CITAE: A Passage-Verifiable Retrieval-Augmented Generation Platform for Researchers and Thesis Writers

Richar Andre Vilca-Solorzano, Dina Maribel Yana-Yucra, Fred Torres-Cruz, Milton Vladimir Mamani-Calisaya, Alain Paul Herrera-Urtiaga and Vladimiro Ibañez-Quispe
Faculty of Statistical and Informatics Engineering (FINESI), Universidad Nacional del Altiplano, Puno, Peru
E-mail: 75521963@est.unap.edu.pe, maribelbel201314@gmail.com, ftorres@unap.edu.pe, mmamanic@unap.edu.pe, aherrera@unap.edu.pe, vibanez@unap.edu.pe

Keywords: Academic literature tools, retrieval-augmented generation, BM25 passage retrieval, grounded language models, reference management, open-source scientific software, educational technology

Received: July 18, 2026

Assistants based on large language models accelerate scholarly reading, but they hallucinate, which is unacceptable in academic work where every claim must be traceable to a source. We present CITAE*, an open-source web application that integrates the complete scientific-literature workflow—discovery, reading, verification, organisation and citation—into a single system, and makes AI grounding* visible to the user *rather than hidden inside the model. Built as a React frontend over a layered Node.js/Express backend and a PostgreSQL database, its reading assistant ranks the user's papers and highlights with Okapi BM25 (*$k_1 = 1.5$*,* $b = 0.75$*) and injects the top-ranked, numbered passages into the prompt; the answer's inline* `[n]` *markers are then bound back to the passages they cite and rendered as* trust cards *that expose the literal supporting text and its source. A second, single-document assistant additionally* verifies *every quoted passage against the source and discards any that cannot be matched, so fabricated evidence never reaches the reader. A provider-agnostic client routes every model call through each user's own keys, failing over across Groq, Google Gemini and OpenRouter, so the AI features run at no cost under a bring-your-own-key scheme and degrade gracefully. Around this core, CITAE unifies multi-source discovery over Crossref, Semantic Scholar, OpenAlex and arXiv with claim verification, paper comparison, literature reports, spaced review and citation in seven styles. We report an engineering evaluation on the production code: in-memory BM25 retrieval takes 0.5 ms over 100 and 6.4 ms over 1,000 passages; the deterministic verification guard attains precision, recall and* F_1 *of* 1.000 *on a labelled 50-quotation probe set, rejecting all adversarial "genuine-prefix + fabricated-tail" strings; and an 86-case test suite covers the deterministic core. Offered bilingually with a Spanish-first default, free and open source, CITAE is an equity-oriented alternative to today's fragmented, mostly proprietary and English-first landscape.*

Povzetek: Članek predstavlja CITAE, ki je odprtokodna platforma za znanstveno delo, ki povezuje iskanje literature, preverjanje trditev, organizacijo in citiranje ter omogoča pregledno preverjanje virov odgovorov umetne inteligence. Sistem zmanjšuje tveganje halucinacij z vezavo odgovorov na dejanske odlomke iz člankov in preverjanjem citatov.

1 Introduction

Working with scientific literature is a core competence in higher and distance education, where learners direct much of their own study [12, 13]. Mastering it means working through a cycle of tasks: *discovering* relevant sources, *reading* them critically, *verifying* their claims, *organising* knowledge and *citing* correctly. These abilities form part of what the literature calls information and digital literacy, widely recognised as a determinant of academic success [20, 1, 7].

In practice, however, this cycle is *fragmented* across independent tools: search engines for discovery, reference managers such as Zotero or Mendeley for organisation and citation, PDF readers, and, more recently, AI assistants for summarisation and question answering [24, 33]. Data rarely flow cleanly between them, and three problems are especially acute in low-resource, Spanish-speaking settings [6]: many capable tools are *paid* or gate their best features, which deepens access disparities in the Global South [14, 39]; most AI interfaces are *English-first*, penalising Spanish-speaking users [40]; and, above all, LLM-based assistants *hallucinate*—producing plausible but false statements or references, which is unacceptable where source traceability is non-negotiable [3, 19, 17].

Retrieval-augmented generation (RAG) can mitigate hallucination by grounding answers in retrieved evidence rather than in the model's parametric memory [27, 11]. Yet grounding is usually an *internal* mechanism: the user still receives a fluent answer and must trust that it is supported. The gap we address is therefore twofold. First, no widely available tool combines the *whole* scholarly cycle—

discovery, reading, verification, organisation and citation—with verifiable AI in a single, *free*, Spanish-first (bilingual) and *open-source* environment; current AI research assistants are typically proprietary, English-first and cover only a slice of the cycle. Second, even tools that cite sources rarely make the *exact supporting passage* a first-class, always-visible object that the reader can check at a glance—and audits of citation-bearing generative search engines show that a large fraction of their statements are not fully supported by the sources they cite [28].

This paper presents **CITAE**, an open-source web platform—in production and offered bilingually (Spanish by default)—that closes this gap, and describes it as a working software system. Concretely, we ask three research questions: *RQ1* — can passage-level grounding be made a first-class, user-facing artefact so that a reader can verify each AI statement against its literal source, and can fabricated quotations be prevented deterministically from reaching the reader? *RQ2* — can such an assistant, together with the full scholarly workflow, run at negligible platform cost on modest hardware (in-memory retrieval, bring-your-own-key models), making it viable for free, low-resource deployment? *RQ3* — can the whole cycle be delivered as an open-source, bilingual (Spanish-first) system rather than a proprietary, English-first one? We report not only the design but how it *actually* behaves—the retrieval pipeline, its robustness fallbacks, the provider-failover client, and an engineering evaluation of the deterministic core. The contributions, mapped to these questions, are:

- A **dual-path passage-verifiable grounding architecture**. For every AI answer, the library assistant binds each inline citation marker back to the retrieved passage it cites, while a stricter single-document assistant additionally *verifies* every quoted passage against the source text and discards unverifiable ones—so that a fabricated passage cannot reach the reader. Grounding becomes an auditable artefact *at the interface*—the *trust card*—rather than a hidden internal step.
- A **robustness ladder over an in-memory BM25 retriever** tuned for the small, heterogeneous personal libraries common in real academic use. The index is rebuilt per query with no offline indexing step, and a cascade of fallbacks—empty-library guard, small-library bypass, query expansion and a first-passages fallback—keeps answers grounded and source-attributed for libraries of any size.
- An **engineering evaluation on the production code**: an empirical validation of the deterministic verification guard on a labelled probe set, and CPU-bound micro-benchmarks showing sub-10 ms retrieval even over a thousand passages, together with a capability comparison positioning CITAE against reference managers, academic search engines and current AI research assistants.
- An **integrated, open-source and bilingual platform** that unifies multi-source discovery, semantic highlighting, the RAG reading assistant, claim verification, paper comparison, literature reports, spaced review and citation—built on a layered Node.js/Express/PostgreSQL backend and a React frontend with a provider-agnostic, bring-your-own-key model client—as an equity-oriented alternative for resource-constrained, Spanish-speaking settings.

The remainder of the paper reviews related work (Section 2), describes the architecture (Section 3) and the verifiable retrieval pipeline (Section 4), details the core features (Section 5), reports the engineering evaluation (Section 6), compares CITAE with existing tools (Section 7), discusses the approach (Section 8) and concludes (Section 9).

2 Related work

Academic discovery infrastructure. Open scholarly metadata and indexes—Crossref [15], OpenAlex [35] and the Semantic Scholar open data platform [23]—have made it possible to build tools that query several sources programmatically. CITAE builds on these to provide unified, multi-source discovery.

Reference management. Studies of reference managers such as Zotero and Mendeley report both their value and their accuracy limitations for citation generation [24, 33]. CITAE integrates citation generation with discovery and reading rather than treating it as a separate step.

Retrieval and grounded generation. Classic probabilistic ranking, notably Okapi BM25 [37], remains a strong, cheap and transparent baseline for lexical retrieval; CITAE uses it as the retriever of its reading assistant. Retrieval-augmented generation composes such a retriever with a language model to ground answers in evidence [27, 11], mitigating hallucination [19, 17]; even so, faithfulness and *attribution*—whether each statement is genuinely supported by its cited source—remain distinct, actively measured problems [30, 36, 10]. On this basis, LLMs are increasingly adopted as tools across the literature-review process [38]. CITAE operationalises this grounding at the interface level through trust cards that expose the supporting passage.

LLMs and RAG in education. Recent work discusses the opportunities and risks of LLMs for education [22, 9, 42] and the ethics of generative AI in scholarly writing [29, 5, 31]. A recurring recommendation is to keep a human in the loop and to make the provenance of AI output explicit [31]; CITAE follows this principle by never returning unsupported answers and keeping the supporting passage one click away.

AI research assistants. A new generation of AI-powered research tools has appeared, such as Elicit, Consensus, Scite [32], Perplexity and ResearchRabbit.[1]

[1] `elicit.com`, `consensus.app`, `scite.ai`, `perplexity.ai`, `researchrabbit.ai` (accessed 2026).

These tools have advanced grounded question answering—returning answers with citations, or extracting claims and their supporting statements from papers—and are valuable for discovery and synthesis; comparative evaluations, however, report uneven coverage and citation reliability across them [8]. They differ from CITAE in three respects. First, they are largely *proprietary and English-first*, and most gate their capabilities behind paid tiers. Second, each typically addresses a *slice* of the workflow (search, or question answering, or citation analysis) rather than integrating discovery, reading, verification, organisation and citation in one place. Third, while several surface citations, CITAE makes the *literal supporting passage* a first-class, always-visible artefact—the trust card—tied to the user's own library and paired with an explicit claim-verification view. CITAE thus complements this landscape with an open, bilingual (Spanish-first) and cost-free alternative in which verifiability is part of the interface contract. Table 2 summarises how representative tools position themselves and what each lacks relative to CITAE.

Open research software. Releasing software openly supports auditability and reproducibility [18, 41, 34]; CITAE is distributed as open source for this reason.

3 System architecture

CITAE is an open-source web application organised as a `pnpm` monorepo with two components that communicate through a REST API. The *frontend* is a single-page application built with React, using a token-based design system, light/dark themes and a responsive layout; application state is held in composable hooks rather than a global store, and heavy pages are code-split. The *backend* is a Node.js/Express service structured in strict layers (routes → controllers → services/models) on top of a PostgreSQL relational database. Papers are stored *globally by their DOI*, so each user's library is defined as the union of their favourites, collections, tags and highlights over that shared corpus—an intentional data-model choice that de-duplicates storage and lets one paper's metadata be enriched once and reused by every user. The system is deployed behind an HTTPS reverse proxy (Nginx) as a managed `systemd` service. Figure 1 shows the overall architecture.

Layering and data access. The backend enforces a strict layering in which *all* SQL is confined to the model layer over a `pg` connection pool, and controllers never issue queries directly; the pool adapter also translates positional `?` placeholders to PostgreSQL's `$n` form. Domain logic lives in stateless services with no Express coupling, so the retrieval, ranking, citation and export routines are unit-testable in isolation (Section 6). Twelve route modules expose the REST surface (`/api/auth`, `/papers`, `/citations`, `/highlights`, `/library`, `/claim`, `/rag`, `/public`, `/branding`, `/admin`, and profile/API-key routes), each with per-feature rate limiting.

Security. Security measures include JSON Web Token authentication with server-side session validation and a role read fresh from the database on each request; a signed anti-CSRF state parameter in third-party (Google) sign-in; magic-byte validation of uploaded files; encryption at rest of user-supplied API keys; and an outbound-request guard against server-side request forgery. The guard is not a mere string filter: before any preview or open-access fetch it rejects non-HTTP(S) schemes and non-standard ports, and it *resolves the hostname via DNS* and checks every returned address against private and reserved ranges (RFC 1918, loopback, link-local, carrier-grade NAT and the IPv6 equivalents), so a public name that resolves to an internal address is still blocked.

Documents and citation. Documents are processed with a PDF text extractor (heuristic extraction of title, authors, abstract, DOI, journal and year, with Crossref enrichment when a DOI is found) and, as a fallback for scanned files, optical character recognition for Spanish and English. The citation formatter produces seven styles and the library can be exported in four bulk formats (Section 5).

Quantitative profile. The production code comprises roughly 7,600 lines in the backend services and models and 11,300 lines in the React frontend (≈25 feature components and 7 state hooks), exposing 87 REST endpoints across the twelve route modules. This profile situates CITAE as a medium-sized, single-maintainer system rather than a research prototype, and motivates the emphasis on a testable, deterministic core (Section 6).

4 The verifiable reading assistant and retrieval pipeline

The reading assistant is the core of CITAE, and the place where "verifiable AI" becomes concrete. A user can ask a question about a single document or about their whole library, and the assistant replies in natural language [27]. Crucially, the reply is never a bare paragraph: each statement is tied to a *trust card* that shows the *literal passage* and the source on which it is grounded (Figure 2). Grounding thus becomes an explicit, inspectable artefact rather than an internal step the reader must take on faith. CITAE enforces this through two complementary paths—library-wide RAG and single-document verification—described below and summarised in Figure 5.

4.1 Corpus construction and lexical retrieval

The retrieval corpus is built *per user* from two natural units: each paper contributes a passage formed by its title and abstract, and each highlight contributes a passage formed by the highlighted quote and the user's note. Every passage keeps its provenance (authors, year, journal, DOI and URL). Given a question, text is tokenised with accent

Table 1: Representative tools versus citae (authors' assessment of each tool as of 2026, not exhaustive benchmarking). *passage-level verification* means the *literal* supporting passage is exposed and checked, not merely a citation. *yes*/*partial*/*no* as in table 7. The recurring gaps are the absence of passage-level, user-facing verification, of open-source availability, and of an integrated, bilingual, cost-free workflow—exactly what citae provides.

Tool	Primary role	Passage-level verification	Open source	Free tier	Main gap vs. CITAE
Elicit	QA over papers	No	No	Partial	Answer-level cites; paid gating
Consensus	Evidence QA	No	No	Partial	Aggregates claims; no literal passage
Scite	Citation-context index	Partial	No	Partial	Cites contexts, not user's own library
Perplexity	General web QA	No	No	Partial	Non-scholarly sources; unverified cites
ResearchRabbit	Discovery/graphs	No	No	**Yes**	No AI grounding or verification
Google Scholar	Search UI	No	No	**Yes**	Discovery only; no reading/AI
Semantic Scholar	Search + TLDR	No	Partial	**Yes**	Summaries, no per-passage verification
Zotero	Reference manager	No	**Yes**	**Yes**	No discovery or grounded AI
Mendeley	Reference manager	No	No	Partial	No discovery or grounded AI
CITAE	**Full cycle + verifiable AI**	**Yes**	**Yes**	**Yes**	—

folding and Spanish/English stop-word removal, and passages are ranked with Okapi BM25 ($k_1 = 1.5$, $b = 0.75$) [37] using the inverse-document-frequency form $\mathrm{idf}(t) = \ln\big(1 + (N - n_t + 0.5)/(n_t + 0.5)\big)$, where N is the number of passages and n_t the number containing term t.

We deliberately use lexical BM25 rather than dense vector retrieval [21]. For a personal library—typically tens to a few hundred passages—BM25 needs no embedding model and no vector database, so the tool stays free and self-hostable; its ranking is fully *inspectable and reproducible*, a property that matters when the goal is verifiability; and it is strong precisely when the user queries with terms present in the text, as is common in scholarly reading. The index is rebuilt in memory on each query from the user's current library, so newly added papers and highlights are searchable immediately with no offline indexing step; Section 6 shows this rebuild-and-rank cost stays under 8 ms even for a thousand passages. The trade-off—weaker recall for paraphrased evidence—is addressed by a query-expansion fallback (below) and discussed in Section 8.

4.2 Robustness ladder

Rather than a single retrieve-then-generate call, the assistant applies a small ladder of fallbacks so that a sparse or idiosyncratic library still yields a useful, grounded answer:

1. If the library is *empty*, the assistant returns a guidance message and *never calls the model*.
2. If the library is *small* (at most twelve passages), retrieval is skipped and the whole library is passed as context, since ranking adds little when everything fits.
3. Otherwise BM25 selects the top passages. If fewer than three score above zero, the assistant asks the model to *expand the query* with key terms and their English equivalents (scholarly literature is largely in English) and retrieves again.
4. If retrieval still returns too few passages, the assistant falls back to the first passages of the library so the user receives a best-effort, still source-attributed answer rather than nothing.

This ladder is a deliberate engineering choice: it trades a little precision for resilience on the very small, heterogeneous libraries that are common in practice. Figure 3 shows the decision flow.

4.3 Prompting and marker binding

The selected passages are assembled into the model context, each numbered and labelled by type (paper or highlight) and truncated to a fixed length. The system prompt instructs the model to answer *only* from the numbered passages, to append the source number `[n]` after every statement drawn from a passage, and to state explicitly when the passages are insufficient; a low decoding temperature further discourages free-form generation (Figure 4). Once the answer is produced, its inline `[n]` markers are parsed and each is bound back to the passage it cites; the frontend renders every marker as an expandable trust card showing the literal snippet of that passage with its source metadata. If the model cites no marker, the assistant attaches all retrieved passages as sources rather than presenting an unattributed answer. Grounding is thus enforced at three points—retrieval, prompting, and the post-hoc binding of markers to passages—rather than left to the model alone.

4.4 Single-document verification

When the question concerns a single open document (an abstract or open-access full text), CITAE applies a stricter,

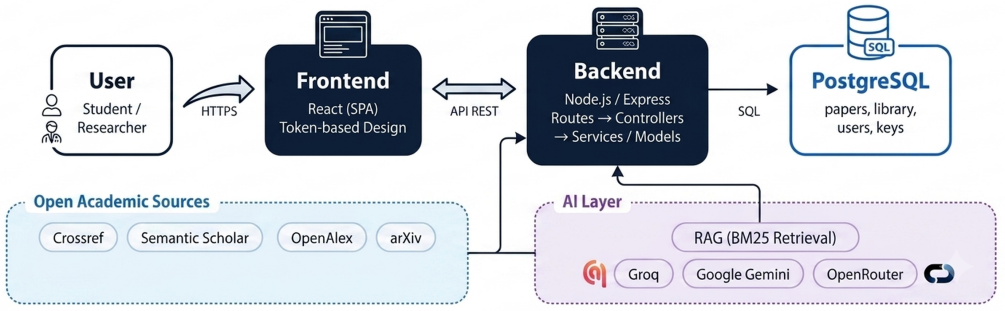

Figure 1: System architecture of citae. A react single-page frontend and a layered node.Js/express backend over PostgreSQL orchestrate the open academic sources (crossref, semantic scholar, OpenAlex, arXiv) and the AI layer, in which an in-memory BM25 retriever grounds a provider-agnostic language-model client (groq, gemini, OpenRouter).

second grounding path. The model is asked to answer only from that document and to append, verbatim, the phrases that support its answer. Before any of those phrases is shown to the reader, the system *verifies* each one against the source text by normalised substring matching and *discards* any phrase that cannot be located in the source. When a phrase is not a full substring, the guard falls back to its longest contiguous leading run that occurs verbatim in the source and accepts it only if that run covers at least 80% of the phrase, so a genuine opening followed by a fabricated tail (a low-coverage match) is rejected rather than shown. Only verified phrases become trust cards; if none survive, the answer is shown with no supporting passage rather than with a fabricated one. This closes the residual gap in which a model, asked to quote, paraphrases or invents a quotation: in CITAE such a phrase is silently dropped, so a fabricated passage never reaches the reader.

4.5 The provider-agnostic model client

A single provider-agnostic client mediates *every* model call in the system; no other module talks to a language-model API directly. All providers are accessed through the OpenAI-compatible chat interface, which lets one code path serve Groq, Google Gemini and OpenRouter. For each request the client assembles a runtime list of providers that puts the *user's own keys first* and the platform keys last, as a safety net; within a provider it schedules keys round-robin and, on a rate-limit or authorisation error (HTTP 429/401/403), places the offending key on a cooldown (honouring any `Retry-After` header) and fails over to the next key or provider. The status of each user key (active, exhausted, invalid) is recorded and surfaced in the interface. Under this bring-your-own-key scheme the assistant runs at no cost to the user and degrades gracefully: if no key is available, AI features simply report themselves as unavailable instead of failing the request.

We make a deliberately *modest* claim about this design. It does not *eliminate* hallucination: the retriever can miss evidence, and the model can still misread a passage it was given. What it does is constrain the model to cite retrieved passages, refuse to answer when the library is empty, verify quoted passages against their source, and place the exact supporting text one glance away, so that any residual error is *immediately detectable* by the reader rather than hidden behind fluent prose [19, 11].

5 Core features

Around the verifiable assistant, CITAE integrates the rest of the literature workflow; every AI-produced output in the system reuses the same grounded, source- attributed machinery.

Unified multi-source discovery. Given a query by DOI, URL, title or an uploaded PDF, CITAE queries four open academic sources *in parallel*—Crossref, Semantic Scholar, OpenAlex and arXiv—unifies and de-duplicates the results, ranks them by relevance, and produces an AI-generated summary of the result set so users can grasp the landscape before opening a single paper (Figure 6). De-duplication groups candidates by normalised DOI, and otherwise by normalised title, merging fields across sources, keeping the highest citation count and recording which sources corroborate each result. Relevance scoring is a transparent, multi-signal function: a title-similarity core (weighted token overlap with inverse-frequency weighting, character-bigram Dice coefficient, word-bigram overlap and an exact-phrase bonus) is combined with additive boosts for multi-source corroboration, citation count (logarithmic), author match, DOI presence and year proximity, and clamped to a bounded range; the per-signal breakdown is retained so a result's ranking can be explained.

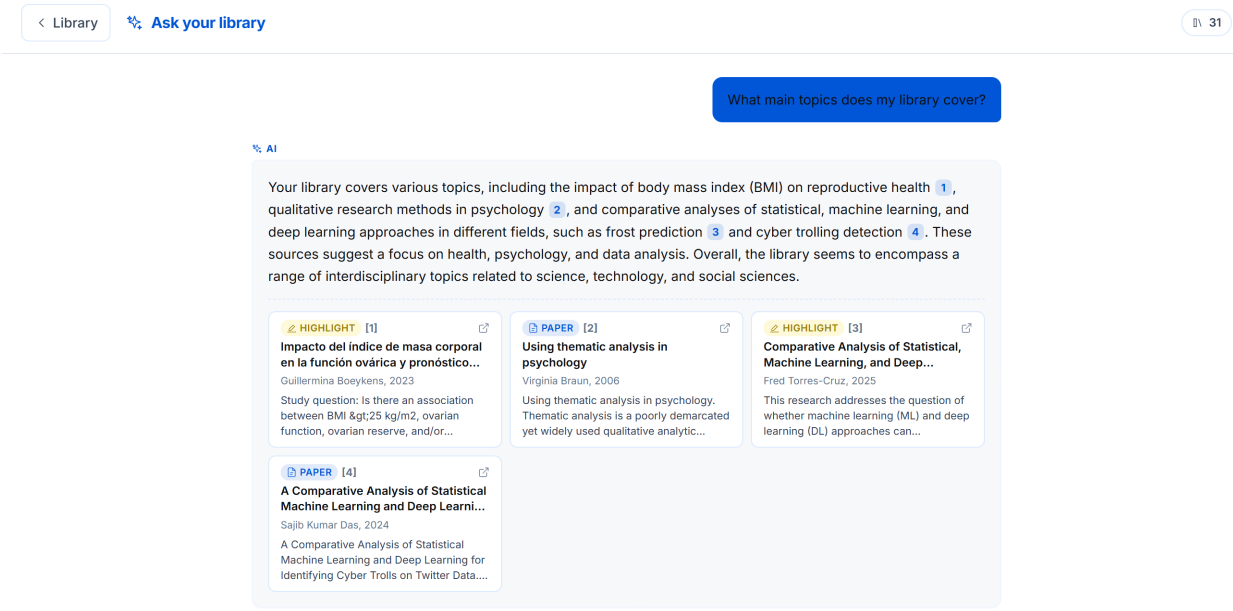

Figure 2: The verifiable reading assistant answering a library-wide question. Each numbered marker in the answer expands into a trust card that exposes the literal supporting passage (verbatim from the source) together with the source and its DOI, so the reader can check every statement against its evidence at a glance. The interface is shown in its english locale; sources retain their original language.

Uploaded PDFs are parsed, with OCR (Spanish and English) as a fallback for scanned documents. For each candidate a preview (cover and open-access PDF, the latter resolved through Unpaywall) is produced behind the anti-SSRF guard. Because the indexes have complementary coverage—Crossref for publisher metadata, OpenAlex and Semantic Scholar for citation graphs and open-access status, and arXiv for preprints—aggregating them increases recall [15, 35, 23]. Natural-language filters (bilingual, regular-expression based) let users constrain results by year range, minimum citation count or document type directly in the query, and an author- discovery view queries OpenAlex for a researcher's profile and works.

Reading and semantic highlighting. While reading an abstract or an open-access full text, the user can highlight passages in five colours that carry academic meaning (for example, objective, method, result, limitation or quotation) and attach notes. Highlights are not throw-away marks: each is stored, linked to its source, and enters the personal library automatically, so that reading itself gradually builds an annotated, searchable corpus that later feeds both the reading assistant and the spaced-review mode.

Verification, synthesis and study tools. Beyond reading a single paper, CITAE supports higher-order tasks that reuse the grounded-AI machinery. A *claim radar* takes a statement, retrieves the top candidate papers from the four sources, and asks the model to classify each as *supporting*, *contradicting*, *mixed* or *neutral* with respect to the claim, returning an aggregate verdict with per-paper evidence. A *comparison* view contrasts several papers along shared dimensions (objective, method, key results, limitations, topic) in a structured table. A *deep-research* mode analyses a collection and assembles a structured literature report—summary, themes, methods, findings, contradictions and gaps—over the papers it contains. For studying, the highlights collected while reading feed a *spaced-review* mode that resurfaces them over time, and any highlight or passage can be exported as a shareable *quote card* image. All of these outputs expose the sources behind them.

Organisation, citation and sharing. A personal library supports favourites, collections and tags (with AI-assisted auto-tagging), full-text search and an activity heatmap; highlighted papers enter the library automatically. Citations are generated instantly in seven styles (APA, MLA, Chicago, Harvard, IEEE, Vancouver and BibTeX), and the whole library can be exported to BibTeX, RIS, CSV and Markdown. Collections and user profiles can be made public, so that curated reading lists can be shared with peers or students.

Design decisions for equity. Three decisions are relevant for a low-resource, Spanish-speaking context. First, the whole interface is *bilingual*, defaulting to Spanish for the target community while also offering English at the click of a button, so the tool is neither English-first nor Spanish-only; the reading assistant and every AI-generated output follow the selected language. The interface is fully localised through per-module message catalogues (one namespace per component) validated so that the Spanish and English key sets match exactly, and the model is instructed, per request, to reply in the selected lan-

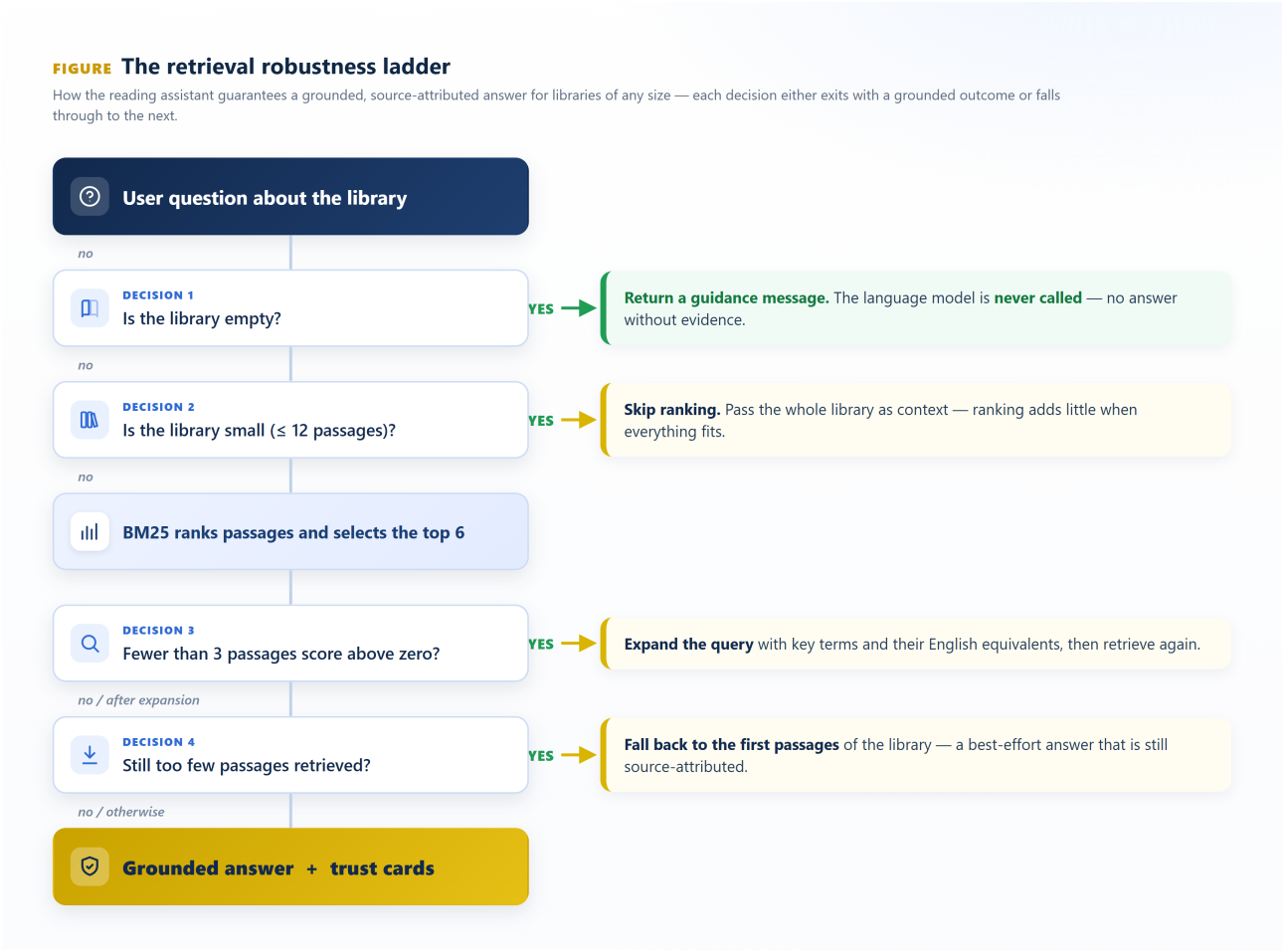

Figure 3: The retrieval robustness ladder. Each decision either exits with a grounded outcome or falls through to the next: an empty library yields a guidance message without ever calling the model, a small library is passed whole, and otherwise BM25 ranking is backed by query expansion and a first-passages fallback, so a source-attributed answer is produced for libraries of any size.

guage. Second, each user can supply their *own API key* for the AI providers (a bring-your-own-key scheme) over services with free tiers, so that using the assistant carries no cost; keys are stored encrypted, and the platform's threat model explicitly covers server-side request forgery (mitigated by an anti-SSRF guard that blocks private and loopback targets) and API-key leakage (keys are encrypted at rest and never returned in responses or logs), while denial-of-service and provider-side abuse are out of scope. Third, CITAE is distributed as *open-source software*, favouring auditability, reuse and sustainability as open educational infrastructure [18, 41].

6 Engineering evaluation

Because the contribution of CITAE is a *system*, we evaluate it as one. Two questions matter for a tool intended to run for free on modest hardware and to be trustworthy for academic use: (i) is the CPU-bound core fast enough to keep the platform's per-request cost negligible, so that the operator's expense stays low and the bring-your-own-key model is sustainable? and (ii) is that core covered by tests dense enough to make its deterministic behaviour reliable? A controlled usability study with students and researchers—whose protocol is already defined—is deliberately out of scope here and stated as future work (Section 8); this paper instead focuses on the engineering properties that are critical for the reliability of grounding and the viability of a free deployment, measured on the shipping code.

Passage-verification guard. The strongest claim of the paper is that a fabricated passage never reaches the reader. Because the single-document verifier (Section 4.4) is deterministic, we test it directly. We built a labelled probe set of 50 candidate quotations against six source texts, in five categories: genuine verbatim quotes (with case, accent and quotation-mark variations), meaning-preserving paraphrases, fabricated statements, trivially short fragments that the guard ignores by design, and an adversarial category that prefixes a genuine opening to a fabricated continua-

```
System: You are the assistant for the user's
academic library. Answer the question using ONLY
the numbered sources below.
SOURCES:
[1] (PAPER) "<title>" -- <authors>, <year>
  <passage text, truncated>
[2] (HIGHLIGHT) "<paper>" -- <color>
  <quote> -- <note>
Rules: reply in the user's language, clearly and
concisely (<= 6 sentences); after every statement
drawn from a source, ALWAYS cite its number in
brackets [n]; if the sources are insufficient,
say so honestly; do not invent anything beyond the
sources.
```

Figure 4: System prompt template used by the library reading assistant (translated and condensed). Retrieved passages are injected as numbered sources; the model is required to cite them and to admit when they are insufficient.

tion. Every probe is derived from its source by construction, so its gold label is unambiguous. Treating "a genuine quote that should be kept" as the positive class, the guard attains precision, recall and F_1 all equal to 1.000 on this set (Table 6): it retained all 20 genuine quotes—so legitimate evidence is robust to the normalisation it applies—and rejected all 30 non-literal passages: every paraphrase, every fabrication, every trivially short fragment, and every adversarial string.

The adversarial category is the one that earlier exposed a weakness. The guard first attempts a full normalised-substring match; only if that fails does it fall back to a leading-run match. In the original design that fallback accepted a candidate whenever its first eight words appeared in the source, so a genuine opening followed by a fabricated tail could slip through (1 of 2 in an earlier 24-probe set). We replaced it with a *coverage-based* fallback: it takes the longest contiguous leading run of the candidate that occurs verbatim in the source and accepts the candidate only when that run covers at least 80% of it. A fabricated tail therefore yields low coverage and is rejected, which brings adversarial false-accepts to 0 of 6 while leaving genuine retention at 20 of 20; the guard's regression suite (86 backend tests) continues to pass. The verifier remains intentionally strict—it accepts only near-verbatim text, the property that makes a trust card trustworthy. This is a unit-level evaluation of the guard, not an end-to-end hallucination study: it substantiates the verification mechanism, not the model's reasoning.

Micro-benchmarks. We benchmarked the CPU-bound services in isolation, without network or database access, using `performance.now()` with a warm-up iteration to let the JIT settle. Table 6 reports throughput and per-operation latency on an Intel Core i5-12500H under Node.js 25, averaged over 10^3–2×10^5 iterations per case. Retrieval was measured separately over synthetic corpora of increasing size, exercising the exact production `retrieve` routine (in-memory index build plus BM25 scoring, top-6): latency grows near-linearly with corpus size but stays around

Table 2: Behaviour of the deterministic passage-verification guard on a labelled probe set of 50 quotations across six source texts. With the coverage-based fallback, genuine quotations are retained despite case/accent normalisation and all non-literal passages—including adversarial genuine-prefix/fabricated-tail strings—are rejected. Precision, recall and F_1 are all 1.000 (positive class: a genuine quote that should be kept).

Probe category	n	Correctly handled
Genuine verbatim (kept)	20	**20** (100%)
Paraphrase (dropped)	10	**10** (100%)
Fabricated (dropped)	10	**10** (100%)
Too-short (dropped)	4	**4** (100%)
Adversarial (prefix+fake, dropped)	6	**6** (100%)

0.05 ms for 10 passages, 0.52 ms for 100, and 6.4 ms for 1,000 (means over repeated runs)—so even an atypically large personal library is ranked in well under a hundredth of a second. Combining retrieval with the post-hoc verification guard, CITAE's total *deterministic* overhead per grounded query (the part the platform controls, with no network or model call) stays at 0.05 ms, 0.51 ms and 6.24 ms for 10, 100 and 1,000 passages respectively. End-to-end, user-perceived latency adds exactly *one* external model call to this overhead; that call—provider- and model-dependent under the bring-your-own-key scheme—dominates the total and is outside CITAE's control (Section 8), so the platform's own contribution is negligible by comparison.

The practical reading of Table 6 is that CITAE adds no meaningful compute cost of its own: ranking a whole result set, formatting a citation, parsing a natural-language filter and exporting a fifty-item library are all sub-millisecond or low-millisecond operations. This is what makes a free, single-server deployment viable, and it is why the bring-your-own-key design shifts essentially all variable cost off the operator and onto the free tiers of the model providers. Because the index is rebuilt per query, memory and time scale linearly with library size; the near-linear growth to 6.4 ms at 1,000 passages implies that even a library an order of magnitude larger than a typical personal one would still rank in tens of milliseconds, well within interactive budgets, before any move to an incremental or on-disk index becomes necessary.

Test coverage. Correctness of the deterministic core is guarded by a test suite of about ninety cases across nine files—the candidate matcher (ranking and de-duplication), the seven citation formatters (backend and frontend), the natural-language query filters, the anti-SSRF guard, the input validators, the bulk exporters and frontend utilities. Coverage is intentionally concentrated on the *pure, re-*

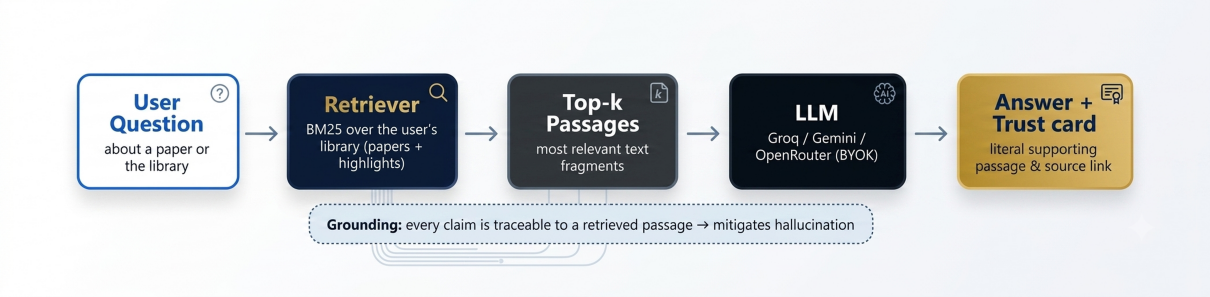

Figure 5: The verifiable reading-assistant pipeline. An in-memory BM25 retriever ranks passages from the user's library; the top-ranked passages are passed to a provider-agnostic LLM client; and the answer is returned with trust cards exposing the supporting passage, verified against the source, so that grounding is verifiable end to end.

Table 3: CPU-bound micro-benchmarks of the citae core (intel core i5-12500h, node.Js 25). Every deterministic operation on the request path completes in microseconds to a few milliseconds, so the platform's own cost per request is negligible next to network and model latency.

Operation	Throughput	Latency
BM25 retrieve, 100 passages	1,931/s	0.52 ms
BM25 retrieve, 1,000 passages	183/s	5.45 ms
Score one candidate	136,710/s	7.3 μs
De-duplicate 200 candidates	5,570/s	180 μs
Rank+classify 200 candidates	437/s	2.29 ms
Format citation (APA)	440,290/s	2.27 μs
Format citation (BibTeX)	538,429/s	1.86 μs
Parse natural-language filter	444,972/s	2.25 μs
Export 50 papers (BibTeX)	9,852/s	102 μs
Export 50 papers (Markdown)	5,183/s	193 μs

producible modules—exactly those on which verifiability depends—while the non-deterministic, provider-dependent AI paths are guarded instead by graceful-degradation contracts (every AI service returns an "unavailable" result, never a crash, when no key or a failing provider is present).

7 Comparison with existing tools

Beyond these measured properties, the higher-level contribution of CITAE is the *integration* of the full literature workflow with verifiable AI in one open and free tool. To characterise it we compare CITAE with three families of widely used tools: reference managers (e.g., Zotero, Mendeley), academic search interfaces (e.g., Google Scholar, Semantic Scholar) and general-purpose AI chatbots. Table 7 summarises the comparison along the stages of the cycle.

Reference managers excel at organisation and citation but do not offer integrated multi-source discovery or a grounded reading assistant. Academic search interfaces cover discovery but not organisation, multi-style citation, or verifiable question answering. General AI chatbots can summarise text but, without retrieval grounding, are prone to hallucination and neither manage a personal library nor produce standard citations. CITAE unifies these capabilities and, distinctively, grounds every AI answer in a literal supporting passage, is free through a bring-your-own-key model, is offered in both Spanish and English, and is released as open source.

The comparison also highlights a design trade-off. Integrating many capabilities in one tool increases its surface and maintenance cost, and specialised tools may still outperform CITAE on any single dimension: dedicated reference managers offer richer citation-style libraries, and dedicated search engines index more sources. The premise behind CITAE is that, for the common case of a student or researcher working through a paper, the reduction in context switching and manual data transfer outweighs the loss of depth in individual features. Grounding the AI in retrieved

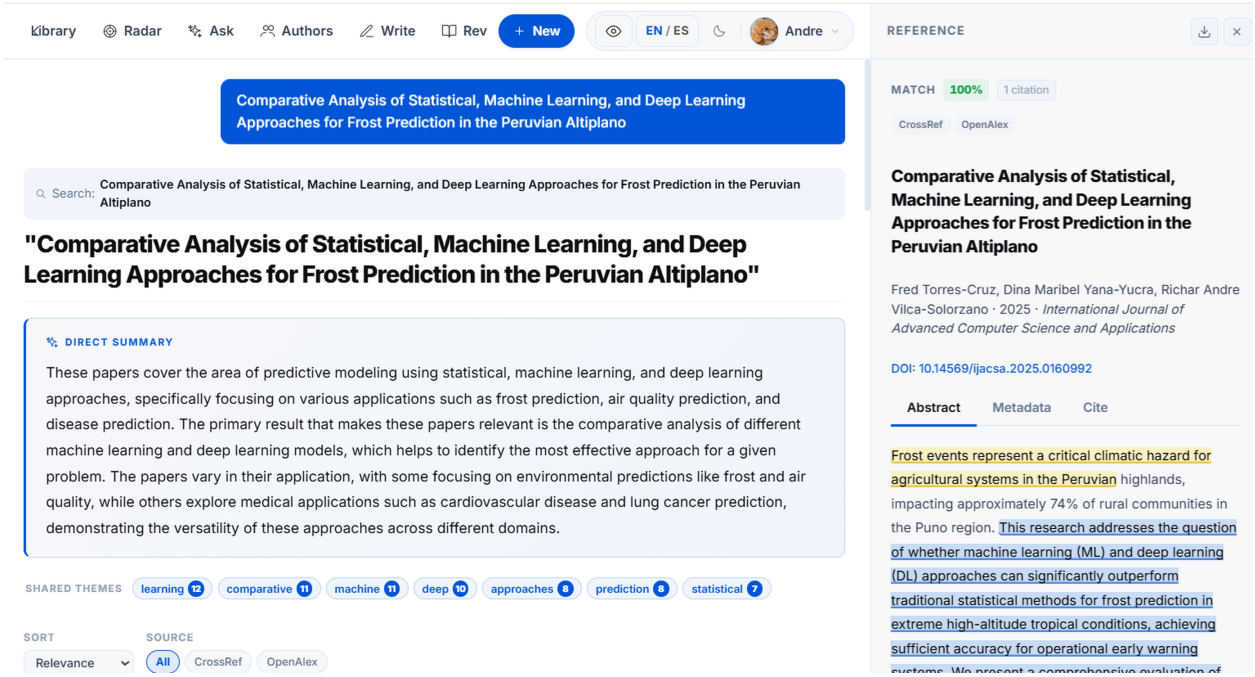

Figure 6: Unified multi-source discovery. Results from crossref, semantic scholar, OpenAlex and arXiv are merged, deduplicated, relevance-scored and summarised, with an AI overview, natural-language filters and a per-result source and match indicator (english locale).

passages, rather than bolting on yet another free-standing chatbot, is what makes this integration trustworthy enough for academic use.

8 Discussion

Results in context. Reading our measurements against the landscape of Table 2 clarifies what is and is not novel. On the engineering axis, CITAE's numbers are not meant to beat a dedicated retrieval engine: in-memory BM25 at 0.5–6.4 ms and a deterministic guard at $F_1 = 1.000$ simply show that the platform's own cost is negligible and that its central safety property holds on the probe set. What no tool in Table 2 offers, and what our results substantiate, is *passage-level, user-facing verification* tied to the user's own library: Elicit and Consensus attach citations or aggregated claims but not the *literal* checked passage; Scite indexes citation contexts rather than a personal corpus; Perplexity may cite non-scholarly pages; reference managers and search UIs perform no grounding at all. We deliberately report engineering metrics rather than retrieval-quality scores such as nDCG, because the contribution is a *verifiable-grounding system*, not a new ranking algorithm; a direct retrieval-quality comparison on a shared benchmark (e.g., BEIR) is a natural but separate study, noted as future work. These differences follow directly from architectural choices—an inspectable lexical retriever, a post-hoc verification guard, and an open, bring-your-own-key design—rather than from scale.

The novelty of CITAE lies not in any single feature—each exists in some separate tool—but in making *verifiable AI* the organising principle of an integrated workflow, and in showing that this can be delivered as a free, bilingual (Spanish-first), open-source system with a small, testable core. By exposing the literal supporting passage, trust cards aim to turn the hallucination risk into an occasion for critical reading, in line with calls to ground language models in their sources [27, 19] and to keep the provenance of AI output explicit in scholarly writing [29, 5, 31]. Because the tool is free, bilingual (Spanish-first) and open source, these safeguards also reach settings that paid, English-first tools often leave out [14].

From an educational standpoint, integrating the literature workflow into a single, guided environment is particularly relevant for higher and distance education, where learners work with high autonomy and benefit from tools that scaffold the whole research cycle rather than isolated steps [12, 22]. By making the evidence behind each AI answer explicit, CITAE can also act as a vehicle for teaching critical reading and the responsible use of AI, turning a potential risk into a learning opportunity [42, 5].

CITAE has been deployed in production and is in active use. Beyond the authors, early users from our academic environment—more than twenty undergraduate researchers, more than ten master's students, and several doctoral candidates—have tried the platform on their own literature: sixteen through their own registered accounts to date, and the remainder in hands-on sessions on the authors' devices. Their informal feedback has been positive,

Table 4: Capability comparison of citae with reference managers, academic search interfaces and general AI chatbots across the literature workflow. Cells reflect the authors' assessment of the tool *families* (not any single product) as of 2026: *yes* = a first-class feature, *partial* = present in some members or in a limited/paid form, *no* = generally absent.

Capability	Ref. managers	Search UIs	AI chatbots	CITAE
Unified multi-source discovery	No	Partial	No	**Yes**
Citation in multiple styles	**Yes**	No	No	**Yes**
Reading and semantic highlighting	Partial	No	No	**Yes**
Grounded (verifiable) AI assistant	No	No	Partial	**Yes**
Claim verification against sources	No	No	No	**Yes**
Personal library and organisation	**Yes**	No	No	**Yes**
Open source	Partial	No	No	**Yes**
Free of charge	Partial	**Yes**	Partial	**Yes**
Bilingual (Spanish + English) UI	Partial	Partial	Partial	**Yes**

and they singled out the search experience and the interface as particularly useful. This offers early, *anecdotal* evidence of feasibility and perceived value at a realistic scale; it is not a substitute for a controlled study, and systematic adoption metrics and a controlled usability evaluation with these same user groups remain planned future work (Section 8). The bring-your-own-key model, in particular, decouples the cost of the AI features from the operator—a practical enabler for sustained free access in educational settings—at the cost of a small configuration step for the user; the interface therefore guides key creation and reports the state of each key.

This work has clear limitations, which we state proactively together with how we intend to address them. (i) The evaluation is an engineering and capability-based assessment rather than a controlled user study; a study with students and researchers—using the System Usability Scale [4, 2, 25, 26] and a think-aloud protocol [16]—is planned and its protocol is already defined. (ii) Lexical BM25 retrieval can miss paraphrased evidence, especially across the Spanish/English boundary; the query-expansion fallback mitigates but does not remove this. As a roadmap item we plan a hybrid lexical–semantic retriever that preserves the zero-cost, self-hostable property by running a small quantised embedding model *on the client* (for example an ONNX or WebAssembly sentence-transformer) or on the existing single server, fusing dense signals with BM25 without adding a paid vector database or a GPU. (iii) A trust card guarantees that a passage was retrieved (and, in the single-document path, that a quotation is genuine), not that the model interpreted it correctly. The single adversarial false-accept observed in an earlier 24-probe set—a genuine opening followed by a fabricated tail—is already addressed by the coverage-based fallback of Section 6, which brings adversarial false-accepts to zero on the expanded set; as a deeper safeguard we further plan an automatic answer–passage entailment check to flag statements that are only weakly supported by their cited passage. (iv) The tool depends on external model providers and on the coverage of the open indexes: in practice, answer quality and latency vary with the chosen model, availability is subject to provider rate limits, and recall is bounded by what Crossref, OpenAlex, Semantic Scholar and arXiv index. The provider-failover client and the multi-source design mitigate, but do not eliminate, these dependencies. None of these limitations affects the central claim: that passage-level, user-facing verification is both feasible and valuable in an integrated, open scholarly tool.

9 Conclusion

We presented CITAE, an open-source, bilingual (Spanish-first) web platform that integrates the scientific-literature workflow around a verifiable reading assistant, and described it as a working system—its layered architecture, its BM25 retrieval pipeline with a robustness ladder and two grounding paths, its provider-failover model client, and an engineering evaluation of its deterministic core. Its central contribution, the passage-verifiable *trust card*, turns AI grounding into a first-class, user-facing artefact: every answer, summary, comparison or verification exposes the literal passage and source that support it, so that readers can *check*—rather than merely trust—what the AI asserts. Combined with full-workflow integration and an equity-oriented design (free through a bring-your-own-key model, and open source), this makes verifiability a property of the interface rather than a hidden internal step, and offers an alternative to today's fragmented, mostly proprietary and English-first tools. Future work includes a controlled user study whose protocol is already defined, a hybrid lexical–semantic retriever, and an answer–passage entail-

ment check, together with several research directions that the platform naturally enables: uncertainty-aware retrieval-augmented generation, reliability assessment and confidence calibration of grounded answers, robustness of the pipeline under noisy or incomplete scholarly metadata, and risk-aware evaluation of AI assistants for scientific writing; integration with learning management systems and a browser extension are also planned. CITAE runs in production and is released as open source to support transparency and reproducibility.

Code and data availability

CITAE is released as open-source software under the GNU Affero General Public License v3.0. The source code is available at https://github.com/Andre031222/CITAE-Ginit, and the exact version described here is archived on Zenodo with a citable DOI: https://doi.org/10.5281/zenodo.21423376 (concept DOI https://doi.org/10.5281/zenodo.21423375 always resolves to the latest version). The production instance runs at https://citae.ginit.dev. The measurements in Section 6 are fully reproducible from the repository: the micro-benchmarks and BM25 retrieval-latency figures from `backend/benchmark/` (`bench.js`, `ragRetrieval.bench.js` and `endToEnd.bench.js`), and the passage-verification precision/recall/F_1 from `backend/benchmark/verifyGuard.eval.js`, which exercises the production verifier over the labelled 50-probe set; each script is run with a single command (`node backend/benchmark/<script>`) and prints the figures reported here. Depositing the software in a long-term, DOI-issuing repository supports its reproducibility and citation [18, 41]. The repository provides a `README`, a contribution guide and an AGPL-3.0 licence to support open, sustainable maintenance.

Declaration on the use of generative AI

The authors used a generative AI assistant to help draft and edit parts of the text; all content was reviewed and verified by the authors, who take full responsibility for the manuscript.

Competing interests

The authors declare that they have no competing interests.

References

[1] Audrin, C. and Audrin, B. (2022) Key factors in digital literacy in learning and education: a systematic literature review using text mining, *Education and Information Technologies*, 27(6), pp. 7395–7419. https://doi.org/10.1007/s10639-021-10832-5

[2] Bangor, A., Kortum, P. T. and Miller, J. T. (2008) An empirical evaluation of the System Usability Scale, *International Journal of Human–Computer Interaction*, 24(6), pp. 574–594. https://doi.org/10.1080/10447310802205776

[3] Bender, E. M., Gebru, T., McMillan-Major, A. and Shmitchell, S. (2021) On the dangers of stochastic parrots: can language models be too big?, *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, ACM, pp. 610–623. https://doi.org/10.1145/3442188.3445922

[4] Brooke, J. (1996) SUS: a 'quick and dirty' usability scale, in *Usability Evaluation in Industry*, Taylor & Francis, pp. 189–194. https://doi.org/10.1201/9781498710411-35

[5] Cotton, D. R. E., Cotton, P. A. and Shipway, J. R. (2023) Chatting and cheating: ensuring academic integrity in the era of ChatGPT, *Innovations in Education and Teaching International*, 61(2), pp. 228–239. https://doi.org/10.1080/14703297.2023.2190148

[6] de la Torre, A. and Baldeon-Calisto, M. (2024) Generative artificial intelligence in Latin American higher education: a systematic literature review, *2024 12th International Symposium on Digital Forensics and Security (ISDFS)*, IEEE, pp. 1–7. https://doi.org/10.1109/ISDFS60797.2024.10527283

[7] Farias-Gaytan, S., Aguaded, I. and Ramirez-Montoya, M.-S. (2023) Digital transformation and digital literacy in the context of complexity within higher education institutions: a systematic literature review, *Humanities and Social Sciences Communications*, 10(1), article 386. https://doi.org/10.1057/s41599-023-01875-9

[8] Featherstone, R., Walter, M., MacDougall, D., Morenz, E., Bailey, S., Butcher, R., Ford, C., Loshak, H. and Kaunelis, D. (2025) Artificial intelligence search tools for evidence synthesis: comparative analysis and implementation recommendations, *Cochrane Evidence Synthesis and Methods*, 3(5), e70045. https://doi.org/10.1002/cesm.70045

[9] Flores-Vivar, J.-M. and Garcia-Penalvo, F.-J. (2023) Reflections on the ethics, potential and challenges of artificial intelligence in the framework of quality education (SDG4), *Comunicar*, 31(74), pp. 37–47. https://doi.org/10.3916/C74-2023-03

[10] Gao, L., Dai, Z., Pasupat, P., Chen, A., Chaganty, A. T., Fan, Y., Zhao, V., Lao, N., Lee, H., Juan, D.-C. and Guu, K. (2023) RARR: researching and revising what language models say, using language models, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ACL, pp. 16477–16508. https://doi.org/10.18653/v1/2023.acl-long.910

[11] Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Wang, M. and Wang, H. (2023) Retrieval-augmented generation for large language models: a survey, *arXiv preprint* arXiv:2312.10997. https://doi.org/10.48550/arXiv.2312.10997

[12] Garcia Aretio, L. (2019) The need for a digital education in a digital world, *RIED. Revista Iberoamericana de Educacion a Distancia*, 22(2), pp. 9–22. https://doi.org/10.5944/ried.22.2.23911

[13] Garcia Aretio, L. (2021) COVID-19 and digital distance education: pre-confinement, confinement and post-confinement, *RIED. Revista Iberoamericana de Educacion a Distancia*, 24(1), pp. 9–32. https://doi.org/10.5944/ried.24.1.28080

[14] Garcia-Martin, J. and Garcia-Sanchez, J.-N. (2022) The digital divide of know-how and use of digital technologies in higher education: the case of a college in Latin America in the COVID-19 era, *International Journal of Environmental Research and Public Health*, 19(6), article 3358. https://doi.org/10.3390/ijerph19063358

[15] Hendricks, G., Tkaczyk, D., Lin, J. and Feeney, P. (2020) Crossref: the sustainable source of community-owned scholarly metadata, *Quantitative Science Studies*, 1(1), pp. 414–427. https://doi.org/10.1162/qss_a_00022

[16] Hertzum, M. (2024) Concurrent or retrospective thinking aloud in usability tests: a meta-analytic review, *ACM Transactions on Computer-Human Interaction*, 31(3), pp. 1–29. https://doi.org/10.1145/3665327

[17] Huang, L., Yu, W., Ma, W., Zhong, W., Feng, Z., Wang, H., Chen, Q., Peng, W., Feng, X., Qin, B. and Liu, T. (2025) A survey on hallucination in large language models: principles, taxonomy, challenges, and open questions, *ACM Transactions on Information Systems*, 43(2), pp. 1–55. https://doi.org/10.1145/3703155

[18] Ince, D. C., Hatton, L. and Graham-Cumming, J. (2012) The case for open computer programs, *Nature*, 482(7386), pp. 485–488. https://doi.org/10.1038/nature10836

[19] Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y. J., Madotto, A. and Fung, P. (2023) Survey of hallucination in natural language generation, *ACM Computing Surveys*, 55(12), pp. 1–38. https://doi.org/10.1145/3571730

[20] Jones, W. L. and Mastrorilli, T. (2022) Assessing the impact of an information literacy course on students' academic achievement: a mixed-methods study, *Evidence Based Library and Information Practice*, 17(2), pp. 61–87. https://doi.org/10.18438/eblip30090

[21] Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D. and Yih, W. (2020) Dense passage retrieval for open-domain question answering, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, ACL, pp. 6769–6781. https://doi.org/10.18653/v1/2020.emnlp-main.550

[22] Kasneci, E. et al. (2023) ChatGPT for good? On opportunities and challenges of large language models for education, *Learning and Individual Differences*, 103, article 102274. https://doi.org/10.1016/j.lindif.2023.102274

[23] Kinney, R. et al. (2023) The Semantic Scholar open data platform, *arXiv preprint* arXiv:2301.10140. https://doi.org/10.48550/arXiv.2301.10140

[24] Kratochvil, J. (2017) Comparison of the accuracy of bibliographical references generated for medical citation styles by EndNote, Mendeley, RefWorks and Zotero, *The Journal of Academic Librarianship*, 43(1), pp. 57–66. https://doi.org/10.1016/j.acalib.2016.09.001

[25] Lewis, J. R. and Sauro, J. (2009) The factor structure of the System Usability Scale, in *Human Centered Design*, LNCS 5619, Springer, pp. 94–103. https://doi.org/10.1007/978-3-642-02806-9_12

[26] Lewis, J. R. (2018) The System Usability Scale: past, present, and future, *International Journal of Human–Computer Interaction*, 34(7), pp. 577–590. https://doi.org/10.1080/10447318.2018.1455307

[27] Lewis, P. et al. (2020) Retrieval-augmented generation for knowledge-intensive NLP tasks, *Advances in Neural Information Processing Systems*, 33, pp. 9459–9474. https://doi.org/10.48550/arXiv.2005.11401

[28] Liu, N. F., Zhang, T. and Liang, P. (2023) Evaluating verifiability in generative search engines, *Findings of the Association for Computational Linguistics: EMNLP 2023*, ACL, pp. 7001–7025. https://doi.org/10.18653/v1/2023.findings-emnlp.467

[29] Lund, B. D., Wang, T., Mannuru, N. R., Nie, B., Shimray, S. and Wang, Z. (2023) ChatGPT and a new academic reality: artificial intelligence-written research papers and the ethics of the large language models in scholarly publishing, *Journal of the Association for Information Science and Technology*, 74(5), pp. 570–581. https://doi.org/10.1002/asi.24750

[30] Maynez, J., Narayan, S., Bohnet, B. and McDonald, R. (2020) On faithfulness and factuality in abstractive summarization, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, ACL, pp. 1906–1919. https://doi.org/10.18653/v1/2020.acl-main.173

[31] Nature (2023) Tools such as ChatGPT threaten transparent science; here are our ground rules for their use (editorial), *Nature*, 613(7945), p. 612. https://doi.org/10.1038/d41586-023-00191-1

[32] Nicholson, J. M., Mordaunt, M., Lopez, P., Uppala, A., Rosati, D., Rodrigues, N. P., Grabitz, P. and Rife, S. C. (2021) scite: a smart citation index that displays the context of citations and classifies their intent using deep learning, *Quantitative Science Studies*, 2(3), pp. 882–898. https://doi.org/10.1162/qss_a_00146

[33] Nitsos, I., Malliari, A. and Chamouroudi, R. (2022) Use of reference management software among postgraduate students in Greece, *Journal of Librarianship and Information Science*, 54(1), pp. 95–107. https://doi.org/10.1177/0961000621996413

[34] Nosek, B. A. et al. (2022) Replicability, robustness, and reproducibility in psychological science, *Annual Review of Psychology*, 73, pp. 719–748. https://doi.org/10.1146/annurev-psych-020821-114157

[35] Priem, J., Piwowar, H. and Orr, R. (2022) OpenAlex: a fully-open index of scholarly works, authors, venues, institutions, and concepts, *arXiv preprint* arXiv:2205.01833. https://doi.org/10.48550/arXiv.2205.01833

[36] Rashkin, H., Nikolaev, V., Lamm, M., Aroyo, L., Collins, M., Das, D., Petrov, S., Tomar, G. S., Turc, I. and Reitter, D. (2023) Measuring attribution in natural language generation models, *Computational Linguistics*, 49(4), pp. 777–840. https://doi.org/10.1162/coli_a_00486

[37] Robertson, S. and Zaragoza, H. (2009) The probabilistic relevance framework: BM25 and beyond, *Foundations and Trends in Information Retrieval*, 3(4), pp. 333–389. https://doi.org/10.1561/1500000019

[38] Scherbakov, D., Hubig, N., Jansari, V., Bakumenko, A. and Lenert, L. A. (2025) The emergence of large language models as tools in literature reviews: a large language model-assisted systematic review, *Journal of the American Medical Informatics Association*, 32(6), pp. 1071–1086. https://doi.org/10.1093/jamia/ocaf063

[39] Valdivieso, T. and González, O. (2025) Generative AI tools in Salvadoran higher education: balancing equity, ethics, and knowledge management in the Global South, *Education Sciences*, 15(2), 214. https://doi.org/10.3390/educsci15020214

[40] Wang, C. (2024) Exploring students' generative AI-assisted writing processes: perceptions and experiences from native and non-native English speakers, *Technology, Knowledge and Learning*, 30(3), pp. 1825–1846. https://doi.org/10.1007/s10758-024-09744-3

[41] Wilson, G., Bryan, J., Cranston, K., Kitzes, J., Nederbragt, L. and Teal, T. K. (2017) Good enough practices in scientific computing, *PLOS Computational Biology*, 13(6), e1005510. https://doi.org/10.1371/journal.pcbi.1005510

[42] Yan, L., Sha, L., Zhao, L., Li, Y., Martinez-Maldonado, R., Chen, G., Li, X., Jin, Y. and Gasevic, D. (2024) Practical and ethical challenges of large language models in education: a systematic scoping review, *British Journal of Educational Technology*, 55(1), pp. 90–112. https://doi.org/10.1111/bjet.13370