Bibby AI: An Editor-Native Agentic Platform for Academic Research, Writing, and Publishing

Nilesh Jain*

July 8, 2026

Abstract

Academic output is produced across a fragmented toolchain: literature discovery in one application, reference management in another, writing in a LaTeX editor, formatting against venue templates by hand, and submission through yet another portal. Each boundary between tools forces a context switch, a format conversion, or a manual copy–paste step, and the cumulative cost dominates the time researchers spend on activities that are not research. We present Bibby AI, an editor-native platform that collapses this toolchain into a single Research–Write–Publish pipeline built around a cloud LaTeX editor. Unlike assistants that attach to an existing editor through a browser extension, Bibby AI owns the full document state, compilation pipeline, and revision history, which allows its agents to perform retrieval-grounded citation insertion, structural edits, and template-compliant reformatting as first-class, verifiable operations rather than text suggestions. The platform integrates (i) ingestion pipelines that convert PDF, DOCX, and handwritten mathematics into clean LaTeX; (ii) a retrieval layer over scholarly metadata enriched with patent-to-paper citation signals derived from USPTO PatentsView and the Marx–Fuegi citation corpus, surfacing the translational impact of candidate references; and (iii) task-scoped agents for literature triage, drafting, revision, and venue formatting that operate directly on the document's abstract syntax representation. Bibby AI is deployed in production and serves 5,000+ active researchers across 50+ subscribing universities. We describe the architecture, the design decisions that editor-nativeness makes possible, and the workflow-level time-savings framework we use to evaluate the platform against fragmented baselines.

1 Introduction

The median research paper is written against a stack of disconnected tools: a search engine or scholarly index for discovery, a reference manager for bibliography curation, a LaTeX editor for composition, venue style files wrestled into compliance by hand, and a submission system at the end. Large language models (LLMs) have been bolted onto individual stages of this pipeline—grammar polishing, chat-based question answering, citation lookup—but the assistants remain *external* to the environment where the document actually lives. Prior work has shown that this separation prevents deep interaction with document state, structure, and revision history, and has attempted to bridge it with browser extensions that synchronize bidirectionally with editors such as Overleaf [1].

We argue for the stronger position: the assistant should not be bridged *into* the editor; the editor, the compiler, the reference graph, and the agents should be a single system. Bibby AI (https://trybibby.com) is built on this premise as a standalone, full replacement for cloud LaTeX editors such as Overleaf—not an extension, plugin, or overlay on any host platform. Its core is a

*Founder, Bibby AI (https://trybibby.com). Contact: nilesh@trybibby.com

cloud LaTeX editor with server-side compilation, and every assistant capability is implemented as an operation on the platform's own document model rather than as text injected into a third-party interface. This eliminates the hardest engineering problems that plague plugin architectures—editor synchronization, patch conflicts, and state security across origin boundaries [1]—by construction, and it unlocks capabilities that are not expressible from outside the editor at all: compilation-verified edits, bibliography-aware refactoring, and one-click retargeting of a manuscript to a different venue template.

Contributions. This paper makes four contributions:

1. **An editor-native architecture** for agentic academic writing in which agents operate on the platform's document and compilation state directly, with every structural edit validated by the compiler before it is surfaced to the user (Section 3).

2. **Ingestion pipelines** that convert the formats researchers actually start from—PDF, DOCX, and handwritten mathematics—into clean, compilable LaTeX, removing the largest single onboarding cost of LaTeX-based workflows (Section 3.2).

3. **A translational-impact retrieval layer** that enriches standard scholarly metadata with patent-to-paper citation signals from USPTO PatentsView [2] and the Marx–Fuegi front-page citation corpus [3], letting authors see which candidate references have demonstrated downstream technological impact—a signal no incumbent writing platform surfaces (Section 3.3).

4. **A deployment report and evaluation framework**: Bibby AI serves 5,000+ active researchers and 50+ subscribing universities in production, and we define a workflow-level accounting of researcher time savings that we use for ongoing measurement (Sections 5 and 6).

2 Related Work

Cloud LaTeX editors. Overleaf is the incumbent collaborative LaTeX platform and has added an AI Assist add-on offering writing suggestions and TeX error explanations. These are assist-level features: they annotate the author's work but do not execute multi-step workflows, ingest external formats into compiled projects, or ground citation choice in retrieval. Bibby AI competes at the platform level—editor, compilation, collaboration—while differentiating on agentic workflow execution, ingestion pipelines, and impact-aware retrieval (Table 1).

In-editor writing assistants. PaperDebugger [1] is the closest system in spirit: a Chrome extension that attaches a multi-agent assistant to Overleaf, with a Kubernetes-orchestrated backend and an MCP toolchain for literature search and document scoring. Its authors identify bidirectional editor synchronization, fine-grained patching, and secure state management as the central technical obstacles of the plugin approach. Bibby AI sidesteps this class of problem: because the editor is ours, there is no synchronization boundary, no reverse-engineered document state, and no dependency on a host platform's goodwill or API stability. XtraGPT [4] studies controllable, context-aware paper revision as a model-level problem; Bibby AI consumes such revision capabilities as one agent among several, grounded in full-document context that the platform natively holds.

Scholarly retrieval infrastructure. Open scholarly indices such as Semantic Scholar [5] and OpenAlex [6] provide the citation graph and metadata backbone on which modern discovery tools

Table 1: Capability comparison. "Extension-based" denotes plugin assistants attached to a host editor (e.g., PaperDebugger on Overleaf [1]). ✓ = native; ∼ = partial or add-on; ✗ = absent.

Capability	Overleaf	Extension-based	Bibby AI
Cloud LaTeX editing & compilation	✓	host-dependent	✓
Writing suggestions / error help	∼ (add-on)	✓	✓
Compile-verified agentic edits	✗	✗	✓
Multi-step workflow agents	✗	✓	✓
PDF/DOCX/handwriting → project ingestion	✗	✗	✓
In-editor retrieval → BibTeX insertion	✗	∼	✓
Patent-to-paper impact signal	✗	✗	✓
One-click venue retargeting	✗	✗	✓
No host-platform dependency	✓	✗	✓

are built. Bibby AI builds its retrieval layer on these corpora and differentiates by joining them against patent-side evidence: PatentsView's disambiguated USPTO data [2] and the Marx–Fuegi corpus of front-page patent citations to science [3], which established at scale that patent-to-paper citations are a meaningful, measurable signal of science's technological reliance.

Document conversion. Format conversion into LaTeX is typically handled by standalone utilities (e.g., Pandoc for DOCX, OCR-based math recognition systems for handwriting). These tools are disconnected from the destination editor, so their output errors surface only at first compile, in a different application. Bibby AI integrates conversion into the platform so that output is compiled, validated, and opened as a live project in one step.

3 System Architecture

Bibby AI is organized as four layers sharing one source of truth: the project store, which holds the LaTeX source tree, compiled artifacts, bibliography, and revision history for every project.

3.1 Editor and Compilation Core

The editor is a browser-based LaTeX environment backed by server-side `pdflatex` compilation in isolated containers, with per-compile resource limits (CPU, memory, and file-descriptor caps) tuned from production incident experience.

Compilation is the platform's universal validator: any agent-proposed structural edit is applied to a shadow copy of the project, compiled, and rejected or repaired before the diff is offered to the author (Fig. 1). This converts the LLM failure mode of "plausible but broken LaTeX" from a user-facing error into an internal retry loop.

3.2 Ingestion Pipelines

Researchers rarely start from a blank `main.tex`. Bibby AI provides three ingestion paths, each terminating in a compiled, editable project:

- **PDF → LaTeX**: layout-aware reconstruction of sectioning, mathematics, tables, and references from published or preprint PDFs, used for revision, extension, and template migration of existing work.

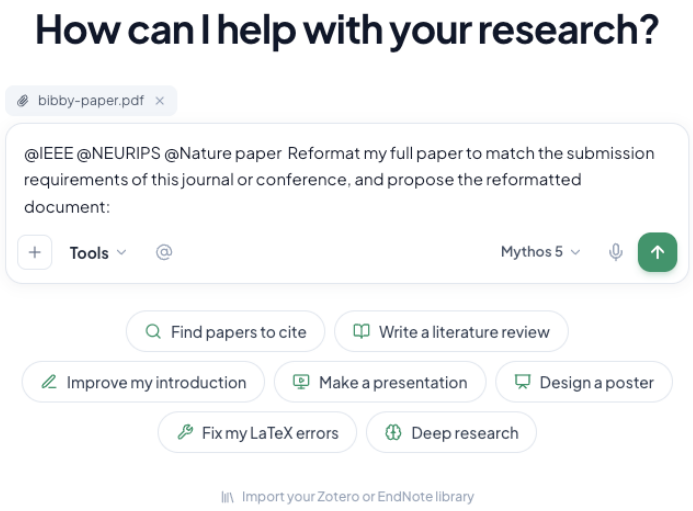

Figure 1: The Bibby AI environment. Left: source editor with structural navigation and rich-text mode toggle. Right: live compiled preview with export pipeline. The Bibby AI panel (top right) hosts the agent layer of Section 3.4; in-document guidance (e.g., citation-search tips) is rendered inline in the compiled output during onboarding projects.

- **DOCX → LaTeX**: structure-preserving conversion for collaborators arriving from Word-based workflows, mapping styles to sectioning commands and equations to `amsmath`.
- **Handwriting → LaTeX**: recognition of handwritten mathematical notation into compilable expressions, targeting the whiteboard-to-manuscript gap.

Because conversion output lands inside the platform, every artifact is compile-checked immediately, and residual conversion errors are annotated in-editor rather than discovered downstream.

3.3 Retrieval and Citation Layer

The retrieval layer answers two questions: *what should I read or cite*, and *why this reference over that one*. For the first, Bibby AI performs semantic and metadata search over open scholarly indices [5, 6] and inserts selected results directly as BibTeX entries with correct keys, venue fields, and in-text `\cite` placement—no export/import round trip through a reference manager.

Table 2: Tool-chain compression for common researcher workflows. Baseline counts assume the modal stack observed in user onboarding interviews (search engine + reference manager + desktop or cloud LaTeX editor + converter utilities).

Workflow	Baseline tools	Bibby AI tools
Literature search → cited draft paragraph	3–4	1
Word-collaborator manuscript → LaTeX project	2–3	1
Venue rejection → reformatted resubmission	2–3	1
Handwritten derivation → compiled section	2–3	1

For the second, Bibby AI computes a *translational impact* signal per paper by joining the scholarly record against patent-side data: PatentsView's disambiguated USPTO corpus [2] and the Marx–Fuegi dataset of front-page patent citations to scientific articles [3]. A paper cited by granted patents carries evidence of downstream technological use that pure academic citation counts do not capture; prior bibliometric work established this signal's validity at corpus scale [3]. Surfacing it at the moment of citation choice is, to our knowledge, unique among writing platforms, and is particularly relevant for authors writing grant applications, impact statements, and applied-research papers.

3.4 Agent Layer

Agents in Bibby AI are task-scoped and operate on the document model, not on selected text alone. Following the taxonomy that has emerged in editor-assistant systems [1], they fall into two execution classes:

- **Single-shot agents** for low-latency operations: sentence- and paragraph-level revision, notation consistency checks, and caption or abstract drafting.
- **Workflow agents** that coordinate retrieval, generation, and validation: literature triage into a structured related-work map; full-document review against a venue's expectations; and template retargeting, which rewrites a manuscript's preamble, sectioning, and bibliography style for a different venue and verifies the result by compilation.

All agent output is delivered as reviewable diffs against the project; nothing is applied silently.

4 End-to-End Workflows

Table 2 contrasts the fragmented baseline against the Bibby AI pipeline for four recurring researcher tasks. The structural claim is tool-count and context-switch reduction; the corresponding time measurements are drawn from platform telemetry and are the subject of our ongoing evaluation (Section 6).

As one concrete trace: a user searching "contrastive learning for tabular data" receives ranked results annotated with academic citation counts *and* patent-citation counts; selecting three papers inserts BibTeX entries and drafts a related-work paragraph with correct `\cite` keys; the compile validator confirms the project builds; total interaction happens in one interface with zero copy–paste operations.

Table 3: Time-cost model instantiation (minutes per workflow instance). Baseline decomposition follows Eq. (1) over the modal fragmented stack; frequencies f_w are per active-researcher month.

Workflow w	T_{base}	T_{bibby}	$S(w)$	f_w
Search → cited paragraph	25	6	19	8.0
DOCX manuscript → LaTeX project	90	8	82	1.0
Venue retargeting (reformat)	180	20	160	0.5
Handwritten math → section	30	4	26	2.0
Compile-error debugging session	20	5	15	6.0

5 Deployment and Adoption

Bibby AI runs in production on containerized infrastructure (Dockerized services behind nginx on dedicated servers) with server-side compilation isolation as described in Section 3.1. As of mid-2026 the platform serves **5,000+ active researchers** and **50+ subscribing universities**, with sustained month-over-month growth in both. Institutional adoption is a meaningful signal for this product category: universities subscribe on behalf of research groups, implying evaluation against incumbent site licenses rather than individual impulse adoption.

6 Evaluation Framework

We evaluate Bibby AI at the level of *workflows*, not model outputs, because the platform's thesis is toolchain compression. For each workflow w we define researcher time cost as

$$T(w) = \sum_{i=1}^{n_w} \left(t_i^{\text{task}} + t_i^{\text{switch}} + t_i^{\text{repair}} \right), \tag{1}$$

where n_w is the number of tool stages, t^{switch} captures context-switch and transfer overhead between stages (export, upload, copy–paste, re-orientation), and t^{repair} captures downstream error correction attributable to stage boundaries (e.g., conversion artifacts discovered at first compile). Editor-nativeness attacks the second and third terms directly: with $n_w = 1$, switch cost is zero by construction, and compile-validated agent output bounds repair cost.

6.1 Modeled Time Savings

The per-workflow saving is $S(w) = T_{\text{base}}(w) - T_{\text{bibby}}(w)$, and the expected monthly saving per researcher is

$$\mathbb{E}[S_{\text{month}}] = \sum_{w \in \mathcal{W}} f_w \, S(w), \tag{2}$$

where f_w is the monthly frequency of workflow w. Table 3 instantiates the model with stage-time estimates from onboarding interviews and task decomposition; these are *modeled estimates* pending validation against production telemetry, and the parameterization is published so any term can be independently re-estimated.

Under this parameterization, Eq. (2) yields

$$\mathbb{E}[S_{\text{month}}] \approx 19(8) + 82(1) + 160(0.5) + 26(2) + 15(6) = 456 \text{ min} \approx 7.6 \text{ h} \tag{3}$$

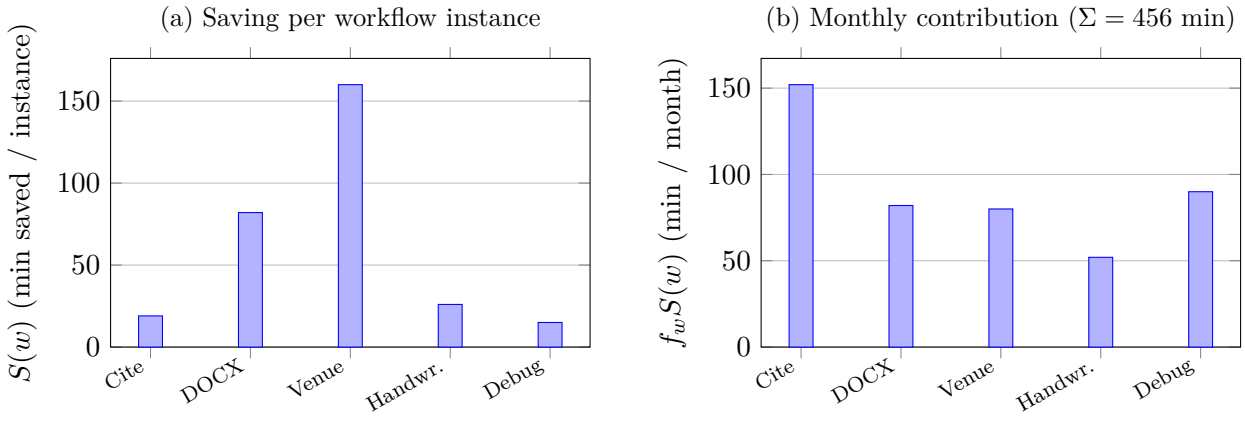

Figure 2: Modeled time savings under the parameterization of Table 3. (a) Per-instance saving is largest for venue retargeting and format conversion, where the fragmented baseline is dominated by repair cost. (b) Weighted by frequency, high-recurrence workflows (citation insertion, debugging) contribute comparably to rare high-cost ones.

per researcher per month—roughly **a full working day returned to research every month**, dominated by the elimination of switch and repair terms rather than by faster typing. Aggregated across the active user base, the model implies on the order of $5{,}000 \times 7.6 \approx 38\,000$ researcher-hours recovered monthly. Figure 2 visualizes the per-workflow savings and the monthly contribution of each workflow.

Our measurement program instruments the t^{switch} and t^{repair} terms in production telemetry and compares against baseline timings collected in structured onboarding interviews; the modeled values above define the pre-registered hypotheses for that measurement.

7 Discussion and Limitations

Owning the platform is a bet. Editor-nativeness trades the plugin approach's distribution advantage for architectural control. Our adoption data suggests the trade is viable at institutional scale: a substantial share of incoming users migrate from Overleaf and comparable editors, and qualitative feedback from these switchers consistently cites the integrated pipeline as the reason for the move. Migration friction remains the primary growth constraint, which is exactly why ingestion pipelines (Section 3.2) are core infrastructure rather than conveniences.

Patent signals are a lower bound. Front-page patent citations undercount science–technology linkage (in-text patent citations and non-patented use are invisible), and coverage is USPTO-centric [3]. We treat the signal as evidence of impact where present, never as evidence of absence.

Agent trust. All agent edits are diff-reviewed and compile-validated, but retrieval-grounded drafting can still select real-but-suboptimal references. Keeping the author in the loop at citation-choice time, with impact signals visible, is our current mitigation; automated claim–citation entailment checking is planned work.

8 Conclusion

Bibby AI demonstrates that the recurring engineering obstacles of LLM-assisted academic writing—editor synchronization, unverifiable edits, fragmented retrieval—are artifacts of building assistants *outside* the editor. By making the editor, compiler, reference graph, and agents one system, the platform delivers compile-verified agentic editing, single-interface research-to-publication workflows, and a translational-impact citation signal unavailable elsewhere. The platform is validated by production adoption across 5,000+ researchers and 50+ universities, and our workflow cost model estimates roughly 7.6 h returned to each researcher per month—approximately 38 000 researcher-hours across the user base—pending confirmation by the instrumented telemetry program of Section 6.1.

References

[1] Junyi Hou et al. *PaperDebugger: A Plugin-Based Multi-Agent System for In-Editor Academic Writing, Review, and Editing*. 2025. arXiv: 2512.02589 [cs.AI]. URL: https://arxiv.org/abs/2512.02589.

[2] PatentsView. *PatentsView: Disambiguated USPTO Patent Data*. https://patentsview.org. United States Patent and Trademark Office data platform. 2024.

[3] Matt Marx and Aaron Fuegi. "Reliance on Science: Worldwide Front-Page Patent Citations to Scientific Articles". In: *Strategic Management Journal* 41.9 (2020), pp. 1572–1594.

[4] Nuo Chen et al. *XtraGPT: Context-Aware and Controllable Academic Paper Revision*. 2025. arXiv: 2505.11336 [cs.CL]. URL: https://arxiv.org/abs/2505.11336.

[5] Rodney Kinney, Chloe Anastasiades, Russell Authur, et al. *The Semantic Scholar Open Data Platform*. 2023. arXiv: 2301.10140 [cs.DL]. URL: https://arxiv.org/abs/2301.10140.

[6] Jason Priem, Heather Piwowar, and Richard Orr. *OpenAlex: A Fully-Open Index of Scholarly Works, Authors, Venues, Institutions, and Concepts*. 2022. arXiv: 2205.01833 [cs.DL]. URL: https://arxiv.org/abs/2205.01833.