

Outgoing invoices

Create tax invoices and credit notes; validate, submit, deliver, and record payments.

Source: <https://alyabooks.ai/docs/guides/outgoing-invoices>

The invoice object

POST /invoices takes { invoices: [...] } with up to 100 inputs. Each input describes one document:

Field	Meaning
invoiceTypeCode	UNCL 1001. Valid on UAE e-invoices: 380 tax invoice (default), 381 credit note, 480 out-of-scope invoice, 81 credit note for out-of-scope goods or services; with self-billing switched on (Settings → Codes & regions), 389 self-billed invoice and 261 self-billed credit note. Other codes such as 383 are only accepted when the organization switched them on, and are rejected at submission. documentKind: "credit_note" is accepted as an alias.
creditNoteReasonCode	BTAE-03, required on credit notes: DL8.61.1.A to DL8.61.1.E or VD (volume discount, which needs no preceding invoice).
transactionTypes	BTAE-02, any combination: free_trade_zone, deemed_supply, profit_margin, summary_invoice, continuous_supply, disclosed_agent, e_commerce, exports. Empty for a standard supply. transactionType (one value, or standard) is still accepted. Some combinations with the document kind are not allowed and some types change the document; see Document kind and transaction type rules.
currency, exchangeRate	ISO 4217, default AED. For any other currency the rate is AED per 1 unit, at most 6 decimals (for USD, 3.6725); leave exchangeRate out to use the UAE Central Bank rate. The XML then also states VAT and the total in AED; see Other currencies.
invoiceNumber	Leave empty to allocate from the series. A supplied number must be unused. Accepted even when the organization option allowManualNumbers is off (that option only locks the number field in the app editor).
issueDate, dueDate, deliveryDate, taxPointDate	YYYY-MM-DD. payment.termsDays computes dueDate when it is missing. A due date is required whenever an amount is due (IBR-127-AE); on a deemed supply it is dropped.
buyer	{ id } of an existing customer, or an inline { name, trn, email, peppolId, address, reference }. Inline buyers are matched by TRN, then by name, otherwise created.

Field	Meaning
items[]	description, quantity, unitPrice (net), unitCode (UN/ECE Rec 20, default EA), taxCategory (standard, zero_rated, exempt, reverse_charge, out_of_scope; margin only with profit_margin), taxRate (defaults from the category), itemId, hsnCode, metadata. On 480 and 81 only zero_rated, exempt and out_of_scope pass validation.
allowances[]	Document-level discounts and surcharges: indicator (discount / surcharge), amount (net), reason, taxCategory, taxRate.
paymentMeans[]	Up to 10 payment instructions (BG-16): code (UNCL 4461; valid in the UAE: 1, 10, 20, 21, 30, 49, 54, 55, 68), reference, information, accountId (IBAN or account number), accountName, bic, bankName, cardLastDigits, cardNetwork, cardHolder, mandateReference, debitedAccount, creditorId, instructionId. A credit transfer (30) without accountId uses the organization's bank account from Settings. At most one card and one direct debit.
payment	termsDays, termsText; and, for a single payment method, meansCode, accountId, accountName, bic, reference (ignored when paymentMeans is given).
precedingInvoice	For credit and debit notes: { id } of the Alya Books invoice being corrected, or { number, issueDate }.
billingPeriod, delivery, payee, taxRepresentative	Optional groups mapping to BG-14, BG-13, BG-10 and BG-11.
notes, subject, introduction, closing, language	Texts; language (en, ar, bilingual) controls the PDF. The option showTextBlocks only decides whether the app editor shows the salutation, introduction and closing blocks on new invoices; texts you send through the API are stored as sent.
metadata	Custom field values of the active template; see Custom fields .
externalId	Your own reference, searchable with ?externalId=.
projectId	Link to a project ; projectReference (BT-11) defaults to its code.
supplyEmirate	Emirate reported in box 1 of the VAT 201 (defaults to the organization's emirate); see VAT return .
Options	draftOnly, reviewMode, validate, submit, send, isTest.

Totals are computed server-side with the same rounding as the editor: line net = quantity × unit price rounded to 2 decimals; VAT per category on (lines – allowances + charges); document total = taxable + VAT.

Validation

Every invoice is validated against PINT AE before it is stored. Blocking errors (severity `error`) reject the element with `E_VALIDATION`, unless `draftOnly` is set; warnings are stored with the invoice.

```
{
  "code": "E_VALIDATION", "statusCode": 400,
```

```

    "message": "The invoice does not pass PINT AE validation",
    "extra": {
      "errors": [{ "field": "seller_trn", "rule": "BT-31", "message": "Seller TRN is required (set in Organization Settings)", "severity": "error" }],
      "warnings": [{ "field": "due_date", "rule": "BT-9", "message": "Due date is recommended", "severity": "warning" }]
    }
  }
}

```

Seller data (BT-27 to BT-43, the trade licence and its licensing authority) comes from the organization profile, not from the payload. IBANs are checked for length, structure and check digits (rule `IBAN`); a plain account number that is not an IBAN is accepted. The [PINT AE field reference](#) lists every business term, its UBL path and the API field that carries it.

`GET /invoices/{id}/validation` re-runs the checks; `POST /invoices/{id}/validate` also moves a draft to `validated`.

Document kind and transaction type rules

The web editor greys these out before you can pick them; the API accepts the input, runs the same validation and stores the invoice as a draft with the finding, so a client sees the rule number in `validation.errors` rather than a 4xx.

Situation	What happens
<code>480</code> or <code>81</code> with <code>summary_invoice</code> , <code>deemed_supply</code> or <code>profit_margin</code>	Validation error <code>IBR-157-AE</code> ; the invoice stays a draft.
<code>480</code> or <code>81</code> with a <code>standard</code> , <code>reverse_charge</code> or <code>margin</code> line	Validation error <code>IBR-122-AE</code> : out-of-scope documents carry only <code>zero_rated</code> , <code>exempt</code> and <code>out_of_scope</code> lines.
<code>deemed_supply</code>	A supply without a payment. <code>dueDate</code> , <code>payment.termsDays</code> and <code>paymentMeans</code> are removed on save whatever was sent (<code>IBR-127-AE</code> , <code>IBR-191-AE</code>); the response has <code>dueDate: null</code> , <code>paymentMeans: []</code> , <code>paymentStatus: "nothing_due"</code> and <code>outstanding: 0</code> . It is never overdue, <code>invoice.paid</code> is never emitted for it and <code>POST /invoices/{id}/payment-link</code> is refused. The buyer endpoint becomes <code>0235:9900000097</code> .
<code>profit_margin</code>	Every line must use <code>taxCategory: "margin"</code> (<code>IBR-116-AE</code>), which states 5 % and no VAT amount; document-level <code>allowances</code> are refused (<code>IBR-114-AE</code>). Send <code>marginVatAmount</code> per line for the VAT return.
Organization not registered for VAT	Only <code>480</code> and <code>81</code> validate (<code>IBR-134-AE</code>).

Other currencies

A document in another currency than AED needs an exchange rate (validation rule IBR-159-AE; at most 6 decimals, IBR-002-AE; a USD rate more than 1 % away from the 3.6725 peg gives a warning). Send `exchangeRate` to use your own, or leave it out and Alya Books fills in the UAE Central Bank rate "for VAT related obligations" for the date of supply: `taxPointDate`, else `deliveryDate`, else `issueDate`. The Central Bank publishes on working days around 18:00 UAE time; on weekends, holidays and before publication the last earlier rate is used. The response says which: `exchangeRateSource` is `cbuae` (with `exchangeRateDate`, the publication date) or `manual`. When the Central Bank has no rate for the currency the invoice fails IBR-159-AE, or stays a draft with `draftOnly`. All amounts in the payload and the response stay in the document currency. Alya Books adds the AED figures the UAE specification asks for:

XML	Value
<code>cbc:TaxCurrencyCode</code>	AED (set automatically; <code>vatAccountingCurrency</code> other than AED is rejected)
<code>cac:TaxExchangeRate</code>	source currency, AED, and the rate (BTAE-04)
second <code>cac:TaxTotal</code>	total VAT in AED (IBT-111)
<code>cac:AdditionalDocumentReference</code> type <code>aedtotal-incl-vat</code>	total with VAT in AED (BTAE-20)
<code>cac:ItemPriceExtension</code> on each line	line total with VAT and line VAT in AED (BTAE-10, BTAE-08); present on AED documents too

AED amounts are the document amounts $\times$ rate, rounded to 2 decimals. `GET /invoices/{id}` returns them as `aedTotals: { currency, rate, taxTotal, total }` (null for AED documents). The rate is stored on the invoice, so issued invoices never change when later rates are published.

Batches and partial failure

Elements succeed or fail independently. When all succeed the response is `202` with an array. When any element fails, the status mirrors the first failure (usually `400`) and `extra.results` holds one entry per element:

```
{
  "code": "E_VALIDATION", "statusCode": 400, "message": "...",
  "extra": { "results": [
    { "index": 0, "success": true, "invoiceId": "...", "invoiceNumber": "INV-0042" },
    { "index": 1, "success": false, "code": "E_VALIDATION", "statusCode": 400, "message": "...", "field": "invoiceNumber" }
  ] }
}
```

Warning: Elements with `success: true` **have been created**. Reconcile with `extra.results` and retry only the failed elements; a blind retry of the whole batch creates duplicate invoices when numbers are auto-allocated.

Submission and delivery

- `submit: true` on creation, or `POST /invoices/{id}/submit`, hands a validated invoice to the ASP. The response carries the ASP request id and the first status; the invoice becomes `submitted` and is locked.
- `send: { to, cc, bcc, subject, message }` on creation, or `POST /invoices/{id}/send`, emails the PDF and (for tax documents) the XML. `state` is `sent`, `skipped` (no email provider configured on the environment) or `failed`.
- `reviewMode: true` creates the invoice with `reviewStatus: pending`; an approver must approve it in the app before it can be submitted.

The [ASP and FTA lifecycle](#) page explains the stages and the `authority[]` entries in `GET /invoices/{id}/status`.

After issue

Issued invoices (`submitted` and later) are immutable. What you can still do:

Action	Route
Record a payment	<code>POST /invoices/{id}/payments { amount, date, method, reference };</code> negative amounts are refunds
Category, cost centre, archive, custom fields	<code>PATCH /invoices/{id}</code>
Correct the amounts	Create a credit note (381) with <code>precedingInvoice: { id }</code> and <code>creditNoteReasonCode</code> ; for an additional charge issue a new tax invoice (380)
Download	<code>GET /invoices/{id}?format=xml</code>
PDF design	<code>designTemplateId</code> on create or <code>PATCH /invoices/{id}</code> (null = the organisation's default); <code>GET /design-templates</code> lists the designs and their layout

Drafts and validated invoices can be replaced entirely with `PUT /invoices/{id}` or deleted with `DELETE /invoices/{id}`.

Listing and search

`GET /invoices` supports `keyword` (number, buyer, TRN, order reference), `invoiceNumber`, `status`, `paymentStatus` (comma-separated: `unpaid`, `partially_paid`, `paid`, `overpaid`, `nothing_due`; the last is a deemed supply, which has no payment), `documentKind`, `buyerId`, `startIssueDate` / `endIssueDate`, `startDate` / `endDate` (creation), `includeArchived`, `isTest`, `externalId`, `sort` (`issueDate`, `createdAt`, `invoiceNumber`, `dueDate`, `total`) and `order`. Test keys only see test documents unless `isTest=false` is passed explicitly.