# **Definición e Implementación de la API del Servicio de Cuentas**

## **Introducción**

El **Servicio de Cuentas** (AccountService) es un microservicio dentro del sistema MyOnlineShoppingService, diseñado para gestionar cuentas de usuarios en un marketplace. Este servicio permite a los clientes y administradores interactuar con cuentas mediante una API RESTful, cumpliendo con los requisitos de seguridad (OAuth2), formatos de respuesta (JSON y XML), y validaciones de correspondencia. La API está implementada usando Spring Boot y sigue un diseño basado en microservicios para garantizar independencia, resiliencia y escalabilidad.

## **Definición de la API**

### **Características generales**

* **Ruta base**: /api/v1 (perfil prod) o /default (perfil dev).
* **Seguridad**: Todos los endpoints están protegidos con OAuth2. Las solicitudes deben incluir un token de acceso válido en el encabezado Authorization: Bearer <token>.
* **Formatos**:
  + **Entrada**: Solo se acepta JSON (application/json) en el cuerpo de las solicitudes.
  + **Salida**: Los endpoints pueden devolver JSON (application/json) o XML (application/xml), según el encabezado Accept de la solicitud.
* **Validaciones**:
  + Los datos de entrada se validan con anotaciones como @Valid, @Min, y @NotBlank.
  + Se valida la correspondencia entre el usuario autenticado y el propietario de las cuentas.
* **Códigos de estado**:
  + 200 OK: Operación exitosa (GET, PUT).
  + 201 Created: Recurso creado (POST).
  + 204 No Content: Eliminación exitosa (DELETE).
  + 400 Bad Request: Error en los datos de entrada o validación.
  + 401 Unauthorized: Token OAuth2 inválido o ausente.
  + 403 Forbidden: El usuario no tiene permisos para la operación.
  + 404 Not Found: Recurso no encontrado.

## **Endpoints de la API**

### **1. Obtener una cuenta específica**

* **Ruta**: /api/v1/default/account/{id}/customer/{owner\_id}
* **Método**: GET
* **Descripción**: Devuelve una cuenta específica, validando que el ID de la cuenta y el propietario se corresponden.
* **Parámetros**:
  + id (path): ID de la cuenta (Long, mínimo 1).
  + owner\_id (path): ID del propietario (Long, mínimo 1).
* **Encabezados**:
  + Authorization: Bearer <token> (requerido, token OAuth2).
  + Accept: application/json (opcional, para forzar JSON).
* **Retorno**:

**Éxito (200 OK)**: Un objeto AccountDTO en formato JSON o XML.  
 json  
ContraerAjusteCopiar  
{

"id": 1,

"type": "Personal",

"balance": 1000.0,

"openingDate": "2023-01-01",

"ownerId": 1

* + }

**Error (400 Bad Request)**: Si los parámetros no cumplen las validaciones.  
 json  
ContraerAjusteCopiar  
{

"error": "El ID debe ser mayor o igual a 1"

* + }

**Error (403 Forbidden)**: Si el usuario autenticado no tiene permiso para acceder a la cuenta.  
 json  
ContraerAjusteCopiar  
{

"error": "La cuenta no pertenece al usuario autenticado"

* + }

**Error (404 Not Found)**: Si la cuenta no existe.  
 json  
ContraerAjusteCopiar  
{

"error": "Cuenta no encontrada"

* + }

### **2. Listar cuentas de un usuario**

* **Ruta**: /api/v1/default/accounts/customer/{id}
* **Método**: GET
* **Descripción**: Devuelve la lista de cuentas asociadas a un usuario.
* **Parámetros**:
  + id (path): ID del propietario (Long, mínimo 1).
* **Encabezados**:
  + Authorization: Bearer <token> (requerido, token OAuth2).
  + Accept: application/json (opcional, para forzar JSON).
* **Retorno**:

**Éxito (200 OK)**: Una lista de objetos AccountDTO en formato JSON o XML.  
 json  
ContraerAjusteCopiar  
[

{

"id": 1,

"type": "Personal",

"balance": 1000.0,

"openingDate": "2023-01-01",

"ownerId": 1

},

{

"id": 2,

"type": "Company",

"balance": 2000.0,

"openingDate": "2023-02-01",

"ownerId": 1

}

* + ]
  + **Éxito (204 No Content)**: Si no hay cuentas para el usuario.

**Error (400 Bad Request)**: Si el ID no cumple las validaciones.  
 json  
ContraerAjusteCopiar  
{

"error": "El ID debe ser mayor o igual a 1"

* + }

**Error (403 Forbidden)**: Si el usuario autenticado no tiene permiso.  
 json  
ContraerAjusteCopiar  
{

"error": "No tienes permiso para listar estas cuentas"

* + }

### **3. Crear una cuenta**

* **Ruta**: /api/v1/default/account/customer/{owner\_id}
* **Método**: POST
* **Descripción**: Crea una nueva cuenta para un usuario.
* **Parámetros**:
  + owner\_id (path): ID del propietario (Long, mínimo 1).

Cuerpo de la solicitud (JSON):  
 json  
ContraerAjusteCopiar  
{

"accountNumber": "12345",

"type": "Personal",

"balance": 1000.0,

"openingDate": "2023-01-01"

* + }
* **Encabezados**:
  + Authorization: Bearer <token> (requerido, token OAuth2).
  + Content-Type: application/json (requerido).
  + Accept: application/json (opcional, para forzar JSON).
* **Retorno**:

**Éxito (201 Created)**: La cuenta creada en formato JSON o XML.  
 json  
ContraerAjusteCopiar  
{

"id": 1,

"accountNumber": "12345",

"type": "Personal",

"balance": 1000.0,

"openingDate": "2023-01-01",

"ownerId": 1

* + }

**Error (400 Bad Request)**: Si los datos no cumplen las validaciones.  
 json  
ContraerAjusteCopiar  
{

"error": "El tipo de cuenta debe ser 'Personal' o 'Company'"

* + }

**Error (403 Forbidden)**: Si el usuario autenticado no tiene permiso.  
 json  
ContraerAjusteCopiar  
{

"error": "No tienes permiso para crear cuentas para este usuario"

* + }

### **4. Actualizar una cuenta**

* **Ruta**: /api/v1/default/account/{id}/customer/{owner\_id}
* **Método**: PUT
* **Descripción**: Actualiza una cuenta existente.
* **Parámetros**:
  + id (path): ID de la cuenta (Long, mínimo 1).
  + owner\_id (path): ID del propietario (Long, mínimo 1).

Cuerpo de la solicitud (JSON):  
 json  
ContraerAjusteCopiar  
{

"accountNumber": "12345",

"type": "Personal",

"balance": 1500.0,

"openingDate": "2023-01-01"

* + }
* **Encabezados**:
  + Authorization: Bearer <token> (requerido, token OAuth2).
  + Content-Type: application/json (requerido).
  + Accept: application/json (opcional, para forzar JSON).
* **Retorno**:

**Éxito (200 OK)**: La cuenta actualizada en formato JSON o XML.  
 json  
ContraerAjusteCopiar  
{

"id": 1,

"accountNumber": "12345",

"type": "Personal",

"balance": 1500.0,

"openingDate": "2023-01-01",

"ownerId": 1

* + }

**Error (400 Bad Request)**: Si los datos no cumplen las validaciones.  
 json  
ContraerAjusteCopiar  
{

"error": "El tipo de cuenta debe ser 'Personal' o 'Company'"

* + }

**Error (403 Forbidden)**: Si el usuario autenticado no tiene permiso.  
 json  
ContraerAjusteCopiar  
{

"error": "No tienes permiso para actualizar esta cuenta"

* + }

**Error (404 Not Found)**: Si la cuenta no existe.  
 json  
ContraerAjusteCopiar  
{

"error": "Cuenta no encontrada"

* + }

### **5. Eliminar una cuenta**

* **Ruta**: /api/v1/default/account/{id}/customer/{owner\_id}
* **Método**: DELETE
* **Descripción**: Elimina una cuenta específica.
* **Parámetros**:
  + id (path): ID de la cuenta (Long, mínimo 1).
  + owner\_id (path): ID del propietario (Long, mínimo 1).
* **Encabezados**:
  + Authorization: Bearer <token> (requerido, token OAuth2).
  + Accept: application/json (opcional, para forzar JSON).
* **Retorno**:

**Éxito (200 OK)**: Mensaje de confirmación.  
 json  
ContraerAjusteCopiar  
{

"message": "Eliminada la cuenta del usuario"

* + }

**Error (400 Bad Request)**: Si los parámetros no cumplen las validaciones.  
 json  
ContraerAjusteCopiar  
{

"error": "El ID debe ser mayor o igual a 1"

* + }

**Error (403 Forbidden)**: Si el usuario autenticado no tiene permiso.  
 json  
ContraerAjusteCopiar  
{

"error": "No tienes permiso para eliminar esta cuenta"

* + }

**Error (404 Not Found)**: Si la cuenta no existe.  
 json  
ContraerAjusteCopiar  
{

"error": "Cuenta no encontrada"

* + }

### **6. Depositar dinero en una cuenta**

* **Ruta**: /api/v1/default/account/{id}/deposit/{amount}/customer/{owner\_id}
* **Método**: PUT
* **Descripción**: Añade dinero al balance de una cuenta.
* **Parámetros**:
  + id (path): ID de la cuenta (Long, mínimo 1).
  + amount (path): Cantidad a depositar (int, mínimo 1).
  + owner\_id (path): ID del propietario (Long, mínimo 1).
* **Encabezados**:
  + Authorization: Bearer <token> (requerido, token OAuth2).
  + Accept: application/json (opcional, para forzar JSON).
* **Retorno**:

**Éxito (200 OK)**: La cuenta actualizada en formato JSON o XML.  
 json  
ContraerAjusteCopiar  
{

"id": 1,

"accountNumber": "12345",

"type": "Personal",

"balance": 2000.0,

"openingDate": "2023-01-01",

"ownerId": 1

* + }

**Error (400 Bad Request)**: Si los parámetros no cumplen las validaciones.  
 json  
ContraerAjusteCopiar  
{

"error": "La cantidad debe ser mayor o igual a 1"

* + }

**Error (403 Forbidden)**: Si el usuario autenticado no tiene permiso.  
 json  
ContraerAjusteCopiar  
{

"error": "No tienes permiso para depositar en esta cuenta"

* + }

**Error (404 Not Found)**: Si la cuenta no existe.  
 json  
ContraerAjusteCopiar  
{

"error": "Cuenta no encontrada"

* + }

### **7. Retirar dinero de una cuenta**

* **Ruta**: /api/v1/default/account/{id}/withdraw/{amount}/customer/{owner\_id}
* **Método**: PUT
* **Descripción**: Retira dinero del balance de una cuenta. Si el monto excede el balance y el usuario tiene otras cuentas, se descuenta de ellas (en la capa de persistencia). Si no es posible, se lanza una excepción.
* **Parámetros**:
  + id (path): ID de la cuenta (Long, mínimo 1).
  + amount (path): Cantidad a retirar (int, mínimo 1).
  + owner\_id (path): ID del propietario (Long, mínimo 1).
* **Encabezados**:
  + Authorization: Bearer <token> (requerido, token OAuth2).
  + Accept: application/json (opcional, para forzar JSON).
* **Retorno**:

**Éxito (200 OK)**: La cuenta actualizada en formato JSON o XML.  
 json  
ContraerAjusteCopiar  
{

"id": 1,

"accountNumber": "12345",

"type": "Personal",

"balance": 500.0,

"openingDate": "2023-01-01",

"ownerId": 1

* + }

**Error (400 Bad Request)**: Si los parámetros no cumplen las validaciones o el retiro no es posible.  
 json  
ContraerAjusteCopiar  
{

"error": "Fondos insuficientes para realizar el retiro"

* + }

**Error (403 Forbidden)**: Si el usuario autenticado no tiene permiso.  
 json  
ContraerAjusteCopiar  
{

"error": "No tienes permiso para retirar de esta cuenta"

* + }

**Error (404 Not Found)**: Si la cuenta no existe.  
 json  
ContraerAjusteCopiar  
{

"error": "Cuenta no encontrada"

* + }

### **8. Eliminar todas las cuentas de un usuario**

* **Ruta**: /api/v1/default/accounts/customer/{owner\_id}
* **Método**: DELETE
* **Descripción**: Elimina todas las cuentas asociadas a un usuario.
* **Parámetros**:
  + owner\_id (path): ID del propietario (Long, mínimo 1).
* **Encabezados**:
  + Authorization: Bearer <token> (requerido, token OAuth2).
  + Accept: application/json (opcional, para forzar JSON).
* **Retorno**:

**Éxito (200 OK)**: Mensaje de confirmación.  
 json  
ContraerAjusteCopiar  
{

"message": "Cuentas eliminadas correctamente"

* + }

**Error (400 Bad Request)**: Si el ID no cumple las validaciones.  
 json  
ContraerAjusteCopiar  
{

"error": "El ID debe ser mayor o igual a 1"

* + }

**Error (403 Forbidden)**: Si el usuario autenticado no tiene permiso.  
 json  
ContraerAjusteCopiar  
{

"error": "No tienes permiso para eliminar estas cuentas"

* + }

**Error (404 Not Found)**: Si no hay cuentas para el usuario.  
 json  
ContraerAjusteCopiar  
{

"error": "No se encontraron cuentas para eliminar"

* + }

### **9. Solicitar un préstamo**

* **Ruta**: /api/v1/default/account/customer/{owner\_id}/loan/{amount}
* **Método**: GET
* **Descripción**: Comprueba si un usuario puede recibir un préstamo, que no debe superar el 80% del monto total de sus cuentas.
* **Parámetros**:
  + owner\_id (path): ID del propietario (Long, mínimo 1).
  + amount (path): Cantidad del préstamo (int, mínimo 1).
* **Encabezados**:
  + Authorization: Bearer <token> (requerido, token OAuth2).
  + Accept: application/json (opcional, para forzar JSON).
* **Retorno**:

**Éxito (200 OK)**: Mensaje de confirmación.  
 json  
ContraerAjusteCopiar  
{

"message": "Préstamo aceptado"

* + }

**Error (400 Bad Request)**: Si el préstamo no puede ser concedido.  
 json  
ContraerAjusteCopiar  
{

"error": "Préstamo denegado: el monto excede el 80% del total de las cuentas"

* + }

**Error (403 Forbidden)**: Si el usuario autenticado no tiene permiso.  
 json  
ContraerAjusteCopiar  
{

"error": "No tienes permiso para solicitar un préstamo"

* + }

**Error (404 Not Found)**: Si el usuario no tiene cuentas.  
 json  
ContraerAjusteCopiar  
{

"error": "No se encontraron cuentas para el usuario"

* + }

## **Implementación**

### **Tecnologías**

* **Framework**: Spring Boot.
* **Persistencia**: Spring Data JPA con H2 (perfil dev) y MySQL (perfil prod).
* **Seguridad**: Spring Security con OAuth2 Resource Server.
* **Validación**: Anotaciones de validación (@Valid, @Min, @NotBlank).
* **Documentación**: Springdoc OpenAPI (Swagger UI en /swagger-ui.html).

### **Configuración**

* **Perfiles**:
  + **Dev**: Puerto 9900, base de datos H2, precarga de datos, logs en nivel DEBUG.
  + **Prod**: Puerto 9943, base de datos MySQL, sin precarga de datos, logs en nivel ERROR, ruta base /api/v1.
* **Seguridad**: Todos los endpoints requieren un token OAuth2 válido.
* **Formatos**: Entrada en JSON, salida en JSON o XML según el encabezado Accept.