S A N A D

Sanad v0.1: an open protocol for recomputable recognition of skills

Specification · Spezifikation

Document	**SANAD-SPEC-0.1**
Status	**Entwurf / Draft for discussion**
Version	**0.1 (draft)**
Date	**7 November 2026**
Licence	**CC BY 4.0**
Steward	**Kermani Kolankeh, Arash**
URL	**https://sanadprotocol.org/spec/0.1/**

This draft has not been reviewed by a standards body. It is not accreditation.

> This draft has not been reviewed by a standards body, a lawyer or an external cryptographer. Sanad (a short word for a document and the chain of support behind it) is to be stewarded by the Open Sanad Foundation (UAE, in preparation); until it exists the author acts as interim steward. Nothing here is accreditation: Sanad practises **decentralised assessment**. It does not accredit anyone and it does not decide what a verifier must accept. A reference implementation with conformance vectors (`reference/` in the project repository) and an explainer (hosted by the Kamangir project for now) accompany this draft. The key words MUST, SHOULD and MAY are used as in RFC 2119 and RFC 8174.

---

0. In one page

A qualification is not a certificate; it is a **chain**: a *claim* about a skill, backed by *evidence*, which is judged by *attestors* who sign what they saw and declare their conflicts of interest. Anyone can fetch the chain and run the same published function, the **recognition function**, with **their own policy**, and get a level, a confidence and a step-by-step explanation. Nothing has to be taken on trust from the issuer; what a verifier does have to choose is which attestors, and how much evidence, it requires. The protocol has no coin and no central authority. Bitcoin's lesson that transfers is *verification by recomputation without a gatekeeper*; its scarcity does not transfer, because the scarce resource here is credible assessment.

Non-goals: ranking people; proving honesty (a person can still be helped off-camera); replacing legal recognition (degrees remain what states and institutions say they are); defining a payment system.

0.1 Scope and normative references

Scope. This document specifies the data model, the canonical form, the recognition function, the transparency log and the conformance rules. Sections 0, 1, 7, 10 (except the no-ranking rule), 12 and 14 are informative; sections 3, 4, 5, 6, 8 (log and proofs), 9 (conflict declaration), 10 (no-ranking rule) and 11 are normative.
Normative references (sources in Appendix B): RFC 2119 and RFC 8174 (key words); RFC 8785 (JSON Canonicalization Scheme: the canonical JSON of section 4 is that scheme); RFC 8032 (EdDSA); RFC 7515 (JWS); W3C Verifiable Credentials Data Model 2.0; W3C Decentralized Identifiers; W3C Bitstring Status List; FIPS 180-4 (SHA-256).

1. Principles

1. **Recomputable.** Every recognised level MUST be derivable by a deterministic public function from public or holder-disclosed data. No hidden scores.
2. **Verifier-side policy.** The issuer does not say "recognised". The verifier applies its own policy to the same evidence. Two verifiers may legitimately disagree; both can show why.
3. **Explainable.** The function returns an ordered explanation: each rule, its inputs, its effect.
4. **Evidence is typed and graded by independence.** Self-run practice is worth less than a sitting watched by independent examiners.
5. **No token, no speculation, no ranking.** Matching is opt-in and never ranks people.
6. **Privacy by default.** The holder controls what is published. Inferred skills are private until the holder publishes them.
7. **Honest wording.** Statuses follow a published stage ladder (stage 0, a self-declared keepsake, to stage 5, an award by an institution with the legal authority to make it); a Sanad result never uses the words accredited or certified by Sanad.
8. **Built on existing standards.** W3C Verifiable Credentials 2.0, Open Badges 3.0 (AchievementCredential), W3C DIDs, Bitstring Status List, SD-JWT VC for selective disclosure; the skill vocabulary is crosswalked to ESCO, MSC2020 and EQF.

2. Terms

- **Skill**: a node of the open concept graph (a versioned, hash-pinned directed acyclic graph of skills), id like `analysis.limits.epsilon-delta`, with crosswalks `{esco?, msc?, eqf?}`.
- **Level**: integer 0..5: 0 none, 1 aware, 2 guided, 3 independent, 4 fluent and transfers to new problems, 5 expert (can teach and extend).
- **Claim**: "subject S has skill K at level L as of date D".
- **Evidence**: a typed record backing a claim (section 3.3).
- **Attestation**: a signed statement by an **attestor** about one evidence item (section 3.4).
- **Policy**: the verifier's rules (section 3.5).
- **Recognition result**: output of the function (section 5).
- **Transparency log**: an append-only Merkle log of attestation hashes (section 8).

3. Data model (JSON, canonicalised per section 4)

3.1 Skill

`{ "id": "analysis.limits.epsilon-delta", "title": "...", "prereq": ["analysis.limits.intuitive", "logic.quantifiers"], "xwalk": { "msc": "26A03", "eqf": 6 } }`. The graph MUST be a DAG; the recognition function uses a pinned graph version hash (`graphHash`).

3.2 Claim

`{ "v": "sanad-0.1", "id": "<128-bit hex>", "subject": "urn:uuid:...", "skill": "<skill id>", "level": 4, "claimedOn": "2026-11-01", "graphHash": "<sha256>" }`. The subject id is pseudonymous; a name is added only with the holder's consent.

3.3 Evidence

Common: `{ "id", "kind": "auto|sitting|artifact|peer", "claimId", "takenOn", "hash" }`. By kind:

- **auto**: seeded exercise runs: `{ "seedSet": ["K7Q", ...], "level": 3, "results": [{ "seed", "ok", "hints", "tries" }] }`. Replayable: anyone can regenerate the exercises from the seeds and re-check the answers. Self-run, so independence is low.
- **sitting**: a live sitting: `{ "sittingId", "paperFingerprint", "strokeLogHead", "recordingSha256?", "examinerAttestations": [ids] }`. The stroke log is a SHA-256 hash chain over the pen strokes written during the sitting.
- **artifact**: `{ "artifactSha256", "description" }` (a proof, a program, a paper), reviewed by an attestor.
- **peer**: a peer review by a learner of equal or higher level.

3.4 Attestation

`{ "id", "evidenceId", "attestor": "did:...", "role": "examiner|registrar|reviewer|peer", "level": 0-5, "conflictDeclared": false, "date", "commit": "<sha256 of level|nonce>", "reveal": "<nonce>", "sig": "<JWS>" }`. Marking is **blind commit-and-reveal**: each attestor publishes `commit` before any reveal, so that marks are independent. `sig` is a JWS (EdDSA) over the canonical attestation. (In v0.1 reference vectors, `sig` is a placeholder string; real verification is section 4.3.)
Attestor statistics are derived, not self-declared: `{ "sittings": n, "agreement":`

`0..1 }` from the transparency log (agreement = share of that attestor's marks within one level of the median of the other attestors on the same sittings).

3.5 Policy (chosen by the verifier)

`{ "minAttestors": 2, "requireLive": true, "halfLifeMonths": 24, "minAttestorSittings": 3, "minAgreement": 0.7, "maxInferenceDepth": 2, "inferenceDecay": 0.7, "asOf": "2026-11-07" }`. A policy is itself a small public document with an id, so a verifier can state "recognised under policy `uni-strict-1`" and anyone can recompute.

---

4. Canonical form, hashes, signatures

1. **Canonical JSON**: the JSON Canonicalization Scheme of RFC 8785 (UTF-8, object keys sorted, no insignificant whitespace, numbers as in ECMAScript JSON, no duplicate keys).
2. **Hash**: SHA-256 of the canonical bytes, hex. Evidence and attestation `hash` fields cover the object without the `hash` and `sig` fields.
3. **Signatures**: attestors sign with Ed25519 keys published in a DID document (`did:web` or `did:key`). Key rotation and revocation use the DID document and a Bitstring Status List.
4. **Chain**: a claim lists evidence ids; evidence lists attestation ids; the credential carries the root hash of this tree (`bundleHash`). Altering any mark changes `bundleHash`.

---

5. The recognition function (normative)

`recognise(policy, credential, attestorStats, graph) -> Result`. The function MUST be deterministic: inputs are sorted by id before use, there is no randomness and no clock (the evaluation date is `policy.asOf`). Version string: `sanad-0.1`. Arithmetic is IEEE-754 binary64; numeric results are rounded to 3 decimals, and an implementation conforms if every numeric result is within 0.001 of the value in the test vectors (this removes dependence on a particular maths library). Ages are measured in months of 30.436875 days (365.2425 / 12) between the evidence date and `policy.asOf`.

Definitions used below. *Human attestor*: an attestor with a DID other than the pseudo-attestor `replay`. *Qualifying evidence*: evidence of kind `sitting` or `artifact` (the kinds that need independent human attestors). *Admissible*: not dropped by R1 or R2.

- **R1 integrity**: drop any evidence or attestation whose hash, commit-reveal check or chain link fails. A chain link fails when an attestation's `prev` is not the stored hash of

its predecessor; that attestation, and therefore also its honest successor, is dropped (a conservative rule). Record each drop with its reason.

- **R2 admissibility**: drop an attestation if `conflictDeclared`; or, for a human attestor, if the attestor has fewer than `minAttestorSittings` sittings or agreement below `minAgreement` (a missing entry in the attestor statistics counts as 0 sittings and 0 agreement); drop evidence older than 4 half-lives, and evidence dated after `policy.asOf`.
- **R3 age factor**: `a = 0.5 ^ (ageMonths / halfLifeMonths)` for each evidence item.
- **R4 per-evidence level**: lower median of the admissible attested levels for that evidence item, capped by the kind (auto 3, peer 3, artifact 4, sitting 5). Automatic records (`auto`) have no human attestor: their replayed level is stored in the hashed evidence body (`autoLevel`) and is treated as one pseudo-attestation by `replay`, exempt from the attestor thresholds of R2 and not counted in R5.
- **R5 independence**: let `n` be the number of distinct admissible human attestors on qualifying evidence. Peer reviewers do not count in `n`. If `n < minAttestors`, or `requireLive` and there is no admissible `sitting`, the status is `provisional` and the level is limited to the best automatic or peer level (0 if there is none; at most 3). Otherwise the status is `recognised`. Automatic and peer evidence alone never lead to `recognised`. The result also carries `attestedLevel`: the level R6 would give if the policy were met, so that a reader can see what was attested although the policy was not satisfied.
- **R6 level**: `recognisedLevel = min(claimedLevel, lower median of the per-evidence levels of the qualifying evidence)`.
- **R7 confidence**: for each admissible evidence item and each of its admissible attestations, form a term `w x a x g`, where `w` is the weight of the kind (sitting 0.6 per attestor, artifact 0.3, auto 0.25, peer 0.15), `a` the age factor of R3, and `g` an agreement factor. The spread of the admissible attested levels of **that evidence item** (maximum minus minimum) decides `g`: `g = 1` if the spread is at most 1, otherwise the weights of that evidence item are halved (`g = 0.5`). Confidence = `1 - product(1 - term)` over all terms, rounded to 3 decimals. With no admissible evidence the status is `not-recognised` and the confidence is 0.
- **R8 inference** (hidden skills): for each prerequisite `p` of the recognised skill at graph distance `d <= maxInferenceDepth` (shortest backward path), and with no direct claim: `inferredLevel = max(0, level - d)` and `inferredConfidence = confidence * decay^d`. Inferred skills are always labelled INFERRED, carry their derivation path, and MUST NOT be published without the holder's choice.
  The result is `{ "alg": "sanad-0.1", "policyId", "status", "level", "attestedLevel", "confidence", "explanation": [{ "rule", "summary",`

`"inputs", "effect" }], "inferred": [...] }`. The explanation is part of the result so that a human can audit it line by line. The agreement statistic of an attestor used in R2 is published with the transparency log (section 8); it is an input of the function, so verifiers can pin the statistics they use. Conformance is tested by `vectors.json` (section 11).

---

6. Carrying a Sanad bundle in a credential

A Sanad-aware credential is a VC 2.0 / Open Badges 3.0 AchievementCredential with `evidence` entries of type `SanadBundle`: `{ "type": ["Evidence", "SanadBundle"], "id": "<URL or IPFS-free static URL>", "bundleHash": "...", "alg": "sanad-0.1" }`. The issuer field states the stage-ladder status (issuers may define their own profile of the stage ladder). A verifier fetches the bundle, recomputes `bundleHash`, runs the recognition function with its own policy and shows the explanation. A credential without a bundle is a stage-0 keepsake.

---

7. Skill graph, crosswalks, inference

The concept graph is versioned and hash-pinned. Crosswalks to ESCO, MSC2020 and EQF are advisory metadata: they let other systems index the same claim; they are not used by the function. Inference follows prerequisite edges only; it never invents a skill outside the graph and never raises a level above the directly demonstrated one.

---

8. Transparency log and anchoring

Attestation hashes are appended to an append-only Merkle log run by any party (the Foundation runs one; others may). A **checkpoint** (tree size, root hash, timestamp, signature) is published daily. A holder or verifier can request an **inclusion proof** that an attestation was in the log at a given time, and a **consistency proof** that the log was never rewritten. Optionally the checkpoint root is timestamped on Bitcoin through the free OpenTimestamps service (no token, no fee charged by the project). The log reveals hashes only, never content or names.

---

9. Attestors and trust

Anyone may become an attestor by publishing a DID and signing. **Nobody vets them for the protocol**, and independence is not enforced by it: it rests on declared conflicts of interest, blind marking and the verifier's policy. Verifiers choose by policy (minimum sittings, agreement rate, named allow-lists, or a web of trust). Attestor statistics are public

and recomputable from the log. Conflicts of interest MUST be declared (family, employer, paid tutor, same cohort); an undeclared conflict that is discovered is recorded as a signed dispute against the attestor. Assignment to a sitting SHOULD be random from the pool; marks are blind until all commits are in.

---

10. Privacy, discovery and matching

- **Selective disclosure**: the holder presents an SD-JWT VC revealing only chosen claims, for example "linear algebra at level 4 or above, recognised under policy P", not the sittings or the date.
- **Skill cards**: an opt-in public record `{ skill, level range, policy id, contact route }` signed by the holder. There is no central profile database; indexers (any party) crawl only cards that holders publish.
- **Hidden skills**: computed locally in the holder's browser from the holder's own bundles (R8). Shown to the holder first, marked INFERRED.
- **Mediated introduction**: contact goes to a pseudonymous inbox. Requesters MUST be identified organisations or individuals; rate limits deter spam (how an index enforces them is outside this specification, which defines no payment system); the holder sees who is asking and why before anything is revealed.
- **No ranking**: indexes return sets that satisfy a query, not ordered "best candidates"; protocol-conformant tools MUST NOT produce a score of a person across skills.
- **Law**: data minimisation and consent (GDPR, UAE PDPL; check both); recordings are not part of the public chain (retention D+90 is procedural); EU AI Act: tools that rank or filter job applicants are high-risk, so employment matching needs human oversight, transparency and a legal review before launch.

---

11. Conformance and test vectors

Conformance classes: **Verifier** (runs the function and prints the explanation), **Issuer** (emits bundles and credentials), **Attestor** (signs blind marks), **Log** (serves checkpoints and proofs), **Indexer** (honours the no-ranking rule). `vectors.json` contains the policy, claim, evidence, attestations and expected result for: no evidence; auto only; one examiner when two are needed; two agreeing examiners; disagreeing examiners; a declared conflict; stale evidence; a tampered hash; depth-2 inference; strict versus loose policy over the same evidence. An implementation conforms if its result equals the expected result on every vector, including the ordered rule ids in `explanation`.

12. Threats and mitigations

Threat	Mitigation in v0.1	Residual risk
Collusion of examiners	random assignment; blind commit-reveal; agreement statistics; disputes in the log; policy can require attestors from different institutions	a determined cartel; needs audits
Sybil attestors	no trust by default; policies require sittings and agreement history; verifiers use allow-lists	slow to bootstrap
Candidate cheating (help off camera)	desk camera, room check, human observation; the chain says what was observed, not honesty	cannot be eliminated
Identity (someone else sits)	attestors record how identity was checked; stage ladder wording	weak without ID checks
Tampering	hash chain, signatures, transparency log, optional Bitcoin timestamp	key compromise
Goodhart (teaching to the test)	endless seeded items; sittings use fresh questions	slow
Algorithmic bias in matching	transparent function, no ranking, human oversight, legal review	unknown unknowns
Key loss	holder backup, DID rotation, re-attestation by two attestors	usability
Exclusion by cost	sponsor-a-seat fund; auto and peer evidence remain free (provisional)	funding

13. Governance and evolution

Versioned (`sanad-MAJOR.MINOR`); a change that alters a vector's expected result is a new minor version; the text is frozen per version. The specification text is released under CC BY 4.0 and the reference code under Apache-2.0. The name "Sanad" is intended to be held by the Foundation (interim: the steward, in trust); a trademark application is planned and is not yet filed. Anyone whose implementation passes the vectors may say that it conforms. The path to a standard: public draft, reference implementation, conformance suite, at least three independent implementers, then a W3C Community Group or a 1EdTech workgroup. Sanad does not declare itself a standard; adoption does. Until a Working Group exists, the Sanad Foundation is the steward, under the governance charter published with this specification.

14. Open questions (need a decision or a lawyer)

1. Who runs the first transparency log, and who audits it? 2. Level descriptors 0..5: adopt EQF-aligned wording or keep project-specific wording? 3. How many attestors for a sitting is the minimum honest number (1, 2, 3)? 4. Attestor payment versus independence (flat fee already decided). 5. Appeals: who judges a dispute (ombudsperson) and how is the outcome recorded. 6. Sanctions and access: attestors and holders in sanctioned places (legal advice). 7. Are policy templates published (`school-loose`, `uni-strict`)? 8. Retention of recordings versus a public chain (D+90).

Appendix A: worked example

The example is invented: no person, sitting or attestor in it is real. It is the sample credential of the reference implementation (`reference/src/sample.ts`) and is reproduced by `node reference/verify.mjs` and by the explainer.

Claim: subject `urn:uuid:7c1f0b2e-sample-learner`, skill `B09` (a first course in real analysis), claimed level 4, as of 2026-11-07. Evidence: E1 auto (40 seeded exercises, level 3), E2 live sitting with attestations att-01 (examiner, level 4) and att-02 (examiner, level 4), E3 artifact (three written proofs) with att-03 (reviewer, level 4), E4 peer review with att-04 (level 3). Attestor statistics: examiner 1 has 14 sittings and agreement 0.91, examiner 2 has 9 and 0.88, the reviewer has 6 and 0.80, the peer has 1 and 0.70. Default policy: minAttestors 2, requireLive true, halfLifeMonths 24, minAttestorSittings 3, minAgreement 0.6, maxInferenceDepth 2, inferenceDecay 0.7.

Result: status `recognised`, level 4, confidence 0.909. R1 finds every hash and reveal intact. R2 ignores att-04 (the peer has 1 sitting, the policy needs 3), so E4 does not count. R3 gives age factors 0.950, 0.983 and 0.966 to E1, E2 and E3. R4 gives levels 3 (E1), 4 (E2, lower median of 4 and 4) and 4 (E3). R5 finds three distinct admissible attestors and a live sitting, so independence is met. R6: level = lower of the claim (4) and the lower median of the sitting and artifact levels (4) = 4. R7: confidence = 1 − the product of (1 − weight × age factor × agreement factor) over the four items = 0.909.

Variations the reader can try: with a policy that requires three attestors on a sitting, the status becomes `provisional` and the level is capped at 3; changing one mark after sealing makes R1 drop it and lowers the confidence.

Appendix B: references

Sources as of 7 November 2026. Only status statements that were checked against the source are made.

- RFC 2119 and RFC 8174 (key words): https://www.rfc-editor.org/rfc/rfc2119 <https://www.rfc-editor.org/rfc/rfc2119>, https://www.rfc-editor.org/rfc/rfc8174 <https://www.rfc-editor.org/rfc/rfc8174>.
- RFC 8785, JSON Canonicalization Scheme (Informational, June 2020, independent submission): https://www.rfc-editor.org/rfc/rfc8785 <https://www.rfc-editor.org/rfc/rfc8785>.
- RFC 8032 (EdDSA): https://www.rfc-editor.org/rfc/rfc8032 <https://www.rfc-editor.org/rfc/rfc8032>. RFC 7515 (JSON Web Signature): https://www.rfc-editor.org/rfc/rfc7515 <https://www.rfc-editor.org/rfc/rfc7515>.
- FIPS 180-4 (SHA-256): https://csrc.nist.gov/pubs/fips/180-4/upd1/final <https://csrc.nist.gov/pubs/fips/180-4/upd1/final>.
- W3C Verifiable Credentials Data Model 2.0: https://www.w3.org/TR/vc-data-model-2.0/ <https://www.w3.org/TR/vc-data-model-2.0/>. W3C Decentralized Identifiers: https://www.w3.org/TR/did-core/ <https://www.w3.org/TR/did-core/>.
- W3C Bitstring Status List v1.0, W3C Recommendation of 15 May 2025: https://www.w3.org/TR/2025/REC-vc-bitstring-status-list-20250515/ <https://www.w3.org/TR/2025/REC-vc-bitstring-status-list-20250515/>.
- RFC 9901, Selective Disclosure for JSON Web Tokens (Proposed Standard, November 2025): https://www.rfc-editor.org/rfc/rfc9901 <https://www.rfc-editor.org/rfc/rfc9901>. SD-JWT-based Verifiable Credentials remained an Internet-Draft (draft-ietf-oauth-sd-jwt-vc, revision 13 of 6 November 2025) when last checked: https://datatracker.ietf.org/doc/draft-ietf-oauth-sd-jwt-vc/ <https://datatracker.ietf.org/doc/draft-ietf-oauth-sd-jwt-vc/>.
- 1EdTech Open Badges 3.0 (final, 17 June 2024): https://www.imsglobal.org/spec/ob/v3p0/ <https://www.imsglobal.org/spec/ob/v3p0/>.
- RFC 6962 and RFC 9162 (Certificate Transparency): https://www.rfc-editor.org/rfc/rfc6962 <https://www.rfc-editor.org/rfc/rfc6962>, https://www.rfc-editor.org/rfc/rfc9162 <https://www.rfc-editor.org/rfc/rfc9162>.
- OpenTimestamps: https://opentimestamps.org/ <https://opentimestamps.org/>.
- European Qualifications Framework: https://europass.europa.eu/en/europass-digital-tools/european-qualifications-framework <https://europass.europa.eu/en/europass-digital-tools/european-qualifications-framework>. ESCO: https://esco.ec.europa.eu/ <https://esco.ec.europa.eu/>. Mathematics Subject Classification 2020: https://msc2020.org/ <https://msc2020.org/>.
- Regulation (EU) 2024/1689 (Artificial Intelligence Act), Annex III on high-risk systems in employment: https://eur-lex.europa.eu/eli/reg/2024/1689/oj <https://eur-lex.europa.eu/eli/reg/2024/1689/oj>.

- UAE Federal Decree-Law No. 45 of 2021 on the Protection of Personal Data (citation of the law; consult the official legislation portal for the text).