

Crear factura en borrador

[beta]

Este servicio crea una **factura de venta en borrador** en nuby. El documento nace en estado `5` (Borrador), **sin numerar**, con su contabilidad generada y dentro de una transacción con reversa: si algo falla, no queda nada a medias. nuby sigue siendo el dueño del documento; tu plataforma solo envía los datos de la operación.

Es el primer paso del ciclo de facturación. Después puedes **editar el borrador** mientras siga en ese estado y, cuando esté listo, **emitirlo a la DIAN**. Crear y emitir son dos operaciones distintas: no existe `emitir: true` en este servicio, y el borrador **no consume** un consecutivo real.

Para navegar esta página, haz clic sobre la sección que desees consultar. Cada sección se expande al dar clic.

1. ¿Para qué sirve este servicio?

Sirve para **registrar una factura de venta sin emitirla todavía**. Tu sistema envía la operación comercial —cliente, fecha, vencimiento, ítems, impuestos— y nuby crea el documento completo: cabecera, detalle, totales, saldo y su asiento en contabilidad, todo dentro de una transacción con reversa. Lo que **no** hace es numerarla ni enviarla a la DIAN: eso ocurre al emitirla.

? Casos de uso concretos

- **Facturación recurrente desde tu plataforma.** Si administras los cánones y los servicios de tus inmuebles en un sistema propio, generas el borrador de la factura del mes por API y nuby se encarga de la contabilidad, del consecutivo y del envío a la DIAN.
- **Pre-facturación con revisión humana.** Creas el borrador, alguien lo revisa en la interfaz de nuby, se corrige con el servicio de edición y solo entonces se emite. El borrador no gasta consecutivo, así que puedes crear y descartar sin ensuciar la

numeración.

- **Integración con tu CRM o con tu e-commerce.** Cada venta cerrada en tu plataforma se convierte en un borrador de factura en nuby, con la misma referencia externa que ya usas en tu sistema.
- **Cargas y migraciones.** Puedes registrar facturas pendientes de emisión que ya existen en otro sistema, una por petición, y emitirlas cuando corresponda.
- **Facturación de servicios periódicos.** Generas los borradores de un lote de clientes uno por uno (el servicio no acepta lotes) y controlas los duplicados con `referencia_externa`.

? Resumen del servicio

Aspecto	Valor
Método y ruta	<code>POST /service/v2/public/invoices</code>
Permiso (scope) requerido	<code>invoices:create</code>
Respuesta exitosa	<code>201</code> con <code>factura_id</code> , <code>estado_id</code> , <code>consecutivo</code> , <code>valor_total</code> , <code>valor_saldo</code> y <code>referencia_externa</code>
Efecto del documento	Estado <code>5</code> (Borrador), sin numerar, con contabilidad y saldo calculados
Protección contra duplicados	Idempotencia por <code>referencia_externa</code> + consumidor + operación
¿Envía a la DIAN?	No. El envío es la operación de emisión

? Lo que este servicio NO hace

- **No numera la factura.** El consecutivo real se asigna al emitir; en el borrador `consecutivo` es un marcador interno (cero o negativo).
- **No emite ni envía a la DIAN.** No existe `emitir: true`: la emisión es una acción explícita y separada.
- **No crea facturas de contrato.** Las facturas de arrendamiento nacen del flujo de contratos de nuby; esta API solo admite facturas de venta y rechaza los datos de contrato.
- **No completa los datos del tercero.** Si envías el objeto `tercero`, nuby no inventa direcciones, teléfonos ni correos faltantes.
- **No abre períodos contables.** Abrirlos es tarea del administrador de nuby.
- **No anula, no registra pagos y no acepta lotes.** Cada petición crea una sola factura.

? Dónde encaja en el ciclo

- | | | |
|--------------------|---|---------------|
| 1. Crear borrador | <code>POST /invoices</code> | (esta página) |
| 2. Editar borrador | <code>PUT PATCH /invoices/{factura_id}</code> | |

- | | |
|----------------------|-----------------------------------|
| 3. Emitir a la DIAN | POST /invoices/{factura_id}/issue |
| 4. Consultar detalle | GET /invoices/{factura_id} |
| 5. Obtener el PDF | GET /invoices/{factura_id}/pdf |

Si todavía no tienes el `tercero_id`, resuélvelo primero con la guía [Crear o Resolver Tercero](#), o envía el objeto `tercero` dentro de esta misma petición (ver sección 3).

2. El Endpoint (La dirección web)

Apunta tu sistema a la siguiente dirección de petición **POST**:

```
POST https://{instancia}/service/v2/public/invoices
```

¿Qué debes colocar en `{{instancia}}`? Es muy sencillo: corresponde a la **dirección web principal** que utilizas a diario para ingresar a tu plataforma, incluyendo la terminación `.nuby.app` o `.arrendasoft.co`. Por ejemplo, si para entrar escribes `inmobiliaria.nuby.app` en tu navegador, esa es exactamente tu instancia. Solo asegúrate de **no incluir** `https://` ni barras diagonales (`/`) al final.

? Encabezados requeridos

Encabezado	Valor	Obligatorio
Content-Type	application/json	Sí
Authorization	Bearer <token>, 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:create`. Hay dos rechazos distintos, y conviene no confundirlos:

- `403 CONSUMIDOR_NO_AUTENTICADO` → el token es válido pero no corresponde a un cliente OAuth habilitado para consumir la API. Mensaje: *El cliente autenticado no está autorizado para consumir este endpoint.*
- `403 RECURSO_NO_AUTORIZADO` → el cliente existe pero le falta el permiso `invoices:create`. Mensaje: *El consumidor no tiene habilitado el recurso solicitado.* Solicita el ajuste a tu

administrador en nuby.

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`.

? Precondiciones del sistema

nuby evalúa estas condiciones **antes de escribir nada**. Ninguna se resuelve reintentando: son configuración o datos de tu cuenta.

Precondición	Qué pasa si no se cumple
La facturación electrónica está habilitada en nuby (<code>ENABLE_FACT_ELECT = 1</code>).	<code>422 FACTURACION_ELECTRONICA_DESHABILITADA</code> . No se invoca el motor de facturación ni se escribe nada.
La <code>fecha</code> cae en un período contable abierto , cuando la contabilidad de la facturación está habilitada (<code>FACTURACION_CONTABILIDAD</code>).	<code>422 PERIODO_CONTABLE_NO_ABIERTO</code> o <code>422 PERIODO_CONTABLE_CERRADO</code> .
Existe una resolución de venta activa, o envías una válida en <code>resolucion_id</code> .	<code>422 RESOLUCION_NO_DISPONIBLE</code> o <code>422 RESOLUCION_CONTRATO_NO_PERMITIDA</code> .
El tercero existe en nuby, o envías el objeto <code>tercero</code> con sus datos fiscales.	<code>422 VALIDATION_ERROR</code> (falta el tercero) o <code>422 DATOS_FACTURA_INVALIDOS</code> (el tercero existe pero está incompleto).
El payload cumple el contrato de entrada y el motor de facturación acepta el documento.	<code>422 VALIDATION_ERROR</code> (error de validación del payload) o <code>422 DATOS_FACTURA_INVALIDOS</code> (el motor rechazó el documento, con la causa en <code>message</code>).

El período contable no lo abre tu integración. Abrirlo es una tarea del administrador de nuby: si recibes un código de período, avisa a tu contacto en nuby y usa una fecha con período abierto. El período **cerrado** pesa más que uno abierto: si la fecha cae en un período cerrado, el rechazo es `PERIODO_CONTABLE_CERRADO` aunque exista otro período abierto que también la contenga.

3. La Petición (¿Qué debes enviarnos?)

Envía un **objeto JSON** en el cuerpo. La raíz del JSON lleva la referencia de idempotencia y, al mismo nivel, la cabecera de la factura y sus ítems.

Elemento	Valor
Método	POST
Content-Type	application/json
Authorization	Bearer <token>
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 tu operación. Es la llave de idempotencia: un reintento con la misma referencia no crea una segunda factura. No puede venir vacía ni con solo espacios. Si falta: <code>422</code> con <i>El campo "referencia_externa" es obligatorio.</i>
<code>tercero_id</code>	entero	Sí, o el objeto <code>tercero</code>	—	Identificador interno del tercero en nuby. Debe ser mayor que 0. 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 (ver Datos fiscales del tercero). Sin ninguno de los dos: <i>Debe indicar <code>tercero_id</code> o el objeto <code>tercero</code>.</i> Si no es un arreglo: <i>El objeto <code>tercero</code> debe ser un arreglo con los datos del tercero.</i>

Parámetro	Tipo	Requerido	Por defecto	Descripción
fecha	texto AAAA-MM-DD	Sí	—	Fecha de la factura. Debe ser una fecha real y caer en un período contable utilizable. Si falta o está mal formada: <i>La fecha de la factura es obligatoria y debe tener el formato AAAA-MM-DD.</i>
dias_vencimiento	entero	Sí, salvo que uses fecha_vencimiento o pago_total: true	—	Días de plazo. Debe ser un entero mayor o igual a 1 . Si falta: <i>Los días de vencimiento son obligatorios y deben ser un entero mayor o igual a 1.</i>
fecha_vencimiento	texto AAAA-MM-DD	No	—	Alternativa a dias_vencimiento: nuby deriva los días desde fecha. Si envías ambos y no coinciden: <i>La fecha de vencimiento no coincide con los días de vencimiento enviados.</i> Formato inválido: <i>La fecha de vencimiento debe tener el formato AAAA-MM-DD.</i>
resolucion_id	entero	No	resolución de venta activa	Resolución de facturación. Debe ser un entero positivo, existir, estar activa y ser de facturación (no de contrato). Si se omite, nuby elige la primera resolución activa de venta.
moneda	texto	No	COP	Código de la moneda del documento. Una cadena vacía equivale a no enviarla.

Parámetro	Tipo	Requerido	Por defecto	Descripción
<code>tasa_cambio</code>	numérico	No	<code>null</code>	Tasa de cambio aplicable. Solo tiene sentido con moneda extranjera; se copia tal cual al documento.
<code>vendedor_id</code>	entero	No	<code>null</code>	Usuario de nuby que figura como vendedor del documento.
<code>centro_id</code>	entero	No	<code>null</code>	Centro de costo del documento.
<code>observaciones</code>	texto	No	<code>null</code>	Nota libre que queda guardada en la factura.
<code>pago_total</code>	booleano	No	<code>false</code>	Si es verdadero, la factura queda de contado : el vencimiento es igual a la fecha y <code>dias_vencimiento</code> se ignora .
<code>formas_de_pago</code>	arreglo de objetos	Sí, cuando <code>pago_total</code> es <code>true</code>	<code>[]</code>	Formas de pago del documento. Con <code>pago_total: true</code> debes enviar al menos una: <i>Debe agregar al menos una forma de pago cuando pago_total es verdadero</i> . Si no es un arreglo: <i>formas_de_pago debe ser un arreglo de formas de pago</i> . Con <code>pago_total: true</code> , nuby usa el <code>tipo_pago_id</code> del payload o, si no lo envías, el de la primera forma de pago.

Parámetro	Tipo	Requerido	Por defecto	Descripción
anticipos[]	arreglo	No	[]	Anticipos que se aplican a la factura. Acepta objetos ([{"documento_id": 5}]) o una lista plana de identificadores ([5]). Los identificadores no positivos se descartan. Se aplican después de guardar el borrador, porque necesitan el saldo ya persistido.
items	arreglo de objetos	Sí	—	Líneas del documento. Debe traer al menos una: <i>Debe enviar al menos un ítem en items.</i>

tipo_factura **no se acepta.** nuby lo fija en **1** (venta). Si lo envías —incluso con el valor correcto— la respuesta es **422 CAMPO_NO_PERMITIDO**.

El cuerpo debe ser un objeto JSON. Un arreglo, un texto o un cuerpo vacío devuelven **400** con *El cuerpo de la solicitud debe ser un objeto JSON*. Los campos que no están en la superficie aceptada se ignoran: no se copian al documento.

? Ítems (items[])

Cada elemento del arreglo es una línea de la factura. Los campos se validan **todos antes** de invocar el motor de facturación: si un ítem es inválido, la factura no se crea.

Campo	Tipo	Requerido	Descripción
producto_id	entero	Sí	Producto o servicio facturado. Mayor que 0. Error: <i>producto_id debe ser un entero mayor que 0.</i>

Campo	Tipo	Requerido	Descripción
<code>cantidad</code>	entero	Sí	Cantidad facturada. Entero mayor que 0 (el motor la guarda como entero). Error: <i>cantidad debe ser un entero mayor que 0.</i>
<code>valor_unitario</code>	numérico	Sí	Precio unitario. Numérico y distinto de cero . Error: <i>valor_unitario debe ser numérico y distinto de 0.</i>
<code>descripcion</code>	texto	No	Descripción de la línea. nuby la envía siempre al motor (vacía si no la mandas).
<code>por_descuento</code> / <code>descuento</code>	numérico	No	Porcentaje y valor del descuento de la línea.
<code>descuentos_id</code>	entero	No	Identificador de la regla de descuento aplicada.
<code>por_iva</code> / <code>iva</code>	numérico	No	Porcentaje y valor del IVA de la línea.
<code>por_retencion</code> / <code>retencion</code>	numérico	No	Porcentaje y valor de la retención en la fuente.
<code>por_reteiva</code> / <code>reteiva</code>	numérico	No	Porcentaje y valor de la retención de IVA.
<code>por_reteica</code> / <code>reteica</code>	numérico	No	Porcentaje y valor de la retención de ICA.
<code>otros_impuestos</code>	numérico	No	Otros impuestos de la línea.
<code>tercero_aplica_id</code>	entero	No	Tercero al que aplican las retenciones de la línea. nuby lo envía siempre al motor (<code>null</code> si no lo mandas).
<code>valor_unitario_ori</code>	numérico	No	Valor unitario en la moneda original. Si no lo envías, lo deriva el motor de facturación.

Por qué se rechazan los ítems inválidos: el motor de facturación descarta en silencio las líneas que no cumplen las reglas de `producto_id`, `cantidad` y `valor_unitario`. Por eso nuby las valida antes: así tu factura nunca queda con menos líneas que las que enviaste. Si un ítem no es un objeto, el

error por campo es *Cada ítem debe ser un arreglo con producto_id, cantidad y valor_unitario*. y la clave del error es `items.<indice>.<campo>`.

? Campos de contrato: rechazo con `warnings[]`

Esta API solo crea facturas de venta. Si el payload —en la cabecera **o dentro de un ítem**— trae datos de contrato o de movimiento, la respuesta es `422`

`FACTURA_CONTRATO_NO_PERMITIDA` con la advertencia estructurada:

- `movimiento_id`, `contrato_id`, `procesado`, `documento_respaldo_id`, `inmobiliaria`.
- Enviarlos en `null`, `""` o `0` **no** es un error: es la forma natural de decir "no aplica". Solo se rechaza el campo que trae un vínculo real.

```
{
  "success": false,
  "status": 422,
  "message": "El payload intenta crear o modificar una factura con datos de contrato (contrato_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": {
    "contrato_id": [
      "El payload no puede incluir el campo contrato_id en una factura de venta."
    ]
  },
  "warnings": [
    { "code": "FACTURA_CONTRATO_NO_PERMITIDA", "scope": "solo_facturas_de_venta" }
  ]
}
```

? Campos protegidos: `422 CAMPO_NO_PERMITIDO`

Solo los asigna nuby. Se evalúa la **presencia** de la clave: enviarla en `null` también se rechaza.

- `factura_id` (la creación siempre crea un documento nuevo; para editar un borrador existente se usa `PUT/PATCH /invoices/{factura_id}`), `estado_id`, `consecutivo`, `cufe`, `dian_id`, `estado_dian`, `creado_por`, `tipo_factura`.

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

? Vencimiento: las tres combinaciones válidas

Lo que envías	Resultado
<code>dias_vencimiento: 30</code>	Vence 30 días después de <code>fecha</code> .
<code>fecha_vencimiento: "2026-11-20"</code>	nuby calcula los días entre <code>fecha</code> y esa fecha. Si la fecha es anterior a la factura: <i>La fecha de vencimiento no puede ser anterior a la fecha de la factura.</i> Si da 0 días en una factura a crédito: <i>La fecha de vencimiento debe ser posterior a la fecha de la factura.</i>
<code>pago_total: true</code>	Factura de contado: los días se fuerzan a 0 y el vencimiento es igual a la fecha. <code>dias_vencimiento</code> se ignora. Si además envías una <code>fecha_vencimiento</code> distinta a <code>fecha</code> : <i>Con pago total, la fecha de vencimiento debe ser igual a la fecha de la factura.</i>

? Datos fiscales del tercero

Con la facturación electrónica habilitada, la validación legal del documento exige que el tercero tenga completos estos datos. Si usas `tercero_id` y al tercero le falta uno, la factura **no se crea** (422 DATOS_FACTURA_INVALIDOS con la lista de campos faltantes en el mensaje). Si usas el objeto `tercero`, envíalos desde el inicio:

- `direccion1` — dirección principal.
- `telefono1` — teléfono principal.
- `municipio_id` — municipio (mayor que 0).
- `email` — correo electrónico (obligatorio con la facturación electrónica activa).
- `documento` — número de documento del tercero.

El tercero y la factura no comparten conexión. Si nuby crea el tercero y la factura falla después, el tercero **queda creado**. No es un duplicado: el reintento lo resuelve por `documento`. Tenlo en cuenta al conciliar tus datos.

? Ejemplos de petición

? Ejemplo mínimo (tercero existente, factura a crédito)

```
{
  "referencia_externa": "INV-2026-0001",
  "tercero_id": 4821,
  "fecha": "2026-10-21",
  "dias_vencimiento": 30,
  "moneda": "COP",
  "observaciones": "Administración octubre 2026",
  "pago_total": false,
  "items": [
    {
      "producto_id": 340,
      "descripcion": "Administración octubre 2026",
      "cantidad": 1,
      "valor_unitario": 1000000,
      "por_iva": 19,
      "iva": 190000
    }
  ]
}
```

? Ejemplo con tercero incrustado y varios ítems

```
{
  "referencia_externa": "INV-2026-0002",
  "tercero": {
    "documento": "900123456",
    "tipo_doc_id": 6,
    "nombre1": "ACME SAS",
    "direccion1": "Calle 10 # 20-30",
    "telefono1": "6044447417",
    "municipio_id": 1,

```

```
    "email": "facturacion@acme.com"
  },
  "fecha": "2026-10-21",
  "fecha_vencimiento": "2026-11-20",
  "moneda": "COP",
  "resolucion_id": 12,
  "observaciones": "Administración y servicios",
  "items": [
    {
      "producto_id": 340,
      "descripcion": "Administración octubre 2026",
      "cantidad": 1,
      "valor_unitario": 1000000,
      "por_iva": 19,
      "iva": 190000
    },
    {
      "producto_id": 512,
      "descripcion": "Servicio de aseo octubre 2026",
      "cantidad": 2,
      "valor_unitario": 150000,
      "por_iva": 19,
      "iva": 57000
    }
  ]
}
```

? Ejemplo de contado, con formas de pago y anticipo

```
{
  "referencia_externa": "INV-2026-0003",
  "tercero_id": 4821,
  "fecha": "2026-10-21",
  "pago_total": true,
  "formas_de_pago": [
    { "tipo_pago_id": 1, "valor": 1190000 }
  ],
  "anticipos": [ { "documento_id": 5 } ],
  "items": [
```

```
{
  "producto_id": 340,
  "cantidad": 1,
  "valor_unitario": 1000000,
  "por_iva": 19,
  "iva": 190000
}
```

4. La Respuesta (¿Qué te entregaremos?)

Si todo es válido, nuby crea la factura en estado **borrador** con su contabilidad y responde `201`:

```
{
  "success": true,
  "status": 201,
  "message": "Factura creada en borrador.",
  "body": {
    "factura_id": 10231,
    "estado_id": 5,
    "consecutivo": -1,
    "valor_total": 1190000,
    "valor_saldo": 1190000,
    "referencia_externa": "INV-2026-0001"
  },
  "idempotent_replay": false
}
```

? Estructura principal de la respuesta

Campo	Tipo	Descripción
<code>success</code>	booleano	<code>true</code> cuando la factura se creó. Los rechazos traen <code>false</code> y un <code>error_code</code> .
<code>status</code>	entero	Código HTTP replicado en el cuerpo. Aquí siempre <code>201</code> .

Campo	Tipo	Descripción
<code>message</code>	texto	Mensaje legible para la persona que opera la integración: <i>Factura creada en borrador</i> .
<code>body</code>	objeto	Resultado de la creación (tabla siguiente).
<code>idempotent_replay</code>	booleano	<code>false</code> en una creación nueva; <code>true</code> cuando la respuesta se recuperó de una solicitud anterior y no se ejecutó nada.

? Campos de `body`

Campo	Tipo	Descripción
<code>factura_id</code>	entero	Identificador interno del borrador. Guárdalo: es lo que necesitas para editarlo, emitirlo, consultar el detalle y pedir el PDF. No es el número de la factura.
<code>estado_id</code>	entero	Estado del documento. Aquí siempre <code>5</code> (Borrador).
<code>consecutivo</code>	entero	Marcador de "sin numerar": cero o negativo . El consecutivo real se asigna al emitir.
<code>valor_total</code>	numérico	Total del documento, calculado por nubly con los impuestos y descuentos del detalle.
<code>valor_saldo</code>	numérico	Saldo pendiente del documento. En un borrador recién creado coincide con el total; los anticipos aplicados lo reducen.
<code>referencia_externa</code>	texto	La misma referencia que enviaste, ya normalizada (sin espacios en los extremos). Queda asociada a la factura para el control de duplicados.

? El borrador NO consume un consecutivo real

En un borrador, `consecutivo` vale **cero o un número negativo** (por ejemplo `-1`). Ese valor es un marcador interno de "sin numerar": nubly lo asigna para que dos borradores nunca colisionen entre sí. **No lo interpretes como número de factura**, no lo muestres al cliente final y no lo uses como identificador de negocio. La garantía del contrato es que el borrador no gasta numeración; el valor exacto

del marcador lo decide nuby, así que no lo compares con un número fijo.

? Reintento de una solicitud ya completada (replay)

Si repites la petición con la **misma** `referencia_externa` y el mismo contenido, nuby devuelve la respuesta original —con el mismo código HTTP— y le agrega la marca `idempotent_replay: true`. No se crea una segunda factura ni se consume contabilidad nueva:

```
{
  "success": true,
  "status": 201,
  "message": "Factura creada en borrador.",
  "body": {
    "factura_id": 10231,
    "estado_id": 5,
    "consecutivo": -1,
    "valor_total": 1190000,
    "valor_saldo": 1190000,
    "referencia_externa": "INV-2026-0001"
  },
  "idempotent_replay": true
}
```

La marca viaja en la **raíz** de la respuesta. La misma referencia con un contenido distinto responde `409 IDEMPOTENCY_PAYLOAD_MISMATCH`: *La referencia externa ya fue usada con un payload distinto. Usa una referencia nueva para una operación nueva.*

? Cómo leer los valores numéricos

`valor_total` y `valor_saldo` llegan como números JSON; no los compares como texto. Para mostrarlos al usuario final, formatea con la moneda del documento (`moneda`, `COP` por defecto).

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 la operación con la **misma** `referencia_externa`: la solicitud que no llegó a procesarse no consumió la clave.

Así se ve el error que devuelve el sistema cuando el token venció o el pase es inválido:

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

Ese **401** no viene del servicio de facturas sino del middleware de autenticación de la API, así que su forma es distinta a la de los demás errores: no trae `success` ni `error_code`. En cambio los rechazos de la operación —validación, permisos, idempotencia y reglas de negocio— sí usan el sobre uniforme que se describe más abajo.

? Sobre uniforme de error

Campo	Tipo	Descripción
<code>success</code>	booleano	Siempre <code>false</code> .
<code>status</code>	entero	El código HTTP, replicado en el cuerpo.
<code>message</code>	texto	Explicación legible del rechazo.
<code>error_code</code>	texto	Código estable de negocio. Programa contra este campo, nunca contra el mensaje.
<code>errors</code>	objeto	Solo en rechazos de validación: causas por campo. En los ítems, la clave es <code>items.<indice>.<campo></code> .

Campo	Tipo	Descripción
warnings	arreglo	Solo cuando la operación se rechaza o se condiciona por alcance. Cada elemento trae <code>code</code> , <code>scope</code> y, según el caso, <code>factura_id</code> .

? Tabla de códigos HTTP, causas y mensajes

HTTP	error_code	Causa y mensaje exacto
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	—	No se envió el encabezado <code>Authorization</code> . <i>JWT Token required.</i>
401	—	El token no existe en el sistema. <i>JWT Token not found.</i>
401	—	El token expiró (dura 1 hora). <i>JWT Token expired.</i> Vuelve a consumir Login .
403	CONSUMIDOR_NO_AUTENTICADO	El token es válido pero 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 OAuth le falta el permiso <code>invoices:create</code> . <i>El consumidor no tiene habilitado el recurso solicitado.</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. <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.), el payload no cumple el formato (El payload no es válido.) o un ítem es inválido. Trae <code>errors</code> por campo.

HTTP	error_code	Causa y mensaje exacto
422	FACTURA_CONTRATO_NO_PERMITIDA	El payload trae <code>movimiento_id</code> , <code>contrato_id</code> , <code>procesado</code> , <code>documento_respaldo_id</code> o <code>inmobiliaria</code> 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	FACTURACION_ELECTRONICA_DESHABILITADA	<code>ENABLE_FACT_ELECT</code> no vale <code>1</code> . No se escribe nada. <i>La facturación electrónica no está habilitada en este sistema, así que no se pueden crear borradores de facturas de venta por API.</i>
422	RESOLUCION_CONTRATO_NO_PERMITIDA	El <code>resolucion_id</code> es una resolución 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	No hay resolución de venta activa, o la 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>
422	PERIODO_CONTABLE_NO_ABIERTO	La fecha no cae en ningún período contable abierto. <i>No existe un período contable abierto para la fecha {fecha}. Debe abrirse el período antes de facturar.</i>
422	PERIODO_CONTABLE_CERRADO	La fecha cae en un período contable cerrado. <i>El período contable de la fecha {fecha} está cerrado. No se pueden registrar movimientos.</i>
422	DATOS_FACTURA_INVALIDOS	El motor de facturación rechazó el documento (por ejemplo, un tercero con datos fiscales incompletos). El <code>message</code> trae la causa concreta.

HTTP	error_code	Causa y mensaje exacto
500	INTERNAL_ERROR	Fallo técnico. El mensaje es genérico a propósito: <i>No fue posible crear la factura de venta. Intente nuevamente o contacte al administrador del sistema.</i>
500	IDEMPOTENCY_STORAGE_ERROR	No se pudo registrar la solicitud para controlar duplicados. <i>No se pudo registrar la solicitud de idempotencia.</i>

Este servicio no devuelve 409 de estado ni 502 de la DIAN. Aquí no hay factura previa que pueda estar fuera de borrador ni se envía nada a la DIAN: esos códigos aparecen al editar el borrador y al emitir la factura.

? Ejemplo de rechazo de validación por campo

```
{
  "success": false,
  "status": 422,
  "message": "El payload no es válido.",
  "error_code": "VALIDATION_ERROR",
  "errors": {
    "dias_vencimiento": [
      "Los días de vencimiento son obligatorios y deben ser un entero mayor o igual a 1."
    ],
    "items.0.valor_unitario": [
      "valor_unitario debe ser numérico y distinto de 0."
    ]
  }
}
```

? Reglas de reintento

Situación	Qué hace nuby con la referencia	Qué debes hacer
Rechazo 422 de negocio	Libera la referencia externa	Corrige el payload y reintenta con la misma referencia.
Fallo 500 o 502	Deja la solicitud fallida	Reintenta con la misma referencia.

Situación	Qué hace nuby con la referencia	Qué debes hacer
Éxito <code>201</code>	Deja la solicitud completada	No la reutilices. Una operación nueva necesita una referencia nueva.
<code>403</code> , <code>401</code> , <code>400</code>	No se reserva ni se consume la referencia	Corrige el acceso, el token o el cuerpo y reintentá con la misma referencia.

6. Ejemplos de integración

Aquí tienes ejemplos listos para que tus desarrolladores los adapten a tu plataforma. Todos guardan la relación `referencia_externa → factura_id`, que es lo que permite reintentar sin duplicar.

? cURL

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

curl -X POST "https://${INSTANCIA}/service/v2/public/invoices" \
  -H "Authorization: Bearer ${TOKEN}" \
  -H "Content-Type: application/json" \
  -d '{
    "referencia_externa": "INV-2026-0001",
    "tercero_id": 4821,
    "fecha": "2026-10-21",
    "dias_vencimiento": 30,
    "moneda": "COP",
    "observaciones": "Administración octubre 2026",
    "pago_total": false,
    "items": [
      {
        "producto_id": 340,
        "descripcion": "Administración octubre 2026",
        "cantidad": 1,
        "valor_unitario": 1000000,
        "por_iva": 19,
        "iva": 190000
      }
    ]
  }
```

```
}  
]  
'
```

? PHP

```
<?php
```

```
$instancia = 'mi-inmobiliaria.nuby.app'; // Reemplaza con tu dirección web completa  
$token = 'TU_TOKEN_AQUI';                // Token obtenido del servicio Login
```

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

```
$payload = [  
    'referencia_externa' => 'INV-2026-0001',  
    'tercero_id' => 4821,  
    'fecha' => '2026-10-21',  
    'dias_vencimiento' => 30,  
    'moneda' => 'COP',  
    'observaciones' => 'Administración octubre 2026',  
    'pago_total' => false,  
    'items' => [  
        [  
            'producto_id' => 340,  
            'descripcion' => 'Administración octubre 2026',  
            'cantidad' => 1,  
            'valor_unitario' => 1000000,  
            'por_iva' => 19,  
            'iva' => 190000,  
        ],  
    ],  
];
```

```
$ch = curl_init($url);  
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);  
curl_setopt($ch, CURLOPT_POST, true);  
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($ch)) {
    // Timeout o caída de red: reintenta con la MISMA referencia externa.
    echo 'Error de conexión: ' . curl_error($ch);
} else {
    $info = json_decode($respuesta, true);

    if ($httpCode === 201) {
        $facturaId = $info['body']['factura_id'];

        echo "Borrador creado: factura_id={$facturaId}"
            . " consecutivo={$info['body']['consecutivo']}"
            . " total={$info['body']['valor_total']}\n";

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

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

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

# 2. Prepara la dirección y el cuerpo
url = f"https://{instancia}/service/v2/public/invoices"

headers = {
    "Content-Type": "application/json",
    "Authorization": f"Bearer {token}"
}

payload = {
    "referencia_externa": "INV-2026-0001",
    "tercero_id": 4821,
    "fecha": "2026-10-21",
    "dias_vencimiento": 30,
    "moneda": "COP",
    "observaciones": "Administración octubre 2026",
    "pago_total": False,
    "items": [
        {
            "producto_id": 340,
            "descripcion": "Administración octubre 2026",
            "cantidad": 1,
            "valor_unitario": 1000000,
            "por_iva": 19,
            "iva": 190000
        }
    ]
}

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

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

```

if response.status_code == 201:
    info = response.json()
    body = info.get("body", {})

    print(f"Borrador creado: factura_id={body.get('factura_id')}")
    print(f"Estado: {body.get('estado_id')} | Consecutivo (sin numerar):
{body.get('consecutivo')}")
    print(f"Total: {body.get('valor_total')} | Saldo: {body.get('valor_saldo')}")

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

    for campo, errores in (info.get("errors") or {}).items():
        print(f" {campo}: {errores}")

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

```

? JavaScript (Node.js / navegador)

```

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

const payload = {
    referencia_externa: 'INV-2026-0001',
    tercero_id: 4821,
    fecha: '2026-10-21',
    dias_vencimiento: 30,
    moneda: 'COP',
    observaciones: 'Administración octubre 2026',

```

```
    pago_total: false,
    items: [
      {
        producto_id: 340,
        descripcion: 'Administración octubre 2026',
        cantidad: 1,
        valor_unitario: 1000000,
        por_iva: 19,
        iva: 190000
      }
    ]
  };
```

// 2. Función para crear el borrador

```
async function crearBorrador() {
  try {
    const response = await fetch(url, {
      method: 'POST',
      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 === 201) {
      const body = info.body || {};

      console.log(`Borrador creado: factura_id=${body.factura_id}`);
      console.log(`Estado: ${body.estado_id} | Consecutivo (sin numerar): ${body.consecutivo}`);
      console.log(`Total: ${body.valor_total} | Saldo: ${body.valor_saldo}`);

      if (info.idempotent_replay) {
        console.log('Nota: reintento idempotente, no se creó otra factura.');
```

```

    }
  } else {
    console.log(`Mensaje: ${info.message}`);
    console.log(`error_code: ${info.error_code}`);

    Object.entries(info.errors || {}).forEach(([campo, errores]) => {
      console.log(`  ${campo}: ${errores}`);
    });
  }
} catch (error) {
  // Ante un timeout, reintentar con la MISMA referencia_externa.
  console.error(`Ocurrió un error de conexión: ${error}`);
}

// 3. Ejecutamos la función
crearBorrador();

```

? Power Query M (Excel / Power BI)

Power Query convierte la respuesta en una tabla de una fila con los datos del borrador. Si la petición devuelve un error HTTP, `Web.Contents` genera una excepción: captúrala con `try ... otherwise` para poder registrar el `error_code`.

```

let
  // 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

  // 2. Prepara la dirección de la petición
  url = "https://" & instancia & "/service/v2/public/invoices",

  // 3. Construye el cuerpo JSON
  cuerpo = Json.FromValue([
    referencia_externa = "INV-2026-0001",
    tercero_id = 4821,
    fecha = "2026-10-21",
    dias_vencimiento = 30,
    moneda = "COP",

```

```
observaciones = "Administración octubre 2026",
pago_total = false,
items = {
    [
        producto_id = 340,
        descripcion = "Administración octubre 2026",
        cantidad = 1,
        valor_unitario = 1000000,
        por_iva = 19,
        iva = 190000
    ]
}
]),
```

```
// 4. Envía la petición POST
response = Web.Contents(url, [
    Headers = [
        #"Content-Type" = "application/json",
        #"Authorization" = "Bearer " & token
    ],
    Content = cuerpo
]),
```

```
// 5. Decodifica la respuesta JSON
jsonResponse = Json.Document(response),
body = jsonResponse[body],
```

```
// 6. Convierte el resultado en tabla
tabla = Table.FromRecords({body}),
expandido = Table.ExpandRecordColumn(tabla, "Column1",
    {"factura_id", "estado_id", "consecutivo", "valor_total", "valor_saldo",
"referencia_externa"},
    {"FacturaID", "EstadoID", "ConsecutivoSinNumerar", "ValorTotal", "ValorSaldo",
"ReferenciaExterna"})
in
expandido
```

Siguiente paso: con el `factura_id` en mano, revisa [Editar Factura en Borrador](#) para ajustar el documento y [Emitir Factura a la DIAN](#) para convertirlo en documento legal. Para consultarlo, revisa [Ver Detalle de una Factura](#).

Revisión #2

Creado el 21 octubre 2026 15:08:27 por Jorge Iván González

Actualizado el 21 octubre 2026 15:34:45 por Jorge Iván González