Documentación API

**Descripción**

Documentación para el challenge de IntuIt en donde se detalla el funcionamiento de las APIs del ABM (Alta, Baja, Modificación). El backend está desarrollado siguiendo principios de Clean Architecture, utiliza MediatR para commands y querys, utiliza algunas librerías populares como automapper, fluentValidation, registra logs de errores y tiene pruebas unitarias con xunit. Se presentará como extra para facilitar la prueba del mismo un archivo con una colección de Postman con los distintos endpoints. Importante aclarar que se desarrolló el DeleteClient, el cual no se solicitaba explícitamente pero como se pedía un ABM, era importante tenerlo.

Es importante aclarar que dentro de las carpetas, hay una que se llama “Front” en donde está el código de la app del clima. Y la carpeta que se llama “challenge” contiene el backend correspondiente al ABM de Clientes.

**Base URL**

[http://localhost:{puerto}/api/Cliente](about:blank)

**Endpoints**

**1. Crear Cliente**

* **URL:** /Create
* **Método HTTP:** POST
* **Descripción:** Crea un nuevo cliente en el sistema.
* **Request Body:**

|  |
| --- |
| {  "name": "string",  "lastName": "string",  "email": "string",  "cellphone": "string",  "cuit": "string",  "address": "string",  "birthdate": "yyyy-MM-dd" } |

* **Respuesta exitosa:**
* **Código:** 201 Created
* **Ejemplo:**

|  |
| --- |
| {  "result": {  "success": true,  "data": {  "clientId": 1  },  "messages": null  } } |

**2. Obtener Cliente por ID**

* **URL:** /GetById
* **Método HTTP:** GET
* **Descripción:** Obtiene información de un cliente según su Id.
* **Parámetros de Query:**
* Id (obligatorio, entero)
* **Ejemplo de request:** /GetById?id=1
* **Respuesta exitosa:**
* **Código:** 200 OK
* **Ejemplo:**

|  |
| --- |
| {  "id": 1,  "name": "Alejandro",  "lastName": "Ibañez",  "email": "alejandro.ibanez@intuit.com",  "cellphone": "3834304799",  "cuit": "20390153652",  "address": "Calle Siempre Viva 999",  "birthdate": "1996-04-23T00:00:00" } |

**3. Obtener Todos los Clientes**

* **URL:** /GetAll
* **Método HTTP:** GET
* **Descripción:** Obtiene una lista paginada de clientes con opción de búsqueda.
* **Parámetros de Query:**
* page (obligatorio, entero)
* pageSize (obligatorio, entero)
* textToSearch (opcional, string)
* **Ejemplo de request:** /GetAll?page=1&pageSize=10&textToSearch=Ale
* **Respuesta exitosa:**
* **Código:** 200 OK
* **Ejemplo:**

|  |
| --- |
| {  "data": [  {  "id": 1,  "name": "Alejandro",  "lastName": "Ibañez",  "email": "alejandro.ibanez@intuit.com"  },  {  "id": 2,  "name": "Bruce",  "lastName": "Wayne",  "email": "bruce.wayne@intuit.com"  }  ],  "totalPages": 1,  "currentPage": 1,  "totalRecords": 2 } |

**4. Actualizar Cliente**

* **URL:** /Update
* **Método HTTP:** PUT
* **Descripción:** Actualiza los datos de un cliente.
* **Request Body:**

|  |
| --- |
| {  "clientId": 1,  "name": "string",  "lastName": "string",  "email": "string",  "cellphone": "string",  "cuit": "string",  "address": "string",  "birthdate": "yyyy-MM-dd" } |

* **Respuesta exitosa:**
* **Código:** 200 Created
* **Ejemplo:**

|  |
| --- |
| {  "result": {  "success": true,  "data": {  "clientId": 1  },  "messages": null  } } |

**5. Eliminar Cliente**

* **URL:** /Delete
* **Método HTTP:** DELETE
* **Descripción:** Elimina un cliente.
* **Parámetros de query:**
* Id (obligatorio, entero)
* **Ejemplo de request:** /Delete?id=1
* **Respuesta exitosa:**
* **Código:** 200 OK
* **Ejemplo:**

|  |
| --- |
| {  "success": true,  "messages": {  "statusCode": 200,  "message": "Client deleted successfully."  } } |

**Unit Tests**

**Propósito:** Se creó un proyecto de xunit, dentro una carpeta llamada handlers en donde habrá un archivo con test para cada uno de los handlers.

* CreateClientHandlerTests.cs
* UpdateClientHandlerTests.cs
* DeleteClientHandlerTests.cs
* GetAllClienstHandlerTests.cs
* GetClientByIdHandlerTests.cs

Cada una de estas pruebas se pueden ejecutar, desde el “explorador de pruebas”, accesible desde la pestaña “pruebas” del visual studio.

**Colección de Postman**

Adjunto una colección de postman en formato json.

Cómo usarlo:

1. Importa el archivo en Postman.

2. Configura la variable de entorno para el BaseURL.

3. Ejecuta los endpoints según las pruebas necesarias.