar la autenticación en el siguiente archivo:**



**Best Practices / Manage Performance**

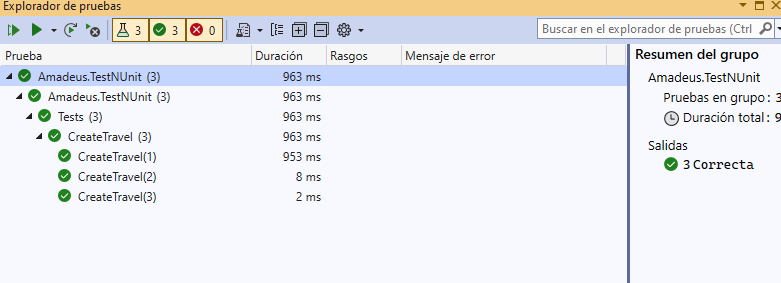
Se usaron Store Procedure dado que esto añade rendimiento sacándole todo el jugo a la Base de datos y teniendo una Aplicación ligera e independiente de los cambios que se produzca en la logica de negocio concentrada en los Store Procedure.



Se uso el Patron de arquitectura MVC que permite un menor acomplamiento entre las capas lo cual permite un mejor mantenimiento del código, despliegues.

**Unit Test**

Se crearon casos de pruebas con NUnit



Se implementa Swagger

