

Eliminar un Borrador de Factura [BETA]

Funcionalidad en desarrollo y pruebas: este servicio está disponible en el ambiente de pruebas y todavía no debe usarse en producción. El funcionamiento, los campos y los mensajes de error pueden cambiar antes del despliegue definitivo.

Este servicio **elimina un borrador de factura de venta** creado por el consumidor autenticado. Es la operación de compensación cuando el flujo local falla después de crear el borrador: nuby borra la factura y toda su cascada (detalle, impuestos, contabilidad y pagos/recibos) para no dejar documentos huérfanos.

Solo aplica a **facturas de venta en estado borrador** creadas por el propio consumidor. Una factura de contrato, una factura ya emitida o una factura de la UI se rechazan.

¿Para qué sirve este servicio?
Úsalo como **compensación de tu propio flujo**: si después de crear el borrador algo falla en tu plataforma, eliminas el documento para no dejarlo colgado en nuby. Los casos de uso más frecuentes son:

- **Deshacer una creación que no pudo completarse** en tu sistema, después de recibir el `factura_id` del alta.
- **Limpiar borradores de prueba** generados por tus validaciones de integración.
- **Compensar un flujo distribuido fallido**: el borrador ya no va a emitirse ni a usarse.
- **Reintentar sin riesgo**: repetir la misma eliminación devuelve la respuesta original en lugar de un `404`.

1. El Endpoint (La dirección web)

Método y dirección

Dato	Valor
------	-------

Método	DELETE
Ruta	/invoices/{factura_id}
Scope requerido	invoices:update
Autenticación	Authorization: Bearer <jwt>
Cuerpo de la petición	No se envía cuerpo

```
DELETE https://{{{instancia}}}/service/v2/public/invoices/{factura_id}
```

¿Qué debo colocar en {{instancia}}?

Es la **dirección web principal** con la que ingresas a tu plataforma, incluyendo la terminación `.nuby.app` o `.arrendasoft.co`. Si entras a nuby escribiendo `inmobiliaria.nuby.app` en el navegador, esa es tu instancia.

Importante: no incluyas `https://` ni barras diagonales (`/`) al final. La dirección se compone así: `https://` + tu instancia + `/service/v2/public/invoices/` + el identificador de la factura.

Encabezados requeridos

Encabezado	¿Obligatorio?	Valor
Authorization	Sí	Bearer <token>, con el token obtenido en el servicio Login .
Content-Type	No	<code>application/json</code> si tu cliente lo envía. Como la petición no lleva cuerpo, el servicio no lo necesita.

Permiso requerido (scope): invoices:update

Este servicio exige el permiso `invoices:update` en el cliente OAuth. Si el token es válido pero al cliente le falta el permiso, la respuesta es `403` con el código `RECURSO_NO_AUTORIZADO`.

Si aún no tienes un token, consulta la página [Login](#) del libro **API nuby V2**. Allí se explica cómo identificarte y cómo se construye el encabezado `Authorization`.

Qué necesitas

- El `factura_id` del borrador que quieres eliminar.

- Un token con el scope `invoices:update`.
- Que la factura haya sido creada por tu consumidor y siga en estado `5` (Borrador).

No hay soft delete. La eliminación es física e irreversible. La trazabilidad de la operación queda en `api_solicitudes` y en los logs de nuby.

2. La Petición (¿Qué debes enviarnos?)

Path parameter

Campo	Tipo	Requerido	Descripción
<code>factura_id</code>	entero > 0	Sí	Identificador interno de la factura en nuby. Va en la dirección. Si no es un entero positivo, la respuesta es <code>400</code> con el mensaje <code>Debe indicar un factura_id entero mayor que 0 para operar sobre la factura.</code>

Body

No se envía cuerpo. Tampoco se envía `referencia_externa`: la idempotencia usa la clave interna `eliminar:{factura_id}`.

Orden de evaluación

1. Scope `invoices:update`.
2. `factura_id` válido.
3. Idempotencia (replay si ya se eliminó).
4. Guard de contrato.
5. Existencia de la factura.
6. Pertenencia al consumidor.
7. Estado `5` (Borrador).

3. La Respuesta (¿Qué te entregaremos?)

Respuesta exitosa

```
{
  "success": true,
  "status": 200,
```

```
"message": "La factura de venta se eliminó correctamente.",
"body": {
  "factura_id": 10231,
  "eliminada": true,
  "idempotent_replay": false
}
}
```

Estructura principal de la respuesta

Campo	Tipo	Descripción
success	booleano	true cuando la eliminación se completó.
status	entero	Replica el código HTTP de la respuesta: 200 en el éxito y también en el reintento idempotente.
message	texto	Mensaje legible del resultado.
body	objeto	Identificador de la factura eliminada e indicador del reintento. En un error, este campo no existe.

Las respuestas de escritura **no** traen pagination ni filters_applied: esos campos pertenecen a las consultas de lectura.

Reintento idempotente

Si repites el DELETE de una factura que ya eliminaste, nuby devuelve la respuesta original con idempotent_replay: true. No intenta borrar dos veces ni responde 404.

Campos de la respuesta

Campo	Tipo	Descripción
factura_id	entero	Factura eliminada.
eliminada	booleano	true si la eliminación se completó.
idempotent_replay	booleano	true cuando la respuesta viene de un reintento idempotente.

Escenario alternativo: el reintento

La segunda llamada sobre el mismo factura_id responde 200 con el body de la respuesta original y el indicador en true:

```
{
  "factura_id": 10231,
  "eliminada": true,
  "idempotent_replay": true
}
```

4. Idempotencia y reintentos

Este servicio no usa `referencia_externa`. La clave de idempotencia es `eliminar:{factura_id}`, registrada en `api_solicitudes` con el recurso `factura` y la operación `eliminar`.

Situación	Qué responde nuby
Reintento de una eliminación ya completada	<code>200</code> con la respuesta original y <code>idempotent_replay: true</code> .
Otra solicitud con la misma clave en curso	<code>409 IDEMPOTENCY_IN_PROGRESS</code> .
La fila de <code>api_solicitudes</code> ya fue purgada por retención	<code>404 FACTURA_NO_ENCONTRADA</code> en lugar del replay.

Reintenta sin miedo. Mientras la solicitud siga registrada, repetir el `DELETE` es seguro: nunca borra dos veces y nunca responde `404` por un borrado previo exitoso.

5. Seguridad y Posibles Errores

Tu pase tiene fecha de caducidad

Por medidas de seguridad, el token que te entregamos **solo dura 1 hora**. Una vez transcurrido ese tiempo, el sistema te bloquea el acceso con un error `401` y tu sistema debe volver a consumir el servicio de **Login** para pedir un pase nuevo.

Esta respuesta la produce el **middleware de autenticación** y no el contrato de la operación, así que usa el envelope heredado (`statusCode` y `error.description`), no `success` ni `error_code`:

Situación	HTTP	Detalle
No se envió el encabezado <code>Authorization</code>	<code>401</code>	<code>JWT Token required.</code>
El encabezado no tiene el formato <code>Bearer {token}</code>	<code>400</code>	<code>JWT Token not send.</code>

Situación	HTTP	Detalle
El token no fue encontrado en el sistema	401	JWT Token not found.
El token expiró	401	JWT Token expired.

Errores del contrato de la operación

Los errores de la operación traen `success: false`, el código HTTP, un `message` legible y un `error_code` estable:

HTTP	error_code	Cuándo
400	VALIDATION_ERROR	El <code>factura_id</code> del path no es un entero positivo.
403	CONSUMIDOR_NO_AUTENTICADO	El token es válido, pero no corresponde a un cliente OAuth habilitado para consumir la API.
403	RECURSO_NO_AUTORIZADO	El consumidor no tiene <code>invoices:update</code> .
403	RECURSO_AJENO	La factura fue creada por la UI u otro consumidor.
404	FACTURA_NO_ENCONTRADA	La factura no existe.
409	FACTURA_CONTRATO_NO_SOPORTADA	La factura es de contrato; se gestiona en el flujo de contratos.
409	FACTURA_NO_ELIMINABLE	La factura no está en estado <code>5</code> (Borrador). El mensaje incluye el estado actual.
409	IDEMPOTENCY_IN_PROGRESS	Otra solicitud con la misma clave está en proceso.
500	INTERNAL_ERROR	Fallo técnico; el detalle queda en el log <code>API_INTEGRACION</code> .

Forma del rechazo de contrato

El rechazo por contrato llega con el código `FACTURA_CONTRATO_NO_SOPORTADA`, un mensaje que explica el motivo y una advertencia que identifica la factura y el alcance del endpoint:

Campo de <code>warnings[]</code>	Tipo	Significado
<code>code</code>	texto	Siempre <code>FACTURA_CONTRATO_NO_SOPORTADA</code> . Es el valor que debes evaluar en tu código.

Campo de <code>warnings[]</code>	Tipo	Significado
<code>factura_id</code>	entero	Identificador de la factura rechazada.
<code>scope</code>	texto	Alcance del endpoint: confirma que el rechazo es por tipo de factura, no por permisos.

Trata el `404` como un caso de negocio, no como una falla técnica: significa que el identificador no existe en nuby (o que nunca lo creó tu consumidor). No lo reintentes con el mismo valor.

Trata el `409 FACTURA_NO_ELIMINABLE` como un caso de negocio: la factura ya salió de borrador y no se elimina por esta vía. Revisa el estado actual en el mensaje antes de decidir qué hacer.

6. Ejemplos de integración

Aquí tienes ejemplos listos para adaptar a tu plataforma. Todos usan `{{instancia}}`, `Authorization: Bearer`, eliminan un borrador y tratan el éxito junto con los rechazos de negocio más frecuentes (`404` y `409`).

cURL

Eliminar un borrador

```
TOKEN="TU_TOKEN_AQUI"
INSTANCIA="mi-inmobiliaria.nuby.app"
FACTURA_ID=10231

curl -X DELETE "https://${INSTANCIA}/service/v2/public/invoices/${FACTURA_ID}" \
-H "Authorization: Bearer ${TOKEN}"
```

Reintentar la misma eliminación

```
curl -X DELETE "https://${INSTANCIA}/service/v2/public/invoices/${FACTURA_ID}" \
-H "Authorization: Bearer ${TOKEN}"
```

La segunda llamada devuelve `200` con `idempotent_replay: true`.

PHP

```
<?php
$instancia = 'mi-inmobiliaria.nuby.app'; // Tu dirección web completa
$token      = 'TU_TOKEN_AQUI';           // Token obtenido del servicio Login
$facturaId = 10231;

$ch = curl_init("https://{$instancia}/service/v2/public/invoices/{$facturaId}");
curl_setopt($ch, CURLOPT_CUSTOMREQUEST, 'DELETE');
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_HTTPHEADER, [
    "Authorization: Bearer {$token}",
]);

$respuesta = curl_exec($ch);
$httpCode  = curl_getinfo($ch, CURLINFO_HTTP_CODE);
curl_close($ch);

$info = json_decode($respuesta, true);

if ($httpCode === 200) {
    echo "Factura eliminada: {$info['body']['factura_id']}\n";
    echo "Reintento idempotente: " . var_export($info['body']['idempotent_replay'], true)
    . "\n";
} elseif ($httpCode === 404 || $httpCode === 409) {
    echo "Rechazo de negocio {$info['error_code']}: {$info['message']}\n";
} else {
    echo "Error HTTP {$httpCode}: {$info['message']}\n";
}
```

Python

```
import requests

# 1. Configura tus datos de acceso
instancia = "mi-inmobiliaria.nuby.app" # Reemplaza con tu dirección web completa
token = "TU_TOKEN_AQUI" # Token obtenido del servicio Login
factura_id = 10231

# 2. Envía la petición
url = f"https://{instancia}/service/v2/public/invoices/{factura_id}"
headers = {"Authorization": f"Bearer {token}"}

respuesta = requests.delete(url, headers=headers, timeout=30)

# 3. Procesa la respuesta
if respuesta.status_code == 200:
    body = respuesta.json()["body"]
    print(f"Factura {body['factura_id']} eliminada: {body['eliminada']}")
    print(f"Reintento idempotente: {body['idempotent_replay']}")
elif respuesta.status_code in (404, 409):
    error = respuesta.json()
    print(f"Rechazo de negocio {error['error_code']}: {error['message']}")
else:
    print(f"Error HTTP {respuesta.status_code}: {respuesta.text}")
```

JavaScript (Fetch API)

```
// 1. Configura tus datos de acceso
const instancia = 'mi-inmobiliaria.nuby.app'; // Reemplaza con tu dirección web completa
const token = 'TU_TOKEN_AQUI'; // Token obtenido del servicio Login

// 2. Función que elimina el borrador
async function eliminarBorrador(facturaId) {
    const url = `https://${instancia}/service/v2/public/invoices/${facturaId}`;
```

```

const response = await fetch(url, {
  method: 'DELETE',
  headers: {
    'Authorization': `Bearer ${token}`,
  },
});

const info = await response.json().catch(() => ({}));

if (!response.ok) {
  console.error(`Error HTTP ${response.status} ${info.error_code ?? ''}: ${info.message ?? ''}`);
  return null;
}

return info.body;
}

// 3. Elimina el borrador y revisa si fue un reintento
eliminarBorrador(10231).then((body) => {
  if (!body) {
    return;
  }

  console.log(`Factura ${body.factura_id} eliminada: ${body.eliminada}`);
  console.log(`Reintento idempotente: ${body.idempotent_replay}`);
});

```

7. Buenas prácticas y límites conocidos

Recomendaciones

- **Elimina solo cuando el flujo local falló.** Si la creación se completó, conserva el borrador para editarlo o emitirlo.
- **No reutilices la referencia de creación.** El DELETE usa su propia clave; no necesitas enviar `referencia_externa`.
- **Trata el `200` con `idempotent_replay: true` como éxito.** Significa que la factura ya estaba eliminada por tu consumidor.

- **No uses DELETE para anular una factura emitida.** La anulación tiene su propio flujo; el DELETE solo acepta borradores.
- **Registra el `factura_id` y el resultado.** La operación es irreversible y la trazabilidad queda en `api_solicitudes`.

Límites de este servicio

- **Borrado físico e irreversible.** No hay soft delete: no existe una operación para recuperar la factura eliminada.
- **Solo borradores.** Una factura en estado distinto de `5` se rechaza con `409 FACTURA_NO_ELIMINABLE`.
- **Solo facturas propias.** Una factura creada por la UI u otro consumidor se rechaza con `403 RECURSO_AJENO`.
- **No anula ni emite.** Este servicio no reemplaza el flujo de anulación ni el de emisión de la factura.
- **Depende de la retención de `api_solicitudes`.** Si la solicitud ya fue purgada, un `DELETE` repetido responde `404` en lugar del replay.

Revisión #1

Creado el 28 octubre 2026 14:11:21 por Jorge Iván González

Actualizado el 28 octubre 2026 14:12:07 por Jorge Iván González