

Roles

Billetera

Alta de usuario

1. Una persona ingresa a la app y se da de **alta** como **usuario**.

POST /api/Usuarios

Parameters

No parameters

Request body

Request body

```
{
  "Mail": "usuaria2@ejemplo.com",
  "Nombre": "Marta",
  "Apellido": "Pérez",
  "Password": "1234",
  "Direccion": "Paz 123",
  "Telefono": "1234567890",
  "Tipo_cliente": "BILLETERA"
}
```

Se cargan los datos que van a ir en el **formulario de alta de usuario** 'BILLETERA'. Para ejecutar el POST de prueba cargar:

```
{
  "nombre": "Leticia",
  "apellido": "Paez",
  "direccion": "Karukinka 65",
  "mail": "lety@gmail.com",
  "Password": "leti25",
  "tipo_cliente": "BILLETERA",
  "telefono": "542901459142"
}
```

Resultado de prueba:

Al correr el post se asigna estado true (activo en frontd) y nro de cliente secuencial (**etapa3 :ver de poner que sea a partir de un X numero fijo**)

Code

Details

201

Undocumented

Response body

```
{
  "$id": "1",
  "id": 8,
  "mail": "usuaria2@ejemplo.com",
  "nombre": "Marta Pérez",
  "tipo_cliente": "BILLETERA"
}
```

En base de datos se observa el insert del usuario nuevo con datos de pass hasheada

	nro_cliente	nombre	apellido	direccion	mail	estado	tipo_cliente	telefono	password_hash
1	1	Maria	Pérez	Calle Falsa 123	maria@example.com	0	BILLETERA	1123456789	tuHashDePasswordAqui
2	3	Maria	Pérez	Calle Falsa 123	maria2@example.com	0	BILLETERA	1123456785	1234
3	4	Sophia	Holm	Acigami 65	sophy@gmail.com	0	BILLETERA	542901617834	sofy25
4	5	Veronica	Sanchez	Onas 765	vero@gmail.com	0	BILLETERA	542901458109	vero25
5	6	Maria Luz	Holm	Acigami 65	mluz@gmail.com	0	BILLETERA	542901617764	lucy25
6	7	Marta	Pérez	Calle Falsa 123	usuariaA@ejemplo.com	0	Regular	1234567890	\$2a\$11\$Nz7AWJV/8PWtuZPCuox2IObmD1QWEIgxUpJ6spTg...
7	8	Marta	Pérez	Paz 123	usuaria2@ejemplo.com	0	BILLETERA	1234567890	\$2a\$11\$AGpQk8tLjQ/PZyd2wx7t2uoMBWOGnI.wLnd7L2huJP...

Login

2. El usuario ingresa a la app con su mail y password.

POST `/api/Usuarios/login`

Usuario incorrecto:

```
{
  "mail": "sophy@gmail.com",
  "password": "sofy25"
}
```

Code	Details
401 <i>Undocumented</i>	Error: Unauthorized Response body Credenciales inválidas.

Usuario correcto:

Si ingresa exitosamente se asigna un token que al validarlo desde el autorizador habilita funcionalidades para el usuario.

Se pone **seguridad JWT** porque no cualquiera debería poder visualizar y editar datos del usuario.

```
{
  "Mail": "usuario2@ejemplo.com",
  "Password": "1234"
}
juan.gonzalez@gmail.com
{
  "Mail": "lety@gnail.com ",
  "Password": "leti25"
}
```

[illegible]

Validación del TOKEN:

Available authorizations

Bearer (http, Bearer)

Ingrese el token

Value:

eyJhbGciOiJIUzI1NiIsInR5cC

Authorize

Close

Available authorizations

Bearer (http, Bearer)

Authorized

Ingrese el token

Value: *****

Logout

Close

Alta de cuenta

3. Con el usuario logueado, se puede **crear** una cuenta automáticamente; no lleva body porque en el frontd se tiene que generar:

POST /api/Cuentas



Parameters

No parameters

Execute

Se **crea** la cuenta asociada al usuario logueado.

Code

Details

201

Undocumented

Response body

```
{
  "$id": "1",
  "nroCuenta": 89453,
  "nroCliente": 1,
  "producto": "Caja de ahorro",
  "cbu": "2680061102089237289453",
  "estado": true,
  "rolCta": "Titular",
  "saldo": 0,
  "fechaCreacion": "0001-01-01T00:00:00"
}
```

	nro_cuenta	producto	CBU	estado	nro_cliente	rol_cta
1	89453	Caja de ahorro	2680061102089237289453	1	1	Titular

Consulta de datos de la cuenta

- El usuario logueado puede **consultar** los datos de su **cuenta**

GET

/api/Cuentas/por-cliente

Parameters

No parameters

Execute

Code	Details
200	<p>Response body</p> <pre>{ "\$id": "1", "\$values": [{ "\$id": "2", "nroCuenta": 89453, "cbu": "2680061102089237289453", "nombre": "Leticia", "apellido": "Paez", "saldo": 0 }] }</pre>

Modificar el alias

PUT `/api/Cuentas/alias`

```
{
  "nroCuenta": 41947,
  "nuevoAlias": "sol.luna.82"
}
```

	nro_cuenta	producto	CBU	estado	nro_cliente	rol_cta	saldo	fecha_alta	alias
1	41947	Caja de ahorro	2680061102086594641947	1	1	Titular	NULL	2025-06-02 11:10:50.8933333	fuego.tierra.sol.sol

Code	Details
200	<p>Response body</p> <pre>Alias actualizado correctamente.</pre>

Consulta de datos del Usuario

- El usuario billetera, puede **consultar** sus datos de **Registración**.

GET `/api/Usuarios/{id}`

Parameters

Name	Description
id * required	
integer(\$int32)	1
(path)	

Execute

Code	Details
200	<p>Response body</p> <pre>{ "\$id": "1", "id": 1, "mail": "lety@gnail.com", "nombre": "Leticia", "tipo_cliente": "BILLETERA" }</pre>

Modificación de datos del Usuario

6. El usuario billetera, puede **modificar** sus datos de **Registración**.

DE:

	nro_cliente	nombre	apellido	direccion	mail	estado	tipo_cliente	telefono	password_hash
1	1	Laura	Alé	Garcia Marquez 4465	lau@gmail.com	0	BILLETERA	542901427491	\$2a\$11\$QLH4Rn4isxwcf3LB8rXMaujUmjkwW3OZdGF69i8vy...

A:

nro_cliente	nombre	apellido	direccion	mail	estado	tipo_cliente	telefono	password_hash
1	Laura	Alé	Neruda 4455	lau@gmail.com	0	BILLETERA	542901427491	\$2a\$11\$QLH4Rn4isxwcf3LB8rXMaujUmjkwW3OZdGF69i8vy...

```
{
  "nombre": "Laura",
  "apellido": "Alé",
  "direccion": "Neruda 4455",
  "mail": "lau@gmail.com",
  "telefono": "542901427491"
}
```

Consulta de un movimiento

7. El usuario billetera, puede **consultar** el detalle de un **movimiento**

GET

/api/Movimientos/{id}

Parameters

Name	Description
id * required	
integer(\$int32)	1
(path)	

Execute

Code	Details
200	<div>Response body</div> <pre>{ "\$id": "1", "id_trx": 1, "fecha": "2025-05-31T16:18:00.9099408", "monto": 10, "nro_cuenta_orig": 91232, "nro_cuenta_dest": 40101, "codigo_transaccion": 2, "cuentaOrig": null, "cuentaDest": null, "transaccion": null }</pre>

8. El usuario logueado puede realizar **transacciones** entre cuentas con validaciones.
Déposito cuenta propia (ingresar dinero)

El cliente realiza un depósito en su cuenta:

POST

/api/Movimientos/depositar

Request body

```
{
  "monto": 50000
}
```

Server response

Code	Details
200	<div>Response body</div> <pre>{ "\$id": "1", "id": 2, "mensaje": "Movimiento registrado." }</pre>

	id_trx	fecha	monto	nro_cuenta_orig	nro_cuenta_dest	codigo_transaccion
1	1	2025-05-31 16:18:00.9099408	10.00	91232	40101	2
2	2	2025-05-31 23:56:18.1822382	5000.00	91232	91232	3

Responses

Code	Description
200	OK

Transferencias

POST

/api/Movimientos/transferir

Se hacen validaciones

Response body

La cuenta destino no puede ser la misma que tu cuenta origen.

Code

Details

400

Undocumented

Error: Bad Request

Response body

Saldo insuficiente en la cuenta de origen.

Transferencia a otra cuenta

El usuario logueado accede a sus cuentas y selecciona su cuenta y como transfiere a una cuenta que no está en la base se genera solo un movimiento en la base de datos. En caso de que la cuenta se encuentre en la base, se generan 2 registros en la tabla 'Movimientos'

```
{
  "nroCuentaOrigen": 60659,
  "nroCuentaDestino": 422,
  "monto": 110
}
```

Code

Details

200

Response body

```
{
  "$id": "1",
  "idDebito": 3,
  "idCredito": null,
  "mensaje": "Transferencia procesada correctamente."
}
```

Se valida si la cuenta tiene saldo y luego de hacer un deposito se puede realizar la transferencia.

```
{
  "nroCuentaOrigen": 60659,
  "nroCuentaDestino": 622,
  "monto": 110
}
```

Code	Details
200	<div>Response body</div> <div><pre>{ "\$id": "1", "idDebito": 5, "idCredito": 6, "mensaje": "Transferencia procesada correctamente." }</pre></div>

Últimos movimientos_resumen

POST /api/Movimientos/ultimos

Parameters

No parameters

Request body

```
{
  "fechaDesde": "2025-06-02",
  "fechaHasta": "2025-06-02"
}
```


Code	Details
200	<p>Response body</p> <pre> { "\$id": "1", "\$values": [{ "\$id": "2", "idTrx": 2, "descripcionTransaccion": "Transferencia a otra cuenta", "monto": 100, "fecha": "2025-06-02T14:12:31.5475363", "nroCuentaOrigen": 41947, "nroCuentaDestino": null }, { "\$id": "3", "idTrx": 1, "descripcionTransaccion": "Depósito cuenta propia", "monto": 50000, "fecha": "2025-06-02T14:11:12.1122517", "nroCuentaOrigen": null, "nroCuentaDestino": 41947 }] }</pre>

Últimos movimientos_pantalla principal:
LISTO! Faltan pantallas en este doc

Administrador

1. El usuario 'ADMINISTRADOR' se crea en base de datos

POST /api/Usuarios/admin/primer_user

nro_cliente	nombre	apellido	direccion	mail	estado	tipo_cliente	telefono	password_h
3	Carlos	Ruiz	Avenida Libertador 782	carlosr@gmail.com	1	ADMINISTRADOR	541131226789	\$2a\$11\$Qr

2. El usuario 'ADMINISTRADOR' ingresa con su ID y tipo de usuario a la aplicación de manera de tener acceso a distintas funcionalidades.

POST /api/Usuarios/login

200 – OK

401 – Rechazado: en caso de que no sea usuario administrador no puede ingresar:

Cuando da el token se pueden correr métodos con seguridad

3. Puede modificar el estado de un cliente sin cuenta (de 1 a 0) –

4. El administrador visualiza datos de cualquier usuario (*mejora etapa3: el jwt del admin*)

5. El administrador visualiza datos de cuentas (*mejora etapa3: el jwt del admin*)

GET /api/Cuentas