Review, release, revision and the register

A second person's signature on the content, issue under the logged number with two hashes of the report's own data, publication whatever the result, and correction by a new report that leaves the old one reachable, marked retracted.

Not yet approved
No approval of this revision is on file. This revision states what the laboratory's results system enforces today; it takes effect when a named reviewer approves it.

Document		Laboratory	
Number	SOP-07	Name	Cupel Laboratories
Revision	1, first issue	Address	2599 Speakman Dr Mississauga, ON L5K 1B1
Status	Not yet approved		
Effective	Not yet in effect; takes effect when approved	Country	Canada
		Not on file	Not yet published: legal entity, postal address, telephone and country
Applies to	Every report, every revision		
Supersedes	Nothing; first issue	Web	cupel.ca
Fingerprint	3917 563f 365b 4e43	Email	help@cupel.ca

1 Purpose

To release a report only after a second person has signed the content as it stands, to issue it under the number its sample was logged under with two hashes of its own data, to publish it under that number whatever it says, and to correct it only by a new report that supersedes the old, which stays reachable, marked retracted.

2 Scope

Every result set from validation to publication, every revision, the register (register.db) and the public site built from it.

Measurements carry uncertainty. When an error is found in an issued report, a correction is published as a new record, the original is marked retracted rather than deleted, and both remain reachable at their original addresses for the life of the register.

3 Responsibilities

Role	Responsibility
Analyst	The person who created the set or is named in its sign-off, with a title. Enters the data; cannot review it.
Reviewer	A second person, under a different name (case is not a difference), with a title. Signs the content as reviewed and releases it; the review page refuses the analyst, an unnamed analyst and a blank "Working as".
Both	Their names, titles and signing times are printed in the signature block on page 1, for and on behalf of the laboratory.

Named: Timothy Ian Benedicto, Medical Laboratory Technician; Roxanne Manlapaz, Laboratory Technician.

4 References and controlled sources

Standards and other documents

- RUNBOOK.md sections 6 (validate), 7 (review), 8 (issue), 10 (revision), 12 (build and publish), 13 (backup) and 15 (the examples).

References and controlled sources, continued

– SOP-01 (logging, the clock, withdrawal), SOP-02 to SOP-06 (what each test must carry to issue), SOP-08 (complaints and corrections).

Controlled sources, read at every build

– data.json integrity, verify, limitations, counts; terms (Publication).
– lab/CONTRACT.md: “Functions lab/results.py MUST expose” (validate), “Set by entry_server”, “The accession is assigned at logging”.
– register.py: issue(), correct(), retract(), record_text(), public(), lookup(); damm.py: the check digit.
– lab/entry_server.py: review, issue, revise, the content hash, the audit trail.

5 Validate

Validate saves first and shows three blocks. Fix, save and validate again until the first two are empty.

– Problems (they block issue, and some block review): a status outside draft, reviewed, issued, withdrawn; a formula that does not parse; a Received time that is not ISO-8601 UTC; the test-specific rules of SOP-02 to SOP-06 (exactly three injections with one main peak each, a cycle that adds up, a criterion the instrument class cannot support, lambda not the lowest standard, fewer than two water replicates, fewer than three calibration levels).
– Refused at issue, checked now so they are seen early: an instrument identifier for every section; a data file and an acquisition time for every figure (every HPLC injection; LC–MS, endotoxin and mass per vial once); the baseline treatment; an acquired trace on every injection; the instrument class.
– Worth checking (they do not block): a suitability criterion missed, a spike off the mid-curve, a replicate CV above its limit, a mass balance that does not close, missing lots or titles. The reviewer reads them.

6 Review

6.1 The reviewer reads the Preview PDF and the Validate page, then presses Review and Sign as reviewer. The server records the time, sets the status to reviewed and stores a hash of the content as reviewed.

6.2 The content hash is the SHA-256 of the set with status, reviewer, accession, sha256, record_sha256, issued_utc, revision_reason, revised_from removed, and inside the sample block holds, guarantee removed: the turnaround bookkeeping is not analytical content and never voids a review.

6.3 Any edit to the content after signing returns the set to draft, clears the signature and says so in the audit trail; review again. A hold or a resume does not. A review signed before the content hash existed is refused at issue with “review again”.

6.4 A reviewer may sign while problems remain, but the set cannot issue until validate is clean.

7 Issue

Issue is final: the register is append-only and there is no undo. A mistake is handled by a revision (Revision and correction, below, and SOP-08).

The checklist

– No validate problems, and none of the refused-at-issue rules outstanding.
– Analyst and reviewer named, different people, status reviewed, content unchanged since the review.
– The sample logged: an accession reserved at logging, its ledger row still “logged” and logged for this set. Received and matrix set; analyte name and formula present.
– Where a conclusion on page 1 is adverse, the notice to the sender on record (below).

Issue, continued

The notice before publication

A report whose page-1 conclusion is “Below claim”, “Does not meet claim”, “Does not conform” or “Not confirmed” is told to the sender before it is published. On the set’s Notice page (/notice/<id>) the backend prepares “Email 3b: told before publication” from the reviewed content and, when it is recorded, writes a “notice” entry to the audit trail with the hash of the content as it stands. Issue is refused until the notice is on record, and again when the content changes after it, until it is recorded again. Send the notice from the laboratory mailbox when it is recorded, as every email (SOP-01, Correspondence with the sender). The period between the notice and publication is the one the site states; while data.json publication.notice_days is not set it is the notice email’s own honest line, and the notice and the report may go on the same day:

> We write to you on the day the report is published.
>
> lab/emails.py, the notice email, while publication.notice_days is not set

The transaction

7.1 Any hold still open is closed with the answer “closed at issue without an answer”.

7.2 The clock is read against the issue time, and the guarantee result is written into the set: late days, and whether the report is free.

7.3 The record hash of the complete result set is computed.

7.4 The certificate is composed once as a dry run.

7.5 In one database transaction the register row is written under the accession reserved at logging, with both hashes, and the ledger row closes as issued. The register refuses a number that was never logged or is already closed, and then nothing is written.

7.6 The accession, the hashes, the issue time and status issued are written into the set and it is locked; an edit of an issued set is refused.

7.7 The issue line is appended to the audit trail with the accession and both hashes; “Email 4: report issued” (“Email 5: correction issued” for a revision) is prepared; the certificate and record.json are written under site/r/<accession>/.

The sender receives the email from the laboratory mailbox with the certificate attached. A late report is sent the same way, with the fee cancelled.

8 The two hashes and the record

- SHA-256 is the hash algorithm (data.json verify). Both hashes are computed by the register at issue, never supplied; a caller’s value that does not match is refused.
- The summary hash (printed as SHA-256) is of payload.json: the public projection of the report (accession, received_utc, issued_utc, specimen, submitter_ref, analyte, cas, formula, matrix, method, result, replicates, figure, impurities, unresolved, analyst, label_claim, submitter_kind, plain_terms, record_sha256, record_fields), serialised as byte-stable JSON with keys sorted, no whitespace, UTF-8.

The two hashes and the record, continued

- The record hash (printed as Record) is of record.json: the complete result set, every trace point, peak, titration, onset and signature. It leaves out only the register's own bookkeeping, which the register holds itself: the accession, the issue time, the status, the summary hash, the record hash, the revision reason, the number a revision supersedes, the reviewer's content hash, analyst.person, reviewer.person, authorised.person, reviewer.decisions, sample.photos, text.plain_terms_override and correction.changes (accession, issued_utc, status, sha256, record_sha256, revision_reason, revised_from, reviewer.content_sha256, analyst.person, reviewer.person, authorised.person, reviewer.decisions, sample.photos, text.plain_terms_override, correction.changes); the order linkage, the sender's side of the file, which the site promises never to publish: the order reference, the sample's place in the order and the order's size (sample.order_ref, sample.order_sample, sample.order_qty). Published, a shared reference would let anyone group the lots one sender shipped together across reports, and the size would say how much they sent; the link lives in the set's audit trail and the order ledger, neither of which is published; the words of every hold: of each the record keeps only when it was opened, when it was closed and its kind (opened_utc, closed_utc, kind), so the pauses of the turnaround clock can be checked against the dates on the certificate, and never the reason for asking, the question sent or the answer received. The count of leaf values it covers is stored beside it.
- Both are printed in the signature block beside the QR, and both files are served beside the certificate at /r/<accession>/. Recomputing a hash over the served file shows whether that file is the one issued; the PDF in hand is checked by its own fingerprint, which the register records at issue (register.py record_file) and /verify looks up, or by comparing its figures with the page. Neither vouches for the measurement.
- The build refuses to publish when either served file does not hash to what the register holds; an issued set file changed by a byte fails the record-hash guard, and is restored from backup, never edited.

9 Revision and correction

9.1 Open the issued set and press Revise; enter what was wrong and what the revision changes. A draft opens with the next revision number and "supersedes" set to the original's number. Only one revision of a report can be open at a time.

9.2 Correct the data, validate, and have a second person review. A revision is a second report on a sample logged once, so nothing is reserved for it: the draft shows no Log receipt and no Withdraw.

9.3 Press Issue. In one transaction the original is marked retracted with the reason as its public note, the new report takes the next serial of the one sequence samples and reports share (minted at issue, the one case where a number is), with supersedes set, and the original is linked forward to it.

9.4 "Email 5: correction issued" is prepared with the previous figures beside the new ones; send it with the new certificate attached.

9.5 Rebuild and publish. The corrections log lists the pair; the retracted certificate stays at its address as issued, its page says first that it is retracted and links its replacement, and the lookup answers the number as retracted. The PDF itself is not re-issued: the retraction is on the page, in the register and in the lookup, not on the paper.

A report's content cannot be changed after issue: the register refuses any update of accession, received_utc, issued_utc, specimen, submitter_ref, analyte, cas, formula, matrix, method, result, replicates, figure, impurities, unresolved, analyst, label_claim, submitter_kind, plain_terms, record_sha256, record_fields, payload, payload_sha256, supersedes. Only the retraction time and note and the forward link may ever be written, each once. Nothing is deleted, by the laboratory or by the customer who paid; a result the customer dislikes is published the same way as one they like. SOP-08 lists every guard the register holds.

10 The register and the public lookup

- The register is an append-only SQLite file, register.db. It holds every issued report with both hashes, every retraction and the sample ledger (SOP-01). Its schema is tested on every build.
- Accession numbers are CL-YYYY-NNNNN-C with a Damm, order 10 check digit, which catches a wrong or swapped digit before the register is touched; it is not a defence against forgery, and the site does not call it one. Serials 99900–99999 are reserved for the worked examples, which say so on their face and are counted nowhere.
- The lookup at /verify needs no account and no email. What it answers, by state:

State	What the lookup answers
malformed	Not in the form of an accession number.
bad_check_digit	Well formed, but the check digit is wrong: a copying mistake, caught before the register is read.
absent	Well formed with a valid check digit, but not a number this laboratory has logged or issued.
logged	A sample is logged under the number and its report is not yet issued; the answer gives the logged date and analyte.
withdrawn	The number was logged and closed with no report, and none will be issued.
found	An issued report: the certificate, its data, both hashes.
retracted	An issued report that was retracted; the answer says so first and links its correction.

- Published for each report: the certificate, payload.json and record.json, the headline, the plain-terms paragraph, the method, the replicates, the analyst, the lot reference as given. Never published: the sender's name, company or contact, the set id, a withdrawal reason.

11 Building and publishing

- Every page, certificate page, data file and PDF on the public site is written by one command from data.json, the register and the result sets; nothing on the site is edited by hand.
- The build first runs its proof suites (mass arithmetic, check digit, register schema, Python and JavaScript parity, contrast, the certificate PDF audit), then writes every page, then runs the layout probe. A failed build writes nothing new; do not publish after one.
- The counts the pages print are read from the register at build, examples excluded; data.json keeps only a cache of them, and a new laboratory's honest count is zero.
- After uploading, open the new certificate page on the live site, press "Recompute this hash in your browser" and read "Matches"; look the number up on /verify.

12 Records and retention

- What cannot be regenerated and is never deleted: register.db; lab/results (every set, audit trail and email); the case logs (lab/records/complaints.jsonl, lab/records/capa.jsonl); the certificate and record.json as issued under site/r. Back up daily and immediately after every issue (python lab/backup.py backup makes a verified archive).
- Every copy of a backup sits on the project's own drive until data.json backup.offsite_path names a folder on another disk (lab/backup.py offsite_path; python lab/backup.py backup --to <folder> copies one archive there and checks its hash). The custody rule for copies that leave the laboratory is written here the day one does.
- The interlaboratory comparison log (lab/records/ilc.jsonl, chained like the case logs; it does not exist yet) and the partner laboratory's reports with their hashes (SOP-11) are kept for the life of the laboratory with the result sets they compare, never deleted; a comparison later found flawed stays with a note, as a retracted report does.
- What the build regenerates: everything else under site, payload.json included.

Records and retention, continued

- The audit trail of a set is append-only: time, name, action, note and the field-by-field difference, for create, edit, validate, logged, review, issue, revise, hold, resume, email, withdraw, labels, export, notice, rebuild, destroy, import, qc_limits, retract, stop, release, photo, deviation.
- Each entry carries the SHA-256 of the entry before it; the history page checks the chain and names the first broken line, if any.
- One set's whole record (the set, its trail, its emails, the certificate and record files, the register's rows, a manifest of hashes) can be exported as a zip for an inspector; the export is itself an entry in the trail.
- Email correspondence is kept as long as is needed to answer it and to invoice, and no longer; it is not linked to any published record.
- The bench worksheets (WS-01 to WS-08, one per record the results need) are the original record of every value measured away from the backend: filled in ink at the moment of measurement, each line initialled and timed; a correction is a single line through, the new value, initials, date and reason; nothing is overwritten or erased. The completed worksheet is photographed or scanned the same day and attached to the set through the backend's photo record (/photo/<id>/<name>, each file hashed and a "photo" entry in the trail), and the values are typed into the form from it; the reviewer checks the form against the attached pages line by line before signing.
- "Working as" is a browser cookie on a shared computer, and the second person who sits down inherits the first one's name: before every save, read the header name and set your own. A save under another person's name is corrected by a note entry under the right name the same day, and is a deviation.

13 Integrity commitments

As the Why Cupel page publishes them, to the sender and the buyer; each is built into the register, not into a policy:

- We do not delete reports, for ourselves or for the customer who paid for one. The register is built to refuse a deletion and to show one at its next integrity check; a result you dislike is published the same way as one you like.
- Corrections are issued as new reports. The superseded report stays reachable at its original address, marked retracted, linking forward to what replaced it.
- Every report carries a SHA-256 of its own data, printed on the certificate, and the data is published beside it. Recomputing the hash shows whether the published data is the data the number was issued with; the certificate file itself is checked on the verify page against the fingerprint the register recorded when it issued, and the figures on any paper against the published report. It does not vouch for us: we publish the report, the data and the hash, so it cannot show that we measured correctly.
- Measurement is our only line of business: we undertake to hold no inventory, to take no affiliate payment and no reseller margin, and to say so on this page if that ever changes. That also means nothing subsidises the price list, so we cannot undercut a laboratory that sells what it tests.
- We do not accept payment for a favourable result, and we do not offer vendors an embargo period, a pre-review, or a chance to withdraw a sample after the data exist.

14 Deviations and open points

- A departure from this procedure, planned or found afterwards, is nonconforming work and is handled under SOP-08. One that bears on a result is stated on the certificate, in the open point or the notes of the section concerned, never left out.
- Cupel holds no ISO/IEC 17025 accreditation; results are not accredited results.
- The three example reports and the one example logged number stay in the register at launch; nothing in it is removed. The first real report takes over the hero card and every link.

Deviations and open points, continued

- The backend holds no field for a date agreed above capacity (SOP-01).
- The integrity checks run with python make.py check (register, test_analytes_reference, test_cds_import, test_e2e, test_emails, test_launch_facts, test_launch_preview, test_management_review, test_metrics, test_orders, test_qc_trend, test_report_api, test_results, test_safety, test_sop, test_tracking, test_validation, test_vendor_kit, test_worksheets, selftest, build, integrity, integrity_selftest, backup_selftest); lab/integrity.py re-verifies the register and the served files from disk.

15 Revision history

Revision	Change	Approval
1	First issue.	Not yet approved

The fingerprint in the control block is the SHA-256 of this document's text as composed, first 16 hex digits. A change in any controlled source changes it; a copy whose fingerprint differs from the approved one is not the approved text, and a change is approved as a new revision.

Cupel holds no ISO/IEC 17025 accreditation; results are not accredited results.

This document is published so that a chemist can judge how a result was produced and what it can and cannot show; it is not an accredited procedure and claims no conformity with any standard it cites.

The current revision of every laboratory document is the one published at cupel.ca/methods. A printed copy is uncontrolled: compare its fingerprint with the published one.

Current revisions

End of document