

Editar Factura en Borrador

[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 **modifica una factura de venta que todavía está en borrador** en nuby. Puedes cambiar la cabecera y los ítems; nuby vuelve a calcular los totales, el saldo y la contabilidad del documento. La factura sigue en estado `5` (Borrador) y sigue **sin numerar**: la edición no consume consecutivo.

Solo puedes editar **tus** borradores: los que creó tu consumidor por API. Una factura generada desde un contrato de arrendamiento, una factura ya emitida o un borrador creado por la interfaz de nuby quedan fuera del alcance de este servicio. La emisión es un paso aparte: editar no envía nada a la DIAN.

¿Para qué sirve este servicio?

Sirve para **corregir un borrador antes de emitirlo**. La creación y la emisión son actos separados justamente para que exista este paso: entre una y otra puedes ajustar el documento tantas veces como necesites, sin gastar numeración y sin enviar nada a la DIAN.

Casos de uso concretos

- **Revisión humana antes de emitir.** Creas el borrador por API, un contador lo revisa en la interfaz de nuby y tu sistema aplica el ajuste final con este servicio.
- **Corregir precios, cantidades o impuestos.** Si el pedido cambió después de crear el borrador, reenvías los ítems completos y nuby recalcula totales y saldo.
- **Cambiar la fecha o el plazo.** Ajustas `fecha`, `dias_vencimiento` o `fecha_vencimiento` para que coincidan con la operación real.
- **Cambiar el tercero.** Si facturaste a la persona equivocada, envías otro `tercero_id` —o el objeto `tercero`— y nuby vuelve a escribir el documento.
- **Completar datos comerciales.** Agregas `observaciones`, `vendedor_id`, `centro_id` o las `formas_de_pago` que faltaban.

- **Aplicar anticipos.** El campo `anticipos[]` también se acepta en la edición, así que puedes aplicar un anticipo que no estaba disponible al crear el borrador.

Resumen del servicio

Aspecto	Valor
Método y ruta	<code>PUT PATCH /service/v2/public/invoices/{factura_id}</code>
Permiso (scope) requerido	<code>invoices:update</code>
Respuesta exitosa	<code>200</code> con <code>factura_id</code> , <code>estado_id</code> , <code>consecutivo</code> , <code>valor_total</code> y <code>valor_saldo</code>
Efecto del documento	Sigue en estado <code>5</code> (Borrador), sin numerar, con totales, saldo y contabilidad recalculados
Protección contra duplicados	Idempotencia por <code>referencia_externa</code> + consumidor + factura + contenido
¿Envía a la DIAN?	No. El envío es la operación de emisión

Lo que este servicio NO hace

- **No numera ni emite.** El consecutivo real y el envío a la DIAN son de la operación de emisión.
- **No acepta cambios parciales.** `PUT` y `PATCH` comparten comportamiento: el payload describe el documento completo y nuby lo vuelve a grabar. Un campo que omitas no se conserva "por defecto": se valida como faltante.
- **No cambia el documento de destino.** El `factura_id` del path manda: un `factura_id` en el cuerpo nunca decide qué factura se edita.
- **No toca facturas fuera de borrador.** Una factura emitida responde `409 FACTURA_NO_EDITABLE`.
- **No toca facturas de contrato.** El alcance se evalúa primero, sin importar quién la creó.
- **No revalida el período contable ni la facturación electrónica.** Ambas cosas se validaron al crear el borrador.
- **No permite editar la factura de otro consumidor.** La pertenencia se evalúa antes del estado.

Dónde encaja en el ciclo

1. Crear borrador	<code>POST /invoices</code>	(guía anterior)
2. Editar borrador	<code>PUT PATCH /invoices/{factura_id}</code>	(esta página)
3. Emitir a la DIAN	<code>POST /invoices/{factura_id}/issue</code>	
4. Consultar detalle	<code>GET /invoices/{factura_id}</code>	
5. Obtener el PDF	<code>GET /invoices/{factura_id}/pdf</code>	



Si todavía no tienes un borrador, créalo primero con **Crear Factura en Borrador**. El `factura_id` que devuelve esa operación es el que va en el path de esta.

1. El Endpoint (La dirección web)

Apunta tu sistema a la siguiente dirección. Puedes usar **PUT** o **PATCH**: las dos llegan al mismo servicio y tienen la misma semántica (documento completo).

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

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

¿Qué debo colocar en `{instancia}`?

La **dirección web principal** que utilizas a diario para ingresar a tu plataforma, incluyendo la terminación `.nuby.app` o `.arrendasoft.co`. Por ejemplo `inmobiliaria.nuby.app`.

Importante: no incluyas `https://` ni barras diagonales (`/`) al final. La dirección se compone así: `https://` + tu instancia + `/service/v2/public/invoices/{factura_id}`.

¿Qué debes colocar en `{factura_id}`? El identificador interno que devolvió la creación (campo `body.factura_id`). Debe ser un **entero mayor que 0**. Si no lo es, la respuesta es `400` con *Debe indicar un factura_id entero mayor que 0 para operar sobre la factura*. **No** construyas ese valor a partir del consecutivo: en un borrador el consecutivo es un marcador sin numerar.

Encabezados requeridos

Encabezado	Valor	Obligatorio
<code>Content-Type</code>	<code>application/json</code>	Sí
<code>Authorization</code>	<code>Bearer <token></code> , con el token obtenido en el servicio Login	Sí

Permiso requerido (scope)

El cliente OAuth con el que se genera el token debe tener habilitado el permiso `invoices:update`. Ojo: es distinto del permiso de creación (`invoices:create`). Un cliente con permiso para crear no necesariamente puede editar.

- `403 CONSUMIDOR_NO_AUTENTICADO` → el token es válido pero no corresponde a un cliente OAuth habilitado. Mensaje: *El cliente autenticado no está autorizado para consumir este endpoint.*
- `403 RECURSO_NO_AUTORIZADO` → al cliente le falta `invoices:update`. Mensaje: *El consumidor no tiene habilitado el recurso solicitado.*

Si aún no tienes un token, consulta la página [Login](#) del libro **API nuby V2**.

Precondiciones del documento

Precondición	Qué pasa si no se cumple
La factura es de venta (no nació de un contrato de arrendamiento).	<code>409 FACTURA_CONTRATO_NO_SOPORTADA</code> con <code>warnings[]</code> . No se evalúa el estado ni la pertenencia.
La factura la creó tu consumidor por API.	<code>403 RECURSO_AJENO</code> .
La factura está en borrador (<code>estado_id = 5</code>).	<code>409 FACTURA_NO_EDITABLE</code> con el estado actual en el mensaje.
El payload cumple el contrato de entrada (mismos campos obligatorios que la creación).	<code>422 VALIDATION_ERROR</code> con <code>errors</code> por campo.

La edición no revalida la configuración del sistema. Ni `ENABLE_FACT_ELECT` ni el período contable de la fecha se vuelven a evaluar aquí: se validaron al crear el borrador y la fecha del documento no cambió de período por editarse.

2. La Petición (¿Qué debes enviarnos?)

El cuerpo es un **objeto JSON** con el mismo contrato de entrada de la creación: la referencia de idempotencia de **esta edición** y, al mismo nivel, la cabecera y los ítems del documento.

Elemento	Valor
Método	<code>PUT</code> o <code>PATCH</code> (equivalen)
<code>Content-Type</code>	<code>application/json</code>
<code>Authorization</code>	<code>Bearer <token></code>

Elemento	Valor
Cuerpo	Objeto JSON con la cabecera y el arreglo <code>items</code>

Cabecera de la factura

Parámetro	Tipo	Requerido	Por defecto	Descripción
<code>referencia_externa</code>	texto	Sí	—	Identificador único de esta edición . Si falta: <code>422</code> con <i>El campo "referencia_externa" es obligatorio</i> . No reutilices la referencia de la creación: esa ya quedó consumida por otra operación.
<code>tercero_id</code>	entero	Sí, o el objeto <code>tercero</code>	—	Tercero de la factura. Si envías <code>tercero_id</code> y <code>tercero</code> , gana <code>tercero_id</code> .
<code>tercero</code>	objeto	Sí, o <code>tercero_id</code>	—	Datos del tercero para resolverlo o crearlo en la misma operación. Sin ninguno de los dos: <i>Debe indicar <code>tercero_id</code> o el objeto <code>tercero</code>.</i>
<code>fecha</code>	texto <code>AAAA-MM-DD</code>	Sí	—	Fecha de la factura. Debes enviarla aunque no la cambies : el payload es el estado completo del documento. Formato inválido: <i>La fecha de la factura es obligatoria y debe tener el formato AAAA-MM-DD.</i>
<code>dias_vencimiento</code>	entero	Sí, salvo que uses <code>fecha_vencimiento</code> o <code>pago_total: true</code>	—	Entero mayor o igual a 1 . Con <code>pago_total: true</code> se ignora.

Parámetro	Tipo	Requerido	Por defecto	Descripción
fecha_vencimiento	texto AAAA-MM-DD	No	—	Alternativa a días_vencimiento: nuby deriva los días. Si envías ambas y se contradicen: <i>La fecha de vencimiento no coincide con los días de vencimiento enviados.</i>
resolucion_id	entero	No	la resolución que ya tiene la factura	Si lo omites, nuby conserva la resolución del documento. Si lo envías, debe existir, estar activa y ser de venta. Cambiarla reescribe la resolución del documento: hazlo solo si sabes lo que implica.
moneda	texto	No	COP	Moneda del documento.
tasa_cambio	numérico	No	null	Tasa de cambio aplicable. Solo tiene sentido con moneda extranjera.
vendedor_id	entero	No	null	Usuario de nuby que figura como vendedor.
centro_id	entero	No	null	Centro de costo del documento.
observaciones	texto	No	null	Nota libre del documento.
pago_total	booleano	No	false	Si es verdadero, la factura queda de contado y días_vencimiento se ignora.
formas_de_pago	arreglo de objetos	Sí, cuando pago_total es true	[]	Con pago_total: true debes enviar al menos una: <i>Debe agregar al menos una forma de pago cuando pago_total es verdadero.</i>

Parámetro	Tipo	Requerido	Por defecto	Descripción
<code>anticipos[]</code>	arreglo	No	<code>[]</code>	Anticipos a aplicar. Acepta objetos (<code>[{"documento_id": 5}]</code>) o una lista plana (<code>[5]</code>).
<code>items</code>	arreglo de objetos	Sí	—	Líneas del documento. Al menos una: <i>Debe enviar al menos un ítem en items.</i> nuby reemplaza el detalle con lo que envíes.

Ítems (`items[]`)

La edición reutiliza el mismo contrato de ítem de la creación: `producto_id`, `cantidad` y `valor_unitario` son obligatorios y se validan todos antes de tocar el documento; el resto de campos (`descuentos`, `impuestos`, `retenciones`, `descripcion`, `descuentos_id`, `otros_impuestos`, `tercero_aplica_id`, `valor_unitario_ori`) son opcionales. Si un ítem es inválido, la factura **no se modifica**.

- `producto_id` → entero mayor que 0. *producto_id debe ser un entero mayor que 0.*
- `cantidad` → entero mayor que 0. *cantidad debe ser un entero mayor que 0.*
- `valor_unitario` → numérico y distinto de cero. *valor_unitario debe ser numérico y distinto de 0.*

El detalle se reemplaza, no se mezcla. Envía todas las líneas que debe tener el documento. Una línea que no envíes deja de existir en la factura, y los totales se recalculan con las líneas enviadas.

El `factura_id` del path manda

Si tu cuerpo trae un `factura_id`, nuby lo descarta: la factura que se edita es la del path. Es una protección deliberada contra errores de tu integración, porque un identificador equivocado en el cuerpo cambiaría el documento equivocado.

```
{
  "referencia_externa": "INV-2026-0001-EDIT-1",
  "tercero_id": 4821,
  "fecha": "2026-10-21",
  "dias_vencimiento": 15,
```

```
"items": [  
  { "producto_id": 340, "descripcion": "Administración octubre 2026", "cantidad": 2,  
    "valor_unitario": 1500 }  
]  
}
```

Enviado a `PUT /invoices/10231`, edita la factura `10231` y la respuesta también dice `factura_id: 10231`.

Campos que siguen prohibidos

La edición valida los mismos guardas que la creación, en este orden:

Grupo	Campos	Respuesta
Datos de contrato o movimiento	<code>movimiento_id</code> , <code>contrato_id</code> , <code>procesado</code> , <code>documento_respaldo_id</code> , <code>inmobiliaria</code>	<code>422 FACTURA_CONTRATO_NO_PERMITIDA</code> con <code>warnings[]</code> . Enviarlos en <code>null</code> , <code>""</code> o <code>0</code> no es un error.
Campos protegidos	<code>estado_id</code> , <code>consecutivo</code> , <code>cufe</code> , <code>dian_id</code> , <code>estado_dian</code> , <code>creado_por</code> , <code>tipo_factura</code>	<code>422 CAMPO_NO_PERMITIDO</code> . Se evalúa la presencia de la clave: enviarla en <code>null</code> también se rechaza.

```
{  
  "success": false,  
  "status": 422,  
  "message": "El payload intenta fijar el campo estado_id, que solo el sistema puede asignar  
en una factura de venta.",  
  "error_code": "CAMPO_NO_PERMITIDO",  
  "errors": {  
    "estado_id": [  
      "El campo estado_id solo lo puede asignar el sistema."  
    ]  
  }  
}
```

Ejemplos de petición

Cambiar plazo y observaciones (documento completo)

```
PUT /service/v2/public/invoices/10231  
Authorization: Bearer <token>  
Content-Type: application/json
```

```
{
  "referencia_externa": "INV-2026-0001-EDIT-1",
  "tercero_id": 4821,
  "fecha": "2026-10-21",
  "dias_vencimiento": 15,
  "moneda": "COP",
  "observaciones": "Administración octubre 2026 (ajustada)",
  "pago_total": false,
  "items": [
    {
      "producto_id": 340,
      "descripcion": "Administración octubre 2026",
      "cantidad": 1,
      "valor_unitario": 1200000,
      "por_iva": 19,
      "iva": 228000
    }
  ]
}
```

Reemplazar el detalle con dos líneas (PATCH)

```
{
  "referencia_externa": "INV-2026-0001-EDIT-2",
  "tercero_id": 4821,
  "fecha": "2026-10-22",
  "fecha_vencimiento": "2026-11-21",
  "items": [
    { "producto_id": 340, "descripcion": "Administración octubre 2026", "cantidad": 1,
"valor_unitario": 1000000 },
    { "producto_id": 512, "descripcion": "Servicio de aseo octubre 2026", "cantidad": 2,
"valor_unitario": 150000 }
  ]
}
```

Pasar el documento a contado

```
{
  "referencia_externa": "INV-2026-0001-EDIT-3",
  "tercero_id": 4821,
  "fecha": "2026-10-21",
```

```
"pago_total": true,
"formas_de_pago": [
  { "tipo_pago_id": 1, "valor": 1190000 }
],
"items": [
  { "producto_id": 340, "cantidad": 1, "valor_unitario": 1000000, "por_iva": 19, "iva":
190000 }
]
}
```

3. La Respuesta (¿Qué te entregaremos?)

Si la edición es válida, nuby regrabó el documento, recalcula totales y saldo, vuelve a escribir la contabilidad y responde `200`:

```
{
  "success": true,
  "status": 200,
  "message": "La factura de venta se actualizó correctamente.",
  "body": {
    "factura_id": 10231,
    "estado_id": 5,
    "consecutivo": -1,
    "prefijo": "FE",
    "valor_total": 3570,
    "valor_saldo": 3570
  },
  "idempotent_replay": false
}
```

Estructura principal de la respuesta

Campo	Tipo	Descripción
<code>success</code>	booleano	<code>true</code> cuando la edición se aplicó.
<code>status</code>	entero	Código HTTP replicado en el cuerpo. Aquí siempre <code>200</code> .
<code>message</code>	texto	<i>La factura de venta se actualizó correctamente.</i>

Campo	Tipo	Descripción
<code>body</code>	objeto	Estado del documento después de la edición (tabla siguiente).
<code>idempotent_replay</code>	booleano	<code>false</code> en una edición nueva; <code>true</code> cuando la respuesta se recuperó de una solicitud anterior. En un replay la marca también aparece dentro de <code>body</code> .

Campos de `body`

Campo	Tipo	Descripción
<code>factura_id</code>	entero	Identificador de la factura editada: siempre el del path.
<code>estado_id</code>	entero	Estado del documento después de editar. Aquí <code>5</code> (Borrador).
<code>consecutivo</code>	entero	Marcador de "sin numerar" (cero o negativo). La edición no consume consecutivo real.
<code>valor_total</code>	numérico	Total recalculado por nuby con los ítems que acabas de enviar.
<code>valor_saldo</code>	numérico	Saldo recalculado. Es el valor que se tendrá en cuenta al emitir.

No des por hecho el valor de `consecutivo`. Compara siempre contra `estado_id` y contra la presencia de `cufe`: mientras `estado_id = 5` el documento no está numerado y su `consecutivo` es un marcador interno que no debes mostrar al cliente final.

Qué se recalcula exactamente

- **Detalle:** se reemplaza con los `items` enviados.
- **Totales y saldo:** nuby recalcula `valor_total` y `valor_saldo` con los impuestos, descuentos y retenciones del detalle.
- **Contabilidad:** el documento contable se vuelve a escribir con el nuevo detalle.
- **Resolución:** se conserva la que tenía la factura cuando no envías `resolucion_id`.

4. Idempotencia y reintentos

Reintento de una edición ya completada (replay)

Si repites la petición con la **misma** `referencia_externa`, la misma factura y el mismo contenido, nuby devuelve la respuesta guardada con `idempotent_replay: true`. No se vuelve a escribir el documento:

```
{
  "success": true,
  "status": 200,
  "message": "La factura de venta se actualizó correctamente.",
  "body": {
    "factura_id": 10231,
    "estado_id": 5,
    "consecutivo": -1,
    "prefijo": "FE",
    "valor_total": 3570,
    "valor_saldo": 3570
  },
  "idempotent_replay": true
}
```

La clave de idempotencia incluye el `factura_id`: la misma referencia aplicada a **otra** factura responde `409 IDEMPOTENCY_PAYLOAD_MISMATCH`, no un replay. Y la misma referencia con otro contenido también es `409`: *La referencia externa ya fue usada con un payload distinto*.

Reglas de reintento

Situación	Qué hace nuby con la referencia	Qué debes hacer
Rechazo <code>422</code> de negocio	Libera la referencia	Corrige el payload y reintenta con la misma referencia.
<code>409 FACTURA_NO_EDITABLE</code>	No consume la referencia	No insistas: la factura ya no es editable. Consulta su estado con Ver Detalle de una Factura .
Fallo <code>500</code>	Deja la solicitud fallida	Reintenta con la misma referencia.
Éxito <code>200</code>	Deja la solicitud completada	Una edición nueva necesita una referencia nueva.

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 pase expira y el sistema bloquea el acceso. Cuando esto ocurra, tu sistema simplemente debe volver a consumir el servicio de **Login** para pedir un pase nuevo y reintentar con la misma `referencia_externa`.

```
{
  "statusCode": 401,
  "error": {
    "type": "SERVER_ERROR",
    "description": "JWT Token expired."
  }
}
```

Ese `401` lo emite el **middleware de autenticación**, no el servicio de facturas, así que su forma es distinta: no trae `success` ni `error_code`.

Orden de evaluación: es contrato, no detalle interno

nuby evalúa en este orden exacto. Importa porque determina **qué** error recibes cuando hay varios problemas a la vez:

#	Qué se evalúa	Rechazo
1	Scope del consumidor	<code>403 RECURSO_NO_AUTORIZADO</code> / <code>403 CONSUMIDOR_NO_AUTENTICADO</code>
2	El <code>factura_id</code> del path es un entero positivo	<code>400 VALIDATION_ERROR</code>
3	El cuerpo es un objeto JSON	<code>400 VALIDATION_ERROR</code>
4	Alcance: ¿es una factura de contrato?	<code>409 FACTURA_CONTRATO_NO_SOPORTADA</code>
5	Pertenencia: ¿la creó tu consumidor?	<code>403 RECURSO_AJENO</code>
6	<code>referencia_externa</code> presente	<code>422 VALIDATION_ERROR</code>
7	Idempotencia de la solicitud	<code>409 IDEMPOTENCY_IN_PROGRESS</code> / <code>409 IDEMPOTENCY_PAYLOAD_MISMATCH</code>
8	Estado: la factura sigue en borrador	<code>409 FACTURA_NO_EDITABLE</code>
9	Validación del payload	<code>422 (VALIDATION_ERROR, CAMPO_NO_PERMITIDO, FACTURA_CONTRATO_NO_PERMITIDA, ...)</code>



El alcance gana sobre la pertenencia y sobre el estado. Una factura de contrato responde `409 FACTURA_CONTRATO_NO_SOPORTADA` aunque además sea de otro consumidor o esté fuera de borrador: nuby no evalúa nada más. Y el estado gana sobre la validación: una factura ya emitida responde `409` aunque tu payload también tenga errores.

Tabla de códigos HTTP, causas y mensajes

HTTP	error_code	Causa y mensaje exacto
400	VALIDATION_ERROR	El <code>factura_id</code> del path no es un entero positivo. <i>Debe indicar un factura_id entero mayor que 0 para operar sobre la factura.</i>
400	VALIDATION_ERROR	El cuerpo no es un objeto JSON. <i>El cuerpo de la solicitud debe ser un objeto JSON.</i>
400	—	El encabezado <code>Authorization</code> no tiene el formato <code>Bearer {token}</code> . <i>JWT Token not send.</i>
401	—	Falta el encabezado <code>Authorization</code> : <i>JWT Token required.</i>
401	—	El token no existe: <i>JWT Token not found.</i>
401	—	El token expiró: <i>JWT Token expired.</i> Vuelve a consumir Login .
403	CONSUMIDOR_NO_AUTENTICADO	El token no corresponde a un cliente OAuth habilitado. <i>El cliente autenticado no está autorizado para consumir este endpoint.</i>
403	RECURSO_NO_AUTORIZADO	Al cliente le falta <code>invoices:update</code> . <i>El consumidor no tiene habilitado el recurso solicitado.</i>
403	RECURSO_AJENO	La factura la creó la interfaz de nuby u otro consumidor. <i>La factura indicada no pertenece al consumidor autenticado.</i>
404	FACTURA_NO_ENCONTRADA	nuby tiene registrada la pertenencia de ese identificador pero la factura ya no existe. <i>La factura {factura_id} no existe.</i>

HTTP	error_code	Causa y mensaje exacto
409	FACTURA_CONTRATO_NO_SOPORTADA	La factura nació de un contrato de arrendamiento. Incluye <code>warnings[]</code> con <code>factura_id</code> . Mensaje: <i>La factura {factura_id} fue generada desde un contrato de arrendamiento. Esta API solo admite facturas de venta; las facturas de contrato se gestionan en el flujo de contratos de nuby.</i>
409	FACTURA_NO_EDITABLE	La factura no está en borrador. Mensaje: <i>La factura {factura_id} no está en borrador (estado {estado_id}). Solo se pueden editar facturas en estado borrador.</i>
409	IDEMPOTENCY_IN_PROGRESS	Hay una solicitud en curso con esa referencia. <i>La solicitud con esta referencia externa está en proceso.</i>
409	IDEMPOTENCY_PAYLOAD_MISMATCH	La referencia ya se usó con otro contenido o sobre otra factura. <i>La referencia externa ya fue usada con un payload distinto.</i>
422	VALIDATION_ERROR	Falta <code>referencia_externa</code> (El campo "referencia_externa" es obligatorio.) o el payload no cumple el formato (El payload no es válido.). Trae <code>errors</code> por campo.
422	FACTURA_CONTRATO_NO_PERMITIDA	El payload trae datos de contrato con valor. Incluye <code>warnings[]</code> . Mensaje: <i>El payload intenta crear o modificar una factura con datos de contrato ({campo}). Esta API solo crea facturas de venta; las facturas de contrato se generan desde el flujo de contratos.</i>
422	CAMPO_NO_PERMITIDO	El payload intenta fijar un campo del sistema. <i>El payload intenta fijar el campo {campo}, que solo el sistema puede asignar en una factura de venta.</i>
422	RESOLUCION_CONTRATO_NO_PERMITIDA	El <code>resolucion_id</code> enviado es de contrato. <i>La resolución {resolucion_id} se usa para facturas de contrato. Esta API solo admite resoluciones de facturación de venta.</i>
422	RESOLUCION_NO_DISPONIBLE	La resolución enviada no existe o no está activa para facturación. <i>No hay una resolución de facturación de venta activa disponible para crear la factura.</i>

HTTP	error_code	Causa y mensaje exacto
422	DATOS_FACTURA_INVALIDOS	El motor de facturación rechazó el documento. El <code>message</code> trae la causa concreta (por ejemplo, datos fiscales incompletos del tercero).
500	INTERNAL_ERROR	Fallo técnico, con mensaje genérico a propósito: el detalle queda en el log de nuby.
500	IDEMPOTENCY_STORAGE_ERROR	No se pudo registrar la solicitud para controlar duplicados. <i>No se pudo registrar la solicitud de idempotencia.</i>

Forma exacta de `warnings[]` al rechazar una factura de contrato

Es un arreglo con un solo elemento, y su forma es fija: `code`, `factura_id` y `scope`.

```
{
  "success": false,
  "status": 409,
  "message": "La factura 990 fue generada desde un contrato de arrendamiento. Esta API solo admite facturas de venta; las facturas de contrato se gestionan en el flujo de contratos de nuby.",
  "error_code": "FACTURA_CONTRATO_NO_SOPORTADA",
  "warnings": [
    { "code": "FACTURA_CONTRATO_NO_SOPORTADA", "factura_id": 990, "scope": "solo_facturas_de_venta" }
  ]
}
```

Y esta es la forma de la advertencia cuando el **payload** trae datos de contrato (sin `factura_id`, porque todavía no hay documento):

```
{
  "success": false,
  "status": 422,
  "message": "El payload intenta crear o modificar una factura con datos de contrato (movimiento_id). Esta API solo crea facturas de venta; las facturas de contrato se generan desde el flujo de contratos.",
  "error_code": "FACTURA_CONTRATO_NO_PERMITIDA",
  "errors": {
    "movimiento_id": [
```

```
    "El payload no puede incluir el campo movimiento_id en una factura de venta."
  ]
},
"warnings": [
  { "code": "FACTURA_CONTRATO_NO_PERMITIDA", "scope": "solo_facturas_de_venta" }
]
}
```

Un `factura_id` que no existe suele responder `403 RECURSO_AJENO`. La pertenencia se evalúa antes de la existencia: si nuby no tiene registrado que tu consumidor creó ese identificador, no puede distinguir "no es tuya" de "no existe" y responde `403`. El `404 FACTURA_NO_ENCONTRADA` aparece cuando la pertenencia sí está registrada pero la factura ya no existe.

6. Ejemplos de integración

Cada ejemplo envía el documento completo y usa una `referencia_externa` nueva por edición.

cURL

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

curl -X PUT "https://${INSTANCIA}/service/v2/public/invoices/${FACTURA_ID}" \
  -H "Authorization: Bearer ${TOKEN}" \
  -H "Content-Type: application/json" \
  -d '{
    "referencia_externa": "INV-2026-0001-EDIT-1",
    "tercero_id": 4821,
    "fecha": "2026-10-21",
    "dias_vencimiento": 15,
    "moneda": "COP",
    "observaciones": "Administración octubre 2026 (ajustada)",
```

```
"pago_total": false,
"items": [
  {
    "producto_id": 340,
    "descripcion": "Administración octubre 2026",
    "cantidad": 1,
    "valor_unitario": 1200000,
    "por_iva": 19,
    "iva": 228000
  }
]
}'
```

PHP

```
<?php
```

```
$instancia = 'mi-inmobiliaria.nuby.app'; // Reemplaza con tu dirección web completa
$token = 'TU_TOKEN_AQUI';                // Token obtenido del servicio Login
$facturaId = 10231;                       // factura_id devuelto al crear el borrador
```

```
$url = "https://{ $instancia }/service/v2/public/invoices/{ $facturaId }";
```

```
$payload = [
  'referencia_externa' => 'INV-2026-0001-EDIT-1',
  'tercero_id' => 4821,
  'fecha' => '2026-10-21',
  'dias_vencimiento' => 15,
  'moneda' => 'COP',
  'observaciones' => 'Administración octubre 2026 (ajustada)',
  'pago_total' => false,
  'items' => [
    [
      'producto_id' => 340,
      'descripcion' => 'Administración octubre 2026',
      'cantidad' => 1,
      'valor_unitario' => 1200000,
```

```

        'por_iva' => 19,
        'iva' => 228000,
    ],
],
];

$ch = curl_init($url);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_CUSTOMREQUEST, 'PUT');
curl_setopt($ch, CURLOPT_POSTFIELDS, json_encode($payload));
curl_setopt($ch, CURLOPT_HTTPHEADER, [
    'Content-Type: application/json',
    "Authorization: Bearer {$token}",
]);

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

if (curl_errno($ch)) {
    echo 'Error de conexión: ' . curl_error($ch); // reintenta con la MISMA referencia
} else {
    $info = json_decode($respuesta, true);

    if ($httpCode === 200) {
        echo "Borrador actualizado: factura_id={$info['body']['factura_id']}"
            . " total={$info['body']['valor_total']}"
            . " saldo={$info['body']['valor_saldo']}\n";

        if (!empty($info['idempotent_replay'])) {
            echo "Nota: fue un reintento idempotente, no se regrabó el documento.\n";
        }
    } else {
        echo "Rechazo HTTP {$httpCode}: {$info['message']}\n";
        echo "error_code: " . ($info['error_code'] ?? 'N/A') . "\n";

        if (!empty($info['warnings'])) {
            print_r($info['warnings']);
        }
    }
}

```

```
}  
  
}  
  
curl_close($ch);  
  
?>
```

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 # factura_id devuelto al crear el borrador  
  
# 2. Prepara la dirección y el cuerpo  
url = f"https://{instancia}/service/v2/public/invoices/{factura_id}"  
  
headers = {  
    "Content-Type": "application/json",  
    "Authorization": f"Bearer {token}"  
}  
  
payload = {  
    "referencia_externa": "INV-2026-0001-EDIT-1",  
    "tercero_id": 4821,  
    "fecha": "2026-10-21",  
    "dias_vencimiento": 15,  
    "moneda": "COP",  
    "observaciones": "Administración octubre 2026 (ajustada)",  
    "pago_total": False,  
    "items": [  
        {  
            "producto_id": 340,  
            "descripcion": "Administración octubre 2026",  
            "cantidad": 1,
```

```

        "valor_unitario": 1200000,
        "por_iva": 19,
        "iva": 228000
    }
]
}

# 3. Envía la petición y procesa la respuesta
try:
    response = requests.put(url, json=payload, headers=headers, timeout=30)

    print(f"Código de estado HTTP: {response.status_code}")

    if response.status_code == 200:
        info = response.json()
        body = info.get("body", {})

        print(f"Borrador actualizado: factura_id={body.get('factura_id')}")
        print(f"Total: {body.get('valor_total')} | Saldo: {body.get('valor_saldo')}")

        if info.get("idempotent_replay"):
            print("Nota: reintento idempotente, no se regrabó el documento.")
    else:
        info = response.json()
        print(f"Mensaje: {info.get('message')}")
        print(f"error_code: {info.get('error_code')}")

        for aviso in info.get("warnings") or []:
            print(f" advertencia: {aviso}")

except Exception as e:
    # Ante un timeout, reintenta con la MISMA referencia_externa.
    print(f"Ocurrió un error de conexión: {e}")

```

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
const facturaId = 10231; // factura_id devuelto al crear el borrador

const url = `https://${instancia}/service/v2/public/invoices/${facturaId}`;

const payload = {
  referencia_externa: 'INV-2026-0001-EDIT-1',
  tercero_id: 4821,
  fecha: '2026-10-21',
  dias_vencimiento: 15,
  moneda: 'COP',
  observaciones: 'Administración octubre 2026 (ajustada)',
  pago_total: false,
  items: [
    {
      producto_id: 340,
      descripcion: 'Administración octubre 2026',
      cantidad: 1,
      valor_unitario: 1200000,
      por_iva: 19,
      iva: 228000
    }
  ]
};

// 2. Función para editar el borrador
async function editarBorrador() {
  try {
    const response = await fetch(url, {
      method: 'PUT', // PATCH tiene la misma semántica
      headers: {
        'Content-Type': 'application/json',
        'Authorization': `Bearer ${token}`
      },
      body: JSON.stringify(payload)
    });
  }
};
```

```

console.log(`Código de estado HTTP: ${response.status}`);

const info = await response.json();

if (response.status === 200) {
  const body = info.body || {};

  console.log(`Borrador actualizado: factura_id=${body.factura_id}`);
  console.log(`Total: ${body.valor_total} | Saldo: ${body.valor_saldo}`);

  if (info.idempotent_replay) {
    console.log('Nota: reintento idempotente, no se regrabó el documento.');
  }
} else {
  console.log(`Mensaje: ${info.message}`);
  console.log(`error_code: ${info.error_code}`);

  (info.warnings || []).forEach((aviso) => console.log(' advertencia:',
aviso));
}
} catch (error) {
  // Ante un timeout, reintenta con la MISMA referencia_externa.
  console.error(`Ocurrió un error de conexión: ${error}`);
}
}

// 3. Ejecutamos la función
editarBorrador();

```

Power Query M (Excel / Power BI)

Power Query no puede elegir el verbo de la petición. `Web.Contents` con `Content` envía siempre `POST`, y este servicio solo acepta `PUT` y `PATCH`. Por eso la edición desde Excel o Power BI se hace a través de una función de tu backend (o de una función de Azure) que reciba el JSON y lo reenvíe a nuby

con `PUT`. El ejemplo muestra el cuerpo exacto que debes enviar a esa función y cómo leer el resultado.

```
let
// 1. Configura la dirección de TU función, que reenvía con PUT a nuby
urlFuncion = "https://tu-servicio.example.com/facturas/10231/editar",
token = "TU_TOKEN_AQUI", // Token obtenido del servicio Login

// 2. Construye el cuerpo JSON del contrato de edición
cuerpo = Json.FromValue([
    referencia_externa = "INV-2026-0001-EDIT-1",
    tercero_id = 4821,
    fecha = "2026-10-21",
    dias_vencimiento = 15,
    moneda = "COP",
    observaciones = "Administración octubre 2026 (ajustada)",
    pago_total = false,
    items = {
        [
            producto_id = 340,
            descripcion = "Administración octubre 2026",
            cantidad = 1,
            valor_unitario = 1200000,
            por_iva = 19,
            iva = 228000
        ]
    }
]),

// 3. Envía el cuerpo a tu función (ella añade el verbo PUT y el token)
response = Web.Contents(urlFuncion, [
    Headers = [
        #"Content-Type" = "application/json",
        #"Authorization" = "Bearer " & token
    ],
    Content = cuerpo
]),
```

```
// 4. Decodifica la respuesta JSON que devuelve nuby
jsonResponse = Json.Document(response),
body = jsonResponse[body],

// 5. Convierte el resultado en tabla
tabla = Table.FromRecords({body}),
expandido = Table.ExpandRecordColumn(tabla, "Column1",
    {"factura_id", "estado_id", "consecutivo", "prefijo", "valor_total",
"valor_saldo"},
    {"FacturaID", "EstadoID", "ConsecutivoSinNumerar", "ValorTotal", "ValorSaldo"})
in
expandido
```

Siguiente paso: cuando el borrador esté correcto, emítelo con **Emitir**
Factura a la DIAN. Recuerda que una vez emitida la factura ya no se puede editar.

Revisión #1

Creado el 28 octubre 2026 14:04:53 por Jorge Iván González

Actualizado el 28 octubre 2026 14:11:12 por Jorge Iván González