Athena Accounting

MyInvois e-Invoice Requirements

Version 1.0.0-rc.1 · AthenaLab

Document date: 2026-08-07 · v1.0.0 (General Availability)

AI-first Malaysian double-entry accounting & e-Invoice

Malaysian e-Invoice (MyInvois) — Requirements Research Summary & Plan (Phase 0)

Purpose: capture the current official Malaysian e-Invoice requirements from authoritative sources, mark what still needs legal/tax-professional confirmation, and define the research plan Phase 5 executes. This is a **research summary, not a compliance certification.**

> ■■ **Not tax or legal advice.** Every requirement below must be confirmed against the *latest* official LHDN documents and reviewed with a qualified Malaysian tax adviser before production use. Versions and dates move; treat this as a dated snapshot. See `COMPLIANCE_CHECKLIST.md`.

Snapshot date: 2026-08-03.

1. Authoritative sources (source of truth)

Only official sources are used. Blogs/vendors are for orientation only and are **not** the source of truth.

Source	URL	Use
LHDN e-Invoice hub	https://www.hasil.gov.my/en/e-invoice/	Landing / official index
Implementation in Malaysia	https://www.hasil.gov.my/en/e-invoice/implementation-of-e-invoicing-in-malaysia/	Overview & model
Implementation timeline	https://www.hasil.gov.my/e-invois/pelaksanaan-e-invois-di-malaysia/garis-masa-pelaksanaan-e-invois/	Mandatory dates by turnover
e-Invoice Guideline (PDF, latest version)	https://www.hasil.gov.my/wp-content/uploads/IRBM-e-Invoice-Guideline.pdf	Core rules, model, workflow
e-Invoice Specific Guideline (PDF)	https://www.hasil.gov.my/media/uwwehxwq/irbm-e-invoice-specific-guideline.pdf	Field-level & scenario rules
General FAQs (PDF)	https://www.hasil.gov.my/wp-content/uploads/lhdnm-e-invoice-general-faqs.pdf	Clarifications
MyInvois SDK (developer portal)	https://sdk.myinvois.hasil.gov.my/	API + data catalogue
SDK — e-Invoice APIs	https://sdk.myinvois.hasil.gov.my/einvoicingapi/	Endpoint contracts
SDK — Getting started	https://sdk.myinvois.hasil.gov.my/start/	Onboarding, auth, signing

Action for Phase 5: re-fetch each of these, record the exact **version number and publication date** seen, and diff against this snapshot before writing any mapping code. The Guideline PDFs are versioned (e.g. v4.x) — pin the version.

---

2. Implementation model (as understood, to confirm)

1. **MyInvois** is LHDN's central platform. Two integration modes:
2. **MyInvois Portal** — manual web entry (fallback / low volume).
3. **API integration** — direct system-to-system submission (what Athena

Accounting targets).

- **Continuous Transaction Control (near-real-time validation):** the supplier

submits the e-Invoice to MyInvois, which **validates** it and returns a validated document with a **unique identifier**, a **validation link/QR code**, and a timestamp. The validated e-Invoice is then shared with the buyer.

- **Format:** documents follow **UBL 2.1**, submitted as **XML or JSON**.
- **Digital signature:** each submitted document must carry a **digital

signature** using a valid digital certificate (authentication, non-repudiation, integrity). *Certificate source/type to confirm.*

- **Async submission:** submission is asynchronous — you submit a (signed) batch,

receive a submission id, then **poll** for per-document validation results.

---

3. Implementation timeline (to confirm against the official timeline page)

Phased by annual turnover/revenue (based on FY2022 audited accounts). Key publicly-referenced markers as of this snapshot:

- Phased mandatory adoption ramps from the largest taxpayers down to smaller ones.
- **1 July 2026** referenced as a concessionary e-Invoice implementation date in

the current rollout.

- An exemption threshold around **RM150,000** annual turnover for the smallest

businesses has been discussed, with dates that have shifted more than once.

> ■ **Confirm exact current dates and thresholds directly from the LHDN timeline page** — these have been revised repeatedly. Determine **AthenaLab's own mandatory date** based on AthenaLab's turnover band. This drives urgency but not architecture.

---

4. Document types (to confirm exact codes)

MyInvois defines a small set of document types; the mapping Athena Accounting needs:

Athena source document	MyInvois type (confirm code)
Sales invoice	Invoice (`01`)
Credit note	Credit Note (`02`)
Debit note	Debit Note (`03`)
Refund note	Refund Note (`04`)
Purchases from non-e-Invoice supplier	**Self-billed** variants (self-billed invoice/CN/DN/refund)

Self-billed scenarios (e.g. foreign suppliers, certain payments) are a distinct workflow — flagged on `suppliers.is_self_billed_applicable`.

---

5. Mandatory data — the fields that shape our schema

The Guideline defines mandatory + optional fields across supplier, buyer, and line level. The **identity fields we must capture now** (already in `DATABASE_SCHEMA.md`) include:

- **Supplier & buyer:** name, **TIN** (Tax Identification Number), registration/

identification number (BRN/NRIC/passport/army), **SST registration number** (if applicable), **MSIC code** (industry), business activity description, address, contact.

- **Invoice level:** supplier/buyer details, unique document reference, date &

time, currency (+ FX rate if not MYR), document type & version.

- **Line level:** description, quantity, unit price, measurement, **classification

code** (e-Invoice item classification), tax type, tax rate, tax amount, subtotal, total, any discounts/charges.

- **Totals:** total excluding tax, total tax, total including tax, and any

rounding amount.

- **TIN validation:** MyInvois exposes a **Validate/Search Taxpayer TIN** API so

we can verify a TIN + registered name **before** issuing — used to set `customers.einvoice_ready` / `suppliers` correctly and to avoid rejections.

> ■ The **exact mandatory-vs-optional matrix and the classification code list** must be taken verbatim from the current Specific Guideline in Phase 5 and confirmed with a tax adviser — do not hardcode from memory or blogs.

---

6. Workflow states we must model (already in schema §12)

Submit → **Submitted** → (async validation) → **Valid** / **Invalid** → optionally **Rejected** (by buyer) → **Cancelled** (by supplier).

- **72-hour rule:** after validation, both supplier and buyer may cancel/reject

within 72 hours of validation. After that window, no cancellation via API — a correction requires a **new** credit/debit note. We store `cancel_deadline_at = validated_at + 72h`.

- **QR / validation link:** the validated e-Invoice carries a QR code / validation

URL that must be shown on the visual invoice; we store `qr_payload` / `validation_link`.

- **Retry & idempotency:** failed submissions retry with the **same idempotency

key** so a retry never creates a duplicate submission or a duplicate ledger entry (Accounting Rule 15).

---

7. Authentication & environments (to confirm precisely)

- API access requires an established **taxpayer digital profile** in MyInvois and

authenticated API calls (OAuth2-style token exchange — **exact grant/flow to confirm from the SDK**), plus per-document **digital signing**.

- **Sandbox (preprod)** and **production** are separate systems with separate

credentials, base URLs, and certificates. **Exact base URLs to be read from the SDK during Phase 5** and placed in config only (never hardcoded).

- Intermediary/ERP-provider models exist (acting on behalf of a taxpayer); for

AthenaLab's own use we integrate as the **taxpayer directly**. The multi-tenant SaaS future (Phase 7) may need the intermediary model — noted, not built now.

---

8. Storage, QR & audit requirements (to confirm)

- Validated e-Invoices and their metadata must be **retained** (record-keeping

period per Malaysian tax law — confirm the exact number of years with the tax adviser).

- Store the **signed document**, the **MyInvois UUID**, **validation link/QR**,

and the **full status history** (immutable) — all provided for in schema §12.

---

9. What is explicitly UNRESOLVED / needs professional confirmation

Marked so nobody mistakes this snapshot for compliance:

1 ■■ **Exact current mandatory dates & turnover thresholds** — verify on the LHDN

timeline page; determine AthenaLab's band.

1 ■■ **The precise mandatory field matrix & classification code list** — take

verbatim from the current Specific Guideline.

1 ■■ **Digital certificate type/issuer and signing algorithm** — confirm from the

SDK signing spec.

1 ■■ **Exact sandbox & production base URLs and the auth grant flow** — read from

the SDK, config-only.

1 ■■ **Tax rounding rule** for MYR at line vs document level — confirm with tax

adviser + Specific Guideline.

1. ■■ **SST applicability & rates** for AthenaLab's actual supplies — tax adviser.
2. ■■ **Record-retention period** — tax adviser.
3. ■■ **Self-billed scenarios** that actually apply to AthenaLab — tax adviser.
4. ■■ **Consolidated e-Invoice** rules (for aggregated B2C) — if relevant to

AthenaLab, confirm.

---

10. Phase 5 research plan (ordered)

1. Re-fetch every source in §1; record version + date; pin the Guideline version.
2. Extract the **mandatory field matrix** and **document-type codes** verbatim

into `EINVOICE_INTEGRATION.md` (created in Phase 5).

1. Extract the **API contracts** (submit, get submission, get document details,

cancel, reject, validate/search TIN, QR retrieval) and the **signing** spec.

1. Confirm **sandbox base URL + auth flow**; obtain sandbox credentials & test

certificate (CEO/finance action — not automatable).

1. Build the **MyInvois service abstraction** behind an interface (so a spec

change does not touch the accounting engine).

1. Map Athena documents → UBL 2.1; validate against MyInvois **sandbox**.
2. Walk the full lifecycle in sandbox: submit → valid/invalid → reject → cancel

(within 72h) → retry.

1. Complete `COMPLIANCE_CHECKLIST.md` with a qualified

Malaysian tax adviser **before** switching any org to `production`.

Design consequence already baked in: the MyInvois integration sits behind a **service interface** (`app/integrations/myinvois/`). Changes to MyInvois requirements change the adapter and the mapping, **not** the accounting engine — exactly as the brief requires.

---

11. Milestone 4A dependency note (tax configuration ≠ e-Invoice)

Milestone 4A implemented **accounting** tax configuration and calculation only (configurable SST codes, per-org tax settings, one deterministic tax engine; see `ACCOUNTING_RULES.md` §16). It deliberately implements **none** of this document: no MyInvois authentication or submission, no e-Invoice validation, QR codes, UUIDs, UBL/XML, or digital certificates, and it makes **no compliance claim**.

The single, deliberate hook for a *future* Phase 5 is the `tax_codes.external_classification` column — reserved to hold a MyInvois tax-type / classification code **later**. It is **never used for calculation** in 4A and is not populated by any default. When Phase 5 maps Athena documents → UBL 2.1, this field is where a code's MyInvois classification will attach, keeping the mapping in the adapter rather than the accounting engine.

---

12. Milestone 4B — MyInvois integration foundation (verified official contracts)

> ■■ **Still not a compliance certification.** Athena Accounting is **not** officially approved by LHDN. The contracts below were read from the **official** LHDN MyInvois SDK; the exact UBL field matrix, classification code list and the signing spec must still be confirmed against the current Guidelines and validated in the MyInvois **sandbox** with a real certificate before any production use.

Sources (official LHDN MyInvois SDK), accessed 2026-08-04:

What	Official URL	Verified value			
Base URLs (FAQ)	https://sdk.myinvois.hasil.gov.my/faq/	Sandbox API+Identity `preprod-api.myinvois.hasil.gov.my`, portal `preprod.myinvois.hasil.gov.my`; Production API+Identity `api.myinvois.hasil.gov.my`, portal `myinvois.hasil.gov.my`			
Login as Taxpayer System	https://sdk.myinvois.hasil.gov.my/api/07-login-as-taxpayer-system/	`POST /connect/token`, form-urlencoded, `grant_type=client_credentials`, `scope=InvoicingAPI`; returns `access_token/token_type=Bearer/expires_in` (~3600); reuse token (~60 min)			
Submit Documents	https://sdk.myinvois.hasil.gov.my/einvoicingapi/02-submit-documents/	`POST /api/v1.0/documentsubmissions/`; body `documents[]{format,document(base64),documentHash(SHA256),codeNumber}`; 202 → `submissionUID`, `acceptedDocuments[{uuid,invoiceCodeNumber}]`, `rejectedDocuments`; ≤5 MB / ≤100 docs / ≤300 KB per doc			
Get Submission	https://sdk.myinvois.hasil.gov.my/einvoicingapi/06-get-submission/	`GET /api/v1.0/documentsubmissions/{submissionUid}`; `overallStatus ∈ {in progress, valid, partially valid, invalid}`; per-doc `status ∈ {Submitted, Valid, Invalid, Cancelled}` with `longId`, `dateTimeValidated`; poll 3–5 s			
Validate Taxpayer TIN	https://sdk.myinvois.hasil.gov.my/einvoicingapi/01-validate-taxpayer-tin/	`GET /api/v1.0/taxpayer/validate/{tin}?idType={NRIC\	PASSPORT\	BRN\	ARMY}&idValue=…`; 200 valid, 400/404 invalid; cache results
Cancel Document	https://sdk.myinvois.hasil.gov.my/einvoicingapi/03-cancel-document/	`PUT /api/v1.0/documents/state/{uuid}/state`; body `{status:"cancelled", reason(≤300)}`; only within **72 h** of validation			

Architecture. One isolated layer, `app/services/myinvois/*`; **every** MyInvois HTTP call passes through the `client.py` transport seam (production `HttpxTransport`; tests install a fake — no test hits the network). Base URLs come from `config.py` (env-var overridable), never hardcoded at call sites.

OAuth. `auth.py` caches a client-credentials token per (org, environment) and refreshes it just before expiry; `api.py` injects the bearer token and transparently re-authenticates **once** on a 401. Secrets and tokens are **never** logged.

Submission flow. `DRAFT` → `READY` → `SUBMITTED` → `VALIDATED / (INVALID | REJECTED)` → `CANCELLED`. States are never skipped and drafts are never auto-submitted — a user initiates submission. Local validation (`validate.py`) runs first; the deterministic mapper (`mapping.py`) builds the UBL-shaped JSON from the approved invoice + settings + customer, reusing the **tax-engine snapshot** on each line (tax is never recomputed). The document is base64-encoded with a SHA-256 `documentHash`.

Status lifecycle & QR. `status.py` polls Get Submission; on the first *Valid* it records `validated_at`, the `longId`, the 72-hour `cancel_deadline_at`, and builds the validation link + QR payload `{portal}/{uuid}/share/{longId}`. `qrcode.py` renders the QR as SVG (via `segno`) **only** from validated identifiers — never a fabricated QR.

Retry policy. `queue.py` retries **only** transient failures (network, HTTP 429/5xx); validation failures, invalid data and duplicate submissions are **never** retried. Retry count is configurable (`ATHENA_ACCT_MYINVOIS_MAX_RETRIES`, default 5) with a fixed backoff. There is no always-on worker; due retries run when the retry-queue endpoint is triggered.

Security. `client_secret` is stored but **never** returned by the API (the settings response only reports `client_secret_set`), never audited, never logged. Every raw API response is stored append-only in `myinvois_responses` (responses carry no secrets).

Known limitation — digital signing (documented gap). MyInvois requires each document to be XAdES-signed with an X.509 certificate. This foundation ships a **pluggable signer seam** with a `PassthroughSigner` (no cryptographic signature — sandbox/testing only) and **blocks PRODUCTION submission** unless a real signer is registered. Implementing + verifying real XAdES signing requires a genuine certificate that only AthenaLab can provide, so it is a **TODO for a later step**, not claimed here.

Not in 4B (deferred): self-billed document workflows, credit/debit/refund note mapping, batch optimisation for consolidated e-Invoices, the full mandatory-field matrix and classification code list verbatim from the Specific Guideline, PDF/print rendering, and the intermediary (act-on-behalf) model for the multi-tenant SaaS future.

---

13. Milestone 4B-2 — production signing & certification readiness

> ■■ **Still not certification.** This milestone builds the *production-grade signing framework and readiness tooling*. Athena Accounting is **not** LHDN-certified or approved, and this claims neither. Real onboarding still needs a genuine LHDN-recognised certificate and a passed sandbox walkthrough.

Verified signing spec (official LHDN MyInvois SDK, accessed 2026-08-04):

What	Official URL
Document Signature Creation	https://sdk.myinvois.hasil.gov.my/signature-creation/
Signing a JSON document (7 steps)	https://sdk.myinvois.hasil.gov.my/signature-creation-json/
Signature structure	https://sdk.myinvois.hasil.gov.my/signature/

What	Official URL
Digital Signature User Guide (PDF)	https://sdk.myinvois.hasil.gov.my/files/Digital_Signature_User_Guide.pdf

Verified **7-step** JSON signing → implemented in `signing.py::ProductionSigner`:

1. Transform: drop `UBLExtensions`/`Signature`, **minify**.
2. `DocDigest` = Base64(SHA-256(minified document)).
3. `Sig` = Base64(RSA-SHA256, PKCS#1 v1.5) over the document, with the certificate **private key**.
4. `CertDigest` = Base64(SHA-256(certificate DER)).
5. `SignedProperties`: `SigningTime`, `SigningCertificate → Cert { CertDigest, IssuerSerial

{ X509IssuerName, X509SerialNumber }, X509Certificate }`.

1. `PropsDigest` = Base64(SHA-256(minified SignedProperties)).
2. Assemble into `UBLExtensions → UBLExtension → ExtensionContent → UBLDocumentSignatures →

SignatureInformation → Signature`; algorithm` http://www.w3.org/2001/04/xmldsig-more#rsa-sha256`.

Certificate architecture (`certificates.py`, `cert_service.py`): the **public** certificate

- metadata (subject/issuer/serial/thumbprint/validity) live in `myinvois_certificates` with

full **rotation history** (a new active record is added; the previous is retained and linked — never overwritten). The **private key is never in the database, never returned, never logged** — it is configured out-of-band via env/files (`ATHENA_ACCT_MYINVOIS_KEY_PEM/_PATH`, `ATHENA_ACCT_MYINVOIS_CERT_PEM/_PATH`, `ATHENA_ACCT_MYINVOIS_KEY_PASSWORD`, optional `_<ENVIRONMENT>` suffix). Uploads are re-serialised to a public PEM so a pasted private key can never be persisted. Expiry monitoring warns at 90 / 60 / 30 days and on expiry, and audits it.

Signing framework (`signing.py`): interface-driven — `BaseSigner`, `FakeSigner` (no cryptographic signature; sandbox/testing only), `ProductionSigner` (the real algorithm). `sign_document(db, settings, json)` is the single entry point; it resolves signing material for a real signer and raises a **friendly** message (never a stack trace) when the certificate/key is missing or unusable. **It never fakes a signature** — with no usable material, submission fails clearly. Production-environment submission is blocked unless the production signer is active (`require_signing_ready`). `verify_signature(...)` re-hashes the document and checks the RSA signature against the embedded certificate (used in tests and available for validation).

Health checks (`health.py`): authentication, token refresh, certificate, sandbox/production connectivity, and the TIN/submission/status endpoints — each returns Healthy/Warning/Failed with a latency and **never crashes** the dashboard. **Submission monitor** (`monitor.py`): MyInvois health only (no financial metrics). **Sandbox certification readiness** (`certification.py`): a checklist derived from real signals; completing it is a prerequisite for onboarding — **not** a certification.

Security. API secrets are optionally **encrypted at rest** (Fernet via `ATHENA_ACCT_MYINVOIS_SECRET_KEY`), always write-only at the API boundary, never logged. Private keys are isolated to configuration and never touch the DB, the API or logs.

Still deferred (TODO, out of 4B-2): onboarding a real LHDN-recognised certificate + a live sandbox certification walkthrough; the exact official signature envelope confirmed against a live sandbox validation; hardware-security-module / external key-store integration.