

CHALLENGE INTUIT BACKEND

¿Cuál es el funcionamiento de la API?

Mediante el uso de esta API, se facilita la administración de clientes. A continuación, una descripción más detallada de cada endpoint:

1. GetAll

Permite la visualización de la lista completa de clientes cargados, incluida toda su información registrada.

Cientes

GET /api/Clientes/GetAll

Parameters

Cancel

No parameters

Execute

Clear

Responses

Curl

```
curl -X 'GET' \
  'https://localhost:7114/api/Clientes/GetAll' \
  -H 'accept: text/plain'
```

Request URL

https://localhost:7114/api/Clientes/GetAll

Server response

Code Details

200

Response body

```
[
  {
    "id": 1,
    "nombres": "Jorge Ignacio",
    "apellidos": "López Ruiz",
    "fechaDeNacimiento": "1980-05-21T00:00:00",
    "cuit": "2050430211",
    "domicilio": "Av Santa Fe 1992",
    "telefonoCelular": "1135953222",
    "email": "jorlopez@mail.com"
  },
  {
    "id": 2,
    "nombres": "Julio",
    "apellidos": "Martínez",
    "fechaDeNacimiento": "1983-12-01T00:00:00",
    "cuit": "2397421",
    "domicilio": "Av Rivadavia 5432",
    "telefonoCelular": "31281231",
    "email": "Julio@mail.com"
  },
  {
    "id": 6,
    "nombres": "Sandra",
```

2. Get (ID)

Igual que el endpoint anterior, pero en este caso traerá solamente el registro que coincida con el ID ingresado como parámetro.

GET

/api/Clientes/Get/{ID}

⌵

Parameters

Cancel

Name	Description
ID • required	
integer(\$int32)	2
(path)	

Execute

Clear

Responses

Curl

curl -X 'GET' \

'https://localhost:7114/api/Clientes/Get/2' \

-H 'accept: text/plain'

📄

Request URL

https://localhost:7114/api/Clientes/Get/2

Server response

Code	Details
200	<div><div>Response body</div><div><pre>{ "id": 2, "nombres": "Julio", "apellidos": "Martínez", "fechaDeNacimiento": "1983-12-01T00:00:00", "cuit": "2397421", "domicilio": "Av Rivadavia 5432", "telefonoCelular": "31281231", "email": "Julio@mail.com" }</pre></div><div>📄 Download</div></div>

3. Search

Muy similar al anterior, solo que, en este caso, la búsqueda se realiza a partir del nombre del cliente, incluso, no es necesario que se escriba completo. El resultado son todas las coincidencias que contengan, en su nombre, el parámetro enviado.

GET

/api/Cientes/search/{nombre}

⌵

Parameters

Cancel

Name	Description
nombre * required	
string	
(path)	

Execute

Clear

Responses

Curl

curl -X 'GET' \
'https://localhost:7114/api/Cientes/search/Eli' \
-H 'accept: text/plain'

Request URL

https://localhost:7114/api/Cientes/search/Eli

Server response

Code	Details
200	<div><div>Response body</div><div><pre>{ "id": 7, "nombres": "Elias", "apellidos": "Castro", "fechaDeNacimiento": "2000-10-09T00:00:00", "cuit": "20321239410", "domicilio": "Gascón 1544", "telefonoCelular": "1123431212", "email": "elias@mail.com" }</pre></div><div><div>Download</div></div></div>

4. Insert

Ahora, en el caso de este endpoint, posibilita agregar clientes a la base de datos, validando campos obligatorios como lo son: nombres, apellidos, CUIT, telefono celular y su email. Además, se valida que la fecha ingresada sea efectivamente una fecha, que el mail tenga el formato correspondiente y lo mismo para el CUIT, el cual será siempre numérico, de 11 dígitos y, por supuesto, mayor a 0. En caso de incumplirse alguna de las validaciones, no se da de alta el cliente en la base de datos. Finalmente, en caso de subirse exitosamente a la base, el JSON se podrá visualizar como respuesta del envío.

POST

/api/Cientes/Insert

Parameters

Cancel

Reset

No parameters

Request body

application/json

```
{
  "id": 0,
  "nombres": "Marcelo",
  "apellidos": "Rojas",
  "fechaDeNacimiento": "1992-07-19",
  "cuit": "25358395018",
  "domicilio": "Arenales 832",
  "telefonoCelular": "118327124",
  "email": "marcelo@mail.com"
}
```

Execute

Clear

Server response

Code

Details

201

Undocumented

Response body

```
{
  "id": 9,
  "nombres": "Marcelo",
  "apellidos": "Rojas",
  "fechaDeNacimiento": "1992-07-19T00:00:00",
  "cuit": "25358395018",
  "domicilio": "Arenales 832",
  "telefonoCelular": "118327124",
  "email": "marcelo@mail.com"
}
```

Download

5. Update

Este último endpoint, cumple con la función de actualizar la información del cliente que recibe como parámetro, realizando la modificación, al igual que en el endpoint de alta de cliente, solo en caso de que todos los campos sean correctos, es decir, luego de ser validados.

PUT

/api/Clientes/Update/{id}

⌵

Parameters

Cancel

Reset

Name	Description
id * required	
integer(\$int32)	9
(path)	

Request body

application/json

```
{  "id": 9,  "nombres": "Marcelo Andrés",  "apellidos": "Rojas",  "fechaDeNacimiento": "1994-07-19",  "cuit": "25388399018",  "domicilio": "Arenales 1832",  "telefonoCelular": "118327124",  "email": "rojasmarcelo@mail.com"}
```

Execute

Clear

Responses

Curl

```
curl -X 'PUT' \  https://localhost:7114/api/Clientes/Update/9' \  -H 'accept: */*' \  -H 'Content-Type: application/json' \  -d '{  "id": 9,  "nombres": "Marcelo Andrés",  "apellidos": "Rojas",  "fechaDeNacimiento": "1994-07-19",  "cuit": "25388399018",  "domicilio": "Arenales 1832",  "telefonoCelular": "118327124",  "email": "rojasmarcelo@mail.com"  }'
```

Request URL

https://localhost:7114/api/Clientes/Update/9

Server response

Code	Details
200	Response headers

Acerca de la API

La API ha sido desarrollada en .NET 8, utilizando, además, Entity Framework y SQL Server. Se ha utilizado un controller para el desarrollo de toda la lógica de la aplicacion y, en la capa “Models” se ha diseñado, por un lado, la clase “Clientes” con toda la información del cliente, es decir, todos los campos que lo componen. Por otro lado, “ClientesContext” para poder relacionarlo y operar con la base de datos.