

Gap Analysis of Annotation Tools for AfricaNLP

*A study of 59 annotation tools against the needs
of AfricaNLP researchers*



Prepared by: HausaNLP, Bayero University, Kano, Nigeria
EthioNLP, Bahir Dar University (ICT4D), Bahir Dar, Ethiopia

Submitted to: Masakhane African Languages Hub

Grant: Annotation and Playbook Grant

Report date: 17 July 2026

Table of Contents

1	Executive Summary	3
2	The Problem	3
3	Methodology	4
3.1	Study design and research questions	4
3.2	Comparison criteria: selection and justification	5
3.3	Community surveys	7
4	Findings: What the Community Told Us	7
4.1	What African-language NLP practitioners struggle with today	7
4.2	Auditing the tool landscape against African-language needs	8
4.3	What published African-language datasets used	9
4.4	Where existing tools fall short of the 14 criteria	10
5	Bridging the Gap: What We Will Build	11
5.1	Why no existing tool closes the gap	11
5.2	Offline and mobile annotation	12
5.3	African scripts and keyboards	12
5.4	Speech recording and multimodal annotation	13
5.5	Quality management	13
5.6	Consent, licensing, and governance	13
5.7	Team and organization management	13
5.8	Integration with the research ecosystem	13
5.9	What we will inherit from Label Studio	14
6	Conclusion	14
	Appendix A: Full Capability Matrices	15
	Appendix B: Abbreviations	27
	Appendix C: Evidence References	27



1 Executive Summary

Advances in modeling have accelerated natural language processing (NLP) research worldwide, yet the tools available for creating African-language datasets have not kept pace. In a practitioner survey, AfricaNLP researchers report that their most-used annotation tool is a general-purpose spreadsheet (e.g., Microsoft Excel or Google Sheets), not a dedicated annotation platform. They rank their hardest day-to-day problems as recruiting native speakers, limited language support in existing tools, lack of offline and mobile access, and keyboard and input difficulties.

A systematic review of 59 annotation tools confirms that African-language needs go largely unmet: fewer than one in five tools (10) offer sourcing of African-language speakers, fewer than one in four (14) provide annotator qualification screening, and nearly half (29) require a constant Internet connection, ruling out the low-connectivity settings where much African-language annotation takes place.

Further, an audit of 96 published African-language dataset papers tells the same story: only 9 report using a dedicated annotation platform. In effect, researchers are building datasets for some of the world's most linguistically diverse languages with tools designed for neither the languages nor the conditions under which the work takes place.

These findings motivate AfriAnnotate, an open-source annotation platform designed to address all 14 platform-level capabilities our study identifies as essential for African-language annotation; no existing tool in our audit covers more than six capabilities. Its core design priorities include offline-capable annotation for the many African regions with limited Internet connectivity, native script and keyboard support for right-to-left and non-Latin writing systems such as Ge'ez, and a mobile-friendly interface so that annotators who rely on phones rather than laptops can participate fully. Taken together, the survey, the tool audit, and the literature review show a quantified need that no current tool meets; nor can any practical assembly of existing platforms cover all 14 capabilities. AfriAnnotate is designed to close this gap in a single open-source codebase.

2 The Problem

African-language NLP (AfricaNLP) work faces operational constraints that make annotation tool selection unusually consequential. The tools that do exist fall into three camps, and none fully serves the community. Commercial platforms (e.g., Label Studio Enterprise, Prodigy, Toloka, Appen) are paywalled, putting them out of reach for most African research groups. Open-source alternatives (e.g., upstream Label Studio, Doccano, Argilla) require self-hosting, cloud infrastructure, and DevOps skills that many researchers and linguists lack. A third category, custom tools built in-house by individual companies or labs, may solve a narrow problem for their creators but rarely scales beyond a single project and is seldom released as free, open-source software that the wider community can reuse.

The people best placed to build African-language datasets, native-speaker linguists, students, teachers, and community organizations, are therefore left without fit-for-purpose tooling. Beyond this access gap, seven operational constraints compound the problem:

- **Data collection starts from scratch.** Unlike well-resourced languages, most African languages lack existing datasets. AfricaNLP teams must collect data from the ground up, so

choosing the wrong tool locks a project into workflow debt for the lifetime of the corpus.

- **Many annotators work on mobile devices.** For many AfricaNLP annotators, a phone is their only device. Yet few existing tools are fully usable on mobile (Section 4.2), making mobile-first design and touch-friendly annotation essential.
- **Connectivity is unreliable and expensive.** AfricaNLP annotators often work in areas where internet access is intermittent and mobile data is costly. A tool that requires a constant connection will stall the moment the link drops, and dropped work discourages annotators from continuing. Offline annotation, safe local caching, and reliable synchronization are therefore baseline requirements (Section 4.1).
- **Scripts and input methods are poorly supported.** AfricaNLP projects routinely work with non-Latin writing systems such as Arabic-based Ajami, Ge'ez, and Tifinagh. However, most annotation tools neither render these scripts correctly nor segment their words properly. Also, keyboards for tonal diacritics and phonetic input are scarce, forcing annotators into workarounds that introduce errors (Section 4.1).
- **Annotators are hard to find.** Most African languages have no established pool of trained annotators. AfricaNLP teams recruit native speakers ad hoc through community networks, Slack channels, and social media. No existing annotation tool helps with this recruitment step (Section 4.1).
- **Paying annotators is difficult.** AfricaNLP annotation often involves cross-border payments (a project lead in one country paying annotators in several others), yet few existing tools support annotator payment at all (Sections 4.1 and 4.2).
- **Existing tools lack corpus governance features.** AfricaNLP datasets are often contributed by community members who may not be familiar with data licensing; per-contributor consent, per-project licensing, and provenance tracking must hold up when the dataset is released.

No existing survey compares annotation tools against the specific needs of African-language NLP research. The general-purpose comparisons the community has relied on are often out of date, and vendor claims frequently overstate what an open-source edition actually provides. Researchers have therefore chosen tools based on vendor pages and word of mouth, or defaulted to spreadsheets (Section 4.1).

This study identifies the features African-language NLP researchers need, documents which of those features are missing from existing tools, and motivates the development of a purpose-built platform to close the gap.

3 Methodology

3.1 Study design and research questions

We approach the problem from both sides: what existing tools provide (supply) and what African-language researchers need (demand). A 59-tool capability audit covers the first; practitioner surveys cover the second. The study addresses three research questions:

RQ1. Which platform features matter for an African-language annotation project?

RQ2. Which tools provide the most needed features, based on public evidence, that anyone can check?

RQ3. Does evidence from the community itself (a practitioner survey and an audit of published dataset papers) confirm the same needs and the same gaps?

Three instruments were used to answer them:

1. A capability comparison and community screening of 59 tools (RQ1, RQ2, and RQ3).
2. A needs assessment survey of 26 practitioners (RQ3).
3. An audit of 96 published dataset papers (RQ3).

Section 3.2 introduces the 14 comparison criteria we use to score each tool, spanning team-level capabilities (e.g., crowdsourcing, offline access, mobile support, script and keyboard support, licensing, consent, and multimodal annotation) and platform-level capabilities (auto-suspend, role-based access control, Hugging Face Hub integration, in-editor computer-assisted translation, cache safety, knowledge base with typed features, and a Python SDK).

Section 3.3 details the three community instruments: a needs-assessment survey of 26 AfricaNLP practitioners covering tools used, challenges faced, and desired features; an audit of 96 published African-language dataset papers recording which annotation tools each study used; and a manual screening of 59 annotation tools against the 14 criteria.

3.2 Comparison criteria: selection and justification

We scored each of the 59 tools against 14 platform-level capabilities derived from the constraints in Section 2 and checked against the practitioner survey (Section 4). Table 1 states each capability and its justification. The first seven concern the team platform (who governs and quality-controls the work); the last seven concern the annotation surface and integration.

Table 1: The 14 comparison criteria and the justification for each.*

Capability	What it means	Why it is a criterion
Crowdsource	A cross-organization pull-flow (annotators apply into open projects) plus an invited push-flow	Growing the annotator pool without new-organization friction is the practical path to scale.
Offline	Per-project annotator and reviewer offline toggles; native desktop and mobile shells with hardened sync	Distributed African-language teams work on constrained connectivity by default.
Mobile access	Annotation on phones and tablets, which for many annotators are the only available device	For many annotators, a phone or tablet is the only available device, so the labeling surface itself must work there.
Script and keyboard support	Rendering and input for RTL and non-Latin scripts	Limited language support, keyboard and input difficulties, and font and Unicode issues were among the most cited obstacles in the needs-assessment survey, so rendering and input for RTL and non-Latin scripts are scored across the tool set.
License	A per-project SPDX license chosen at design time, attached as machine-readable metadata	A dataset released without a machine-readable license is legally and operationally under-defined.
Consent	Per-contributor, per-purpose consent snapshots with a tamper-evident audit trail	Documented consent is a practical prerequisite for lawful and ethical dataset release under regimes such as the GDPR.
Multimodal canvas	One labeling surface binding audio, video, aligned transcripts, and time-series under one interface	The “one recording, many views” pattern is central to speech, gesture, and audio-video datasets.
Auto-suspend	Automatic quality-based annotator suspension (inactivity, low agreement, rejection rate)	Manual review of annotator quality does not scale.
RBAC	Role-permission enforcement at the data layer, not only URL gating	A multi-annotator program cannot audit or govern access without it.
HF Hub bidirectional	Import of Hugging Face Hub datasets with provenance; push of annotations back with aggregation and typed columns	The Hub is the primary distribution surface for machine-learning datasets; a first-class round trip is the expected workflow.
In-editor CAT	Translation memory, glossary, fuzzy lookup, industry-format ingest (XLIFF, TMX, DOCX)	Translation work at scale is impractical in a bare text box.
Cache-safe	No-store caching on API responses, search-index exclusion on annotation URLs, token rotation	Annotation data leaked to browser caches or search indexes is a real compliance risk.
KB + typed features	A knowledge-based backend, typed feature layers, entity linking, typed attribute forms on regions	Entity linking and structured attributes are the standard layer above NER in biomedical, legal, historical, and cross-lingual work.
Python SDK	A maintained, pip-installable client covering projects, tasks, and the Hub round trip	Data teams script platforms from notebooks and CI; a REST-only tool adds friction.

* A 15th dimension, task design flexibility, is classified separately (composable, template-bound, fixed-task, or bring-your-own) and reported in Appendix A.

Note: We selected the comparison criteria and a different set of criteria could rank tools

differently; the survey provides only a partial check on the criteria.

3.3 Community surveys

We collected evidence from three instruments, each targeting a different angle of the problem.

- **Needs assessment survey:** We surveyed 26 AfricaNLP practitioners (13 researchers, 5 data annotators, 4 NLP engineers, 3 students, 1 linguist) recruited through Masakhane, HausaNLP, and EthioNLP. The survey covered tools used, annotation types performed, challenges encountered, and desired features. Because this is a convenience sample, the results cannot be treated as representative of all practitioners.
- **Dataset paper audit:** We reviewed 96 published African-language dataset papers spanning text, speech, and multimodal work, recording the annotation tool each paper used. Some papers rely on proprietary tools whose names are not disclosed, so those tools could not be compared.
- **Community tools screen:** We reviewed 59 annotation tools against the 14 capability criteria in Table 1 and seven task modalities (text classification, sequence labeling, relation extraction, speech transcription, speech recording, translation, and image annotation).

Section 4 presents what these three instruments found.

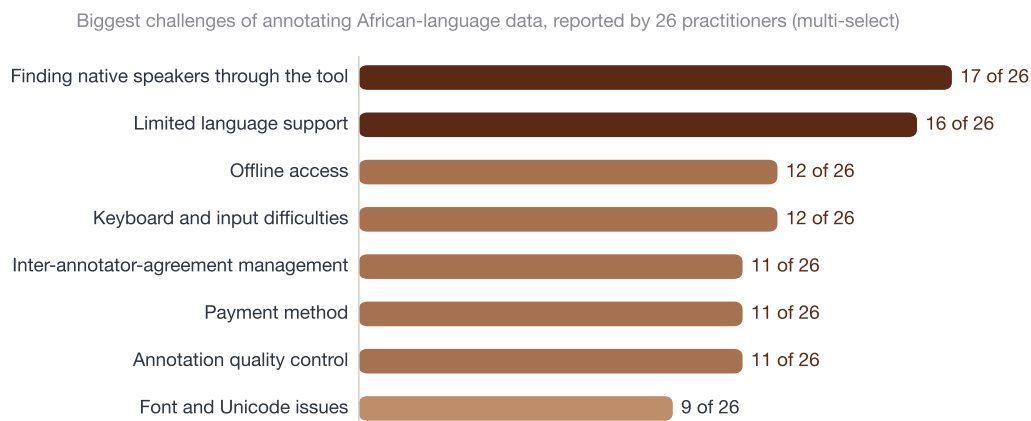
4 Findings: What the Community Told Us

The three instruments in Section 3.3 examine the problem from the practitioners' side. All three converge on the same finding: existing tools do not serve African-language annotation projects well, and the gaps practitioners report align with the 14 comparison criteria.

4.1 What African-language NLP practitioners struggle with today

In the practitioner survey ($n = 26$), we asked respondents to name the biggest challenges of annotating African-language data. Finding native speakers through the tool ranked first (17 of 26), followed by limited language support (16), offline access (12), and keyboard and input difficulties (12). Inter-annotator agreement management, payment, and annotation quality control tied at 11 each. They also identified font and Unicode issues as among the biggest challenges. Figure 1 shows the full ranking.

Native-speaker sourcing tops the challenge list



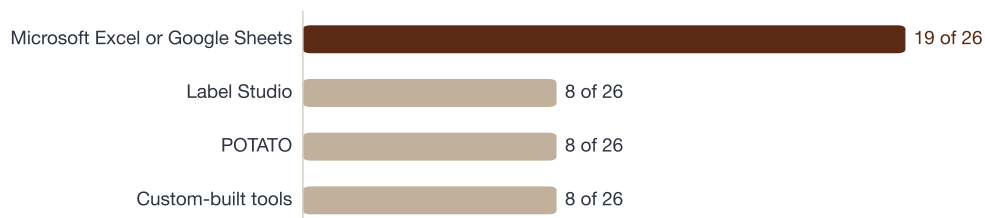
Source: AfriAnnotate needs-assessment survey, July 2026 (n = 26).

Figure 1: Needs-assessment survey (n = 26; multi-select).

The same survey asked which annotation tools respondents had used. The most frequent answer was a spreadsheet: Microsoft Excel or Google Sheets (19 of 26), more than double the count of Label Studio, POTATO, or custom-built tools (8 each). In the free-text responses, respondents describe spreadsheets as a fallback, consistent with current tools' inability to meet these practitioners' operating needs.

The most-used annotation tool is a spreadsheet

Annotation tools respondents had used before (multi-select); free-text answers describe spreadsheets as a fallback



Source: AfriAnnotate needs-assessment survey, July 2026 (n = 26).

Figure 2: Needs-assessment survey (n = 26; multi-select).

4.2 Auditing the tool landscape against African-language needs

In the community tool screen, we evaluated 59 annotation tools and find that the capabilities AfricaNLP practitioners rank highest are the least common (Figure 3).

- Mainstream capabilities are well represented: 43 of 59 tools offer user and project management, and 37 support agreement scoring.
- The capabilities specific to African-language work, however, are rare. Only 30 of 59 tools work offline, 13 run on mobile, and 10 or fewer support annotator qualification screening, annotator payment, or African-language-speaker sourcing, the need respondents ranked first.

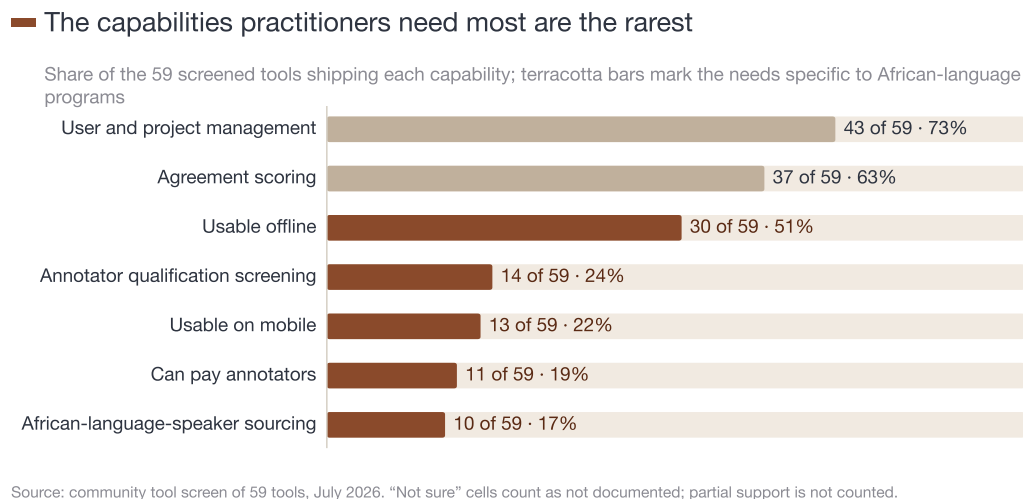


Figure 3: Capability audit of 59 tools. "Usable offline" is the complement of the 29 online-only tools.

Coverage skews toward text. Most tools handle text classification or sequence labeling; fewer support speech annotation, and very few offer speech recording. Yet speech recording is the modality that most African speech-corpus projects need first, and it is a primary target for AfriAnnotate.

The tool audit shows what is available; the dataset-paper audit in the next section (Section 4.3) shows what researchers actually chose.

4.3 What published African-language datasets used

In the dataset-paper audit, we reviewed 96 published AfricaNLP dataset papers to identify which annotation tools researchers actually used. Only 9 of the 96 papers reported the use of a dedicated annotation platform (e.g., ELISA, LightTag, POTATO, Mechanical Turk, or Label Studio). The rest rely on manual work without a named tool (33), automated or scraped pipelines (20), studio or community recording (15), or custom in-house tools (8); 8 papers do not state their tooling at all. Figure 4 gives the full breakdown.

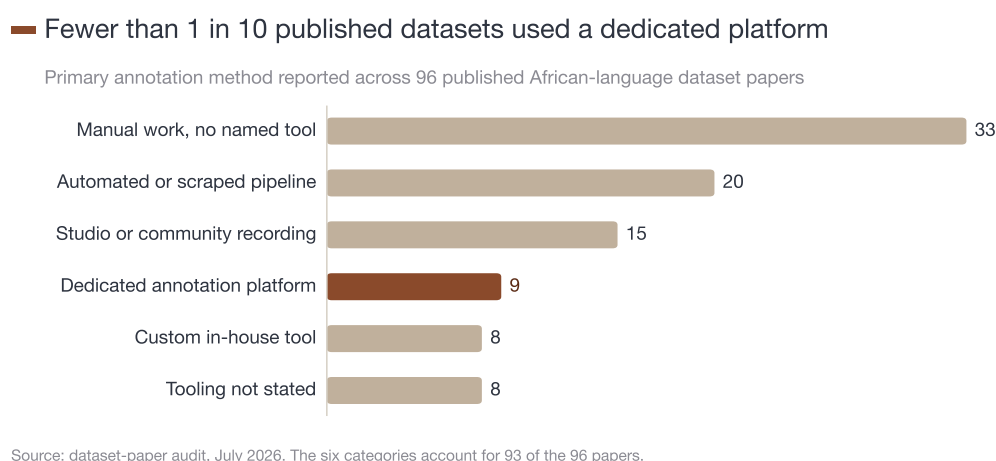


Figure 4: Dataset-paper audit. The highlighted bar is papers using any dedicated platform.

Projects that rely on manual work, spreadsheets, or one-off custom tools lack consent tracking, agreement scoring, and reusable workflows for the lifetime of the dataset. Several teams found existing tools so unsuitable that they built their own rather than adopting any available platform.

The tool screen and the dataset-paper audit converge on the same conclusion: the capabilities AfricaNLP practitioners need most (offline access, mobile support, speaker sourcing, annotator payment) are the ones fewest tools provide, and fewer than 1 in 10 published datasets were built with a dedicated annotation platform. Section 4.4 maps each of the 59 tools against the full set of 14 criteria, organized by practitioner priority, to quantify exactly where the gaps fall and how wide they are.

4.4 Where existing tools fall short of the 14 criteria

We now map the 59 tools from the community tool screen against the 14 criteria (Table 1), organized by practitioner priority. For five of the seven needs below, the practitioner survey and tool screen agree: practitioners rank the need high and tools rarely meet it. Two needs (mobile access and payment) require additional argument, which we flag in their respective entries.

- **Native-speaker recruitment:** Practitioners rank finding native speakers as their top challenge (17 of 26), yet only 10 of 59 tools offer any recruitment support.
- **Script and keyboard support:** Limited language support ranks second among practitioner challenges (16 of 26). On the tool side, only 16 of 59 tools support right-to-left or non-Latin scripts (e.g., Ge'ez, Hausa Ajami) and 5 do so partially. Respondents also request custom phonetic and diacritic keyboards, a problem that sits at the operating-system level rather than the annotation-tool level.
- **Offline access:** Offline access ranks among the top challenges (12 of 26), yet 29 of 59 tools work online only. Section 5 describes how AfriAnnotate addresses this gap.
- **Quality control:** Inter-annotator-agreement management and annotation quality control both rank among the top needs (11 of 26 each). On the tool side, 37 of 59 tools support agreement scoring, making this the best-served need in the list, but few tools go beyond a single agreement metric to offer configurable quality gates or adjudication workflows.
- **Mobile access (flagged):** Only 13 of 59 tools are usable on mobile, yet only 4 of 26 survey respondents name mobile inaccessibility as a challenge. We retain mobile access as a criterion despite the low survey signal because of sample composition: 13 of our 26 respondents are researchers and only 5 are data annotators, under-representing the phone-first population most likely to need mobile annotation. The same respondents rank finding native speakers as their top priority (17 of 26), and native speakers are precisely that phone-first population. The argument is inferential, not demand-driven.
- **Speech work:** Nearly half of respondents (12 of 26) annotate speech data, yet only 7 of 59 tools support speech recording, and fewer still offer transcript alignment or audio-specific quality control.
- **Payment:** Respondents identify annotator payment as a need (11 of 26), and 11 of 59 tools claim to support it. However, “support” here typically means integration with a third-party

service (e.g., Mechanical Turk payments), not built-in cross-border disbursement, which is the form most relevant to African-language annotation projects.

5 Bridging the Gap: What We Will Build

5.1 Why no existing tool closes the gap

Section 4 establishes that no existing tool addresses more than 6 of the 14 capabilities we identify as essential; the closest competitor, POTATO, documents 6, and most tools cover 3 or fewer (Table 2). These findings motivate AfriAnnotate, an open-source annotation platform we propose to develop on top of Label Studio, designed to integrate all 14 capabilities into a single codebase and give AfricaNLP researchers a purpose-built environment for building high-quality datasets, from recruitment and consent through annotation, quality control, and publication to the Hugging Face Hub.

Table 3 shows the per-criterion breakdown for AfriAnnotate and the five closest competitors.

Table 2: Documented capability counts across the 14 criteria. “Not documented” cells make competitor counts lower bounds.

Tool	Capabilities documented as shipped (of 14)	Additional “Partial” entries
AfriAnnotate	14 (Planned)	0
POTATO	6	3
Argilla 2.x	5	0
Roboflow	5	0
Toloka / Mindrift	5	0
Prodigy (Explosion AI)	4	0
CATMA	3	2
Datasaur	3	2
Appen	3	1
CloudFactory	3	1
All other reviewed tools	0–3 each	0–3 each

Table 3 makes this pattern visible at a glance, showing AfriAnnotate and the five closest competitors across all 14 criteria, simplified from the full Appendix A matrices.

Table 3: The 14 criteria for AfriAnnotate and the five closest competitors, simplified from the Appendix A matrices. ✓ shipped; P = Partial (paid tier or named caveats); ✗ not shipped; – not documented; descriptive cells that do not meet a criterion appear as ✗. Auto = auto-suspend; Cache = cache-safe; MM = multimodal canvas. AfriAnnotate’s ✓ marks represent planned capabilities, not audited shipped features.

Tool	RBAC	Auto	Consent	License	Cache	Offline	Crowd	Mobile	Script	CAT	KB	MM	HF	SDK
AfriAnnotate	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
POTATO	✓	✓	P	✗	✗	P	✓	✓	–	✗	✓	P	✓	✗
Argilla 2.x	–	✗	–	–	–	–	✓	✓	✓	✗	✗	✗	✓	✓
Roboflow	✓	✗	–	✓	–	✗	✓	✗	–	✗	✗	✗	✓	✓
Toloka / Min-drift	✗	✓	✓	–	–	✗	✓	✓	–	✗	✗	✗	–	✓
Prodigy (Exploration AI)	✗	✗	✗	✗	–	✓	✗	✓	✓	✗	✗	✗	✗	✓

The remainder of this section describes the capabilities we will implement in AfriAnnotate, each answering a gap documented earlier in this report. Because AfriAnnotate will be built on Label Studio, the labeling editor, more than fifty task types, and a stable, well-documented interface already exist; our work will go into the capabilities that African-language teams need and that no existing tool provides: the script-support fixes and roughly forty additional features spanning governance, operations, new labeling tools, quality gates, and native offline distribution. The complete matrices behind Table 2 and Table 3 are reproduced in Appendix A: team-platform capabilities (Matrix 1), annotation-surface and integration capabilities (Matrix 2), and the configuration audit.

5.2 Offline and mobile annotation

Our survey ranks offline access among the top challenges (12 of 26 respondents), and for many annotators a phone is the only available device. AfriAnnotate will work offline by design. We will build native desktop and mobile applications around a local data store that continues to work when the connection drops and synchronizes safely when it returns. Downloads will resume where they stopped, interrupted uploads will not create duplicates, and touch input will work with the same precision as a mouse. Project managers will be able to enable or disable offline access for each project to protect sensitive data.

5.3 African scripts and keyboards

Many African languages are written in Arabic-based Ajami, Ge’ez, Tifinagh, and other non-Latin scripts. AfriAnnotate will provide support for these unsupported scripts. Word and sentence boundaries will be detected with language-aware rules, so selecting a word in Amharic will work as reliably as it does in English.

5.4 Speech recording and multimodal annotation

Nearly half of our survey respondents (12 of 26) annotate speech, and most African speech projects must record their data before they can label it. AfriAnnotate will support recording directly in the browser or on a phone. Automatic quality checks will flag clipped, noisy, or too-quiet recordings as they arrive, and a re-recording can be requested in one step. We will also provide tools that align a transcript with its audio word by word, and a shared canvas that presents audio, video, transcript, and measurements in one view. This supports the one-recording-many-views pattern that field linguists rely on and that no reviewed tool offers in a single surface.

5.5 Quality management

AfriAnnotate will build quality management into the annotation workflow. Agreement between annotators will be scored continuously, disagreements will be routed to a reviewer, and annotators whose quality drops will be paused automatically, with clear warnings and a path back. A full review workflow with accept, reject, and fix decisions will apply to every annotation.

5.6 Consent, licensing, and governance

African-language datasets carry obligations to the people who contribute to them. In AfriAnnotate, every contributor will sign a clear, versioned consent form, with eight kinds of consent including voice release, recorded in a tamper-evident log. Every project will choose a machine-readable license through a guided wizard before data collection starts. A six-role permission system will control what each person can see and do, enforced in the platform's data layer, and every sensitive action will be recorded in an append-only audit log.

5.7 Team and organization management

Research groups run many projects with many partners. In AfriAnnotate, each organization will have its own members, projects, branding, and consent library, and one person will be able to belong to several organizations. The platform will notify people about events that concern them, such as a rejected annotation, a growing review queue, or a revoked consent, by email or in the application. Administrators will get dashboards, quotas, and maintenance tools designed for small teams and modest budgets.

5.8 Integration with the research ecosystem

AfriAnnotate will connect directly to the Hugging Face Hub, the main distribution platform for machine-learning datasets. Projects will import data from the Hub with full provenance and push finished annotations back with the license, consent summary, and per-annotator credit attached automatically. A Python library will allow research teams to script the platform from their own tools, and knowledge-base annotation will allow annotators to link words to curated concepts such as Wikidata entries, a standard requirement for African-language named-entity and semantic annotation.

5.9 What we will inherit from Label Studio

AfriAnnotate will inherit what Label Studio already does well: the labeling editor, the 52 existing task types spanning text, image, audio, video, and documents, and the stable interfaces its community already knows. On top of that foundation we will add roughly forty capabilities across governance, operations, new labeling tools, quality gates, and native offline distribution. This project will therefore develop only the missing pieces, and the fixes we develop along the way, starting with the script-support patches, will be contributed upstream so the wider open-source community benefits as well.

6 Conclusion

This study produces two deliverables: (1) the gap analysis documented in this report, including the practitioner survey, the 59-tool capability audit, and the 96-paper dataset audit; and (2) a repeatable audit methodology that can be refreshed as the tool landscape evolves. Upon completion, AfriAnnotate will additionally produce public landscape documentation, an at-a-glance capability matrix, and a tool-selection guide on its documentation site.

Our study establishes three converging lines of evidence. First, the community survey confirms that AfricaNLP researchers face a consistent set of unmet tooling needs—native-speaker recruitment, offline access, script support, quality control, and multimodal annotation—and that most teams default to spreadsheets because no available tool meets their constraints. Second, the 59-tool capability audit shows that no single existing platform addresses more than 6 of the 14 capabilities we identify as essential, and that assembling all 14 from the market would require at least 5 non-interoperable tools. Third, the 96-paper dataset audit documents the consequence: a community building some of the world’s most linguistically diverse datasets on tools designed for neither the languages nor the conditions.

These findings provide a clear and evidence-based justification for AfriAnnotate. By integrating all 14 capabilities into a single open-source platform built on Label Studio, AfriAnnotate is designed to address every gap this study documents. The audit methodology is reproducible and the codebase will be open, so funders and the community alike can verify the claims and track progress.

In short: the AfricaNLP research community needs purpose-built annotation tooling, current tools do not provide it, and AfriAnnotate is designed to close that gap. A grant for AfriAnnotate is an investment in the infrastructure this community has so far been improvising without.

Appendix A: Full Capability Matrices

Evidence rules and scoring

Every capability claim had to satisfy at least one of four evidence rules: (1) a source-code check, meaning the vendor’s open repository contains the feature or measurably does not; (2) a published paper or a specific vendor-documentation URL naming the feature; (3) a public issue tracker documenting the gap; or (4) a hands-on test in a hosted or self-hosted instance. Each matrix cell is scored ✓ (ships it), ✗ (does not), “Partial” (paid tier only, or with named caveats), or “–” (not documented). A “–” records that no qualifying evidence could be located; it is not scored as a fail. Evidential depth varies by column and by tool, and the matrices should not be read as though it were uniform: where a capability is absent from a closed-source vendor’s documentation and no source-code check was possible, the cell records the absence of documented evidence rather than tested absence, and that tool’s count is a lower bound. One consequence should be carried into the findings: every “only tool that ships X” statement in this report means “the only tool documented to ship X under these rules,” bounded by the public evidence available for closed-source competitors.

For AfriAnnotate itself, the entries describe the proposed design rather than audited shipped evidence: each ✓ maps to a specified capability in the design (the data model behind every platform-capability claim, the permission model behind every RBAC claim, and the XML tag set behind every labeling claim). Once the platform is built, we will re-audit these entries against the implementation under the same four evidence rules applied to every other tool.

Legend: ✓ = ships it; ✗ = does not; – = not documented (no qualifying evidence located; not scored as a fail); “Partial” = only under a paid tier or with named caveats. AfriAnnotate entries represent proposed capabilities; all other tools are scored on documented shipped evidence. Because “–” cells are concentrated among closed-source commercial platforms, exclusivity statements are statements about *documented* capability. Tools no longer maintained are flagged in the Tool column (dormant, deprecated, archived, or defunct, as at the report date). An interactive rendering with full cell contents is published in the landscape documentation.

Table 4: Matrix 1. Team-platform capabilities across AfriAnnotate and the 59 reviewed tools.

Tool	RBAC	Auto-suspend	Consent	License	Cache-safe	Offline	Crowd
AfriAnnotate	✓ 6 roles	✓ 3 evaluators	✓ per invite	✓ SPDX + URL	✓ middleware	✓ per-project + hardened shells	✓ push + pull
Vanilla Label Studio	<i>Partial</i> (Enterprise)	✗	✗	✗	✗	✗	✗
INCEpTION	–	✗	✗	✗	✗	✗	✗
ELAN	✗ (single-user)	✗	✗	✗	✗	✓ (desktop)	✗
Audino	–	✗	✗	✗	✗	✗	✗
MateCat	–	✗	–	–	–	–	✗
OmegaT	✗ (single-user)	✗	✗	✗	✗	✓ (desktop)	✗
Doccano	–	✗	✗	✗	✗	–	✗
Argilla 2.x	–	✗	–	–	–	–	✓ (min_submitted)
Prodigy (Explosion AI)	✗ (per-user licensed)	✗	✗	✗	–	✓ (self-hosted)	✗
Datasaur	✓ (paid)	✗	–	–	–	✗	✗
tagtog	✓ (org roles)	✗	✗	✗	–	✗	✗
CVAT Community	–	✗	✗	✗	✗	✗	✗
CVAT Online / Enterprise	✓ (paid)	✗	–	–	–	✗	–
Roboflow	✓ (paid)	✗	–	✓ dataset license	–	✗	✓ marketplace
Kili	✓ 2-tier	–	–	–	–	✗	–
Labelbox	✓ 7 rigid	✗	–	–	–	✗	–
SuperAnnotate	<i>Partial</i>	–	–	–	–	✗	–
Encord	✓ 5 roles + tabs	Manual only	–	–	–	✗	–
Dataloop	<i>Partial</i>	✗	–	–	–	✗	–
V7 Labs	✓ 3 roles	Manual only	–	–	–	✗	–
Prolific	✗ 2-party	–	–	–	–	✗	✓ marketplace
Toloka / Mindrift	Requester + performer	✓ (SDK archived 2024; platform active)	✓ contract-layer	–	–	✗	✓ marketplace
Amazon Mechanical Turk	✗ 2-party	Manual only	✗	✗	✗	✗	✓ marketplace
Scale Rapid	–	–	–	–	–	✗	✓ workforce
brat (dormant)	✗	✗	✗	✗	✗	✗	✗

Continued on next page

Matrix 1 – continued from previous page

Tool	RBAC	Auto-suspend	Consent	License	Cache-safe	Offline	Crowd
WebAnno (deprecated)	✓ per-project annotator/curator/manager roles	✗	✗	✗	✗	✗	Partial (CrowdFlower bridge, removed in v3)
GATE Teamware (dormant)	✓ 3 roles; managers see all	Partial (auto gold-test entry gate only)	✗	✗	✗	✗	Partial (open signup pool, no marketplace)
POTATO	✓ admin/adjudicator/annotator + custom roles	✓ attention-check block_threshold auto-blocks	Partial (consent phase, stored responses; no audit trail)	✗	✗	Partial (mobile PWA, prefetched batches sync)	✓ Prolific/MTurk integration
Anafora (dormant)	Partial (annotator vs adjudicator group only)	✗	✗	✗	✗	✗	✗
TeamTat (dormant)	✓ admin/manager/annotator roles	✗	✗	✗	–	✗	✗
CATMA	✓ 4 roles, per-resource responsibility	✗	✗	✗	✗	Partial (Standalone Docker, local server)	✗
TextAE	✗	✗	✗	✗	– (REST client, undocumented)	Partial (client-side, JSON file import/export)	✗
PubAnnotation	✓ owner/associate-maintainer/root, model-level	✗	✗	✓ per-project license in JSON meta	–	✗	✗
ezTag (dormant)	✗	✗	✗	✗	–	✗	✗
Hypothesis	✓ group owner/admin/moderator/member roles	✗	✗	✗	✗	✗	✗
SLATE	✗	✗	✗	✗	✗	✓ local terminal app, Python-only	✗
Knowtator 2.0	✗	✗	✗	✗	✗	✓ desktop, file-based offline	✗
Universal Data Tool (dormant)	✗	✗	✗	✗	✗	✓ Electron desktop, local file://	✗
Diffgram (dormant)	✓ admin/editor/annotator + custom roles	✗	✗	✗	✗	✗	Partial (legacy billing-gated Market share)

Continued on next page

Matrix 1 – continued from previous page

Tool	RBAC	Auto-suspend	Consent	License	Cache-safe	Offline	Crowd
UbiAI	<i>Partial</i> (admin/ reviewer/labeler; paid tiers)	✗	–	–	–	✗	✗
LightTag (defunct)	✓ manager/reviewer/ annotator dataset perms	✗	–	–	–	✗	✗
LabelMe	✗	✗	✗	✗	✗	✓ fully offline desktop app	✗
VGG Image Annotator (VIA)	✗	✗	✗	✗	✗	✓ single HTML file, fully offline	✗
Make Sense (dormant)	✗	✗	✗	✗	✗	<i>Partial</i> (client-side; local hosting works disconnected)	✗
labelImg (archived)	✗	✗	✗	✗	✗	✓ fully offline desktop app	✗
Supervisely	✓ roles scope data to assigned jobs	✗	–	–	<i>Partial</i> (short-lived revocable tokens only)	✗	✗
Datature	<i>Partial</i> (paid tier, workspace roles)	✗	–	–	–	✗	✗
Labellerr	✓ six roles, per-role screens	✗	–	–	–	✗	<i>Partial</i> (paid managed workforce, no marketplace)
Sama	<i>Partial</i> (annotator/QA/ client portals; RBAC undocumented)	–	–	–	–	✗	✗
Praat	✗	✗	✗	✗	✗	✓ local-only desktop app	✗
SayMore	✗	✗	<i>Partial</i> (per-person, not per-purpose)	✗	✗	✓ offline Windows desktop app	✗
Audacity	✗	✗	✗	✗	✗	✓ fully offline desktop app	✗
TranscriberAG (dormant)	✗	✗	✗	✗	✗	✓ native desktop app (Win/Mac/Linux)	✗

Continued on next page

Matrix 1 – continued from previous page

Tool	RBAC	Auto-suspend	Consent	License	Cache-safe	Offline	Crowd
Appen	✓ 4 roles, team/org scoping	✓ auto-removal below min accuracy	–	–	–	✗	✓ core marketplace, crowd channels
iMerit (Ango Hub)	✓ org + project multi-role	✗	–	–	–	✗	<i>Partial</i> (managed iMerit workforce, no marketplace)
CloudFactory	✓ custom roles, 15 permissions	✗	–	–	–	✗	✓ managed workforce (push), paid service
Surge AI	–	–	–	–	–	✗	✓ managed workforce
Snorkel Flow	✓ workspace-scoped roles + connector RBAC	–	–	–	<i>Partial</i> (token rotation only)	–	–
Rasa	Different category: conversational-AI framework, not a per-item annotation tool.						

Table 5: Matrix 2. Annotation-surface and integration capabilities across AfriAnnotate and the 59 reviewed tools.

Tool	KB + typed features	Multimodal shared canvas	HF Hub bidirectional	Python SDK	Mobile	Script/keyboard	CAT
AfriAnnotate	✓ KBRef + TypedFeature + NEL template	✓ MultiModalCanvas + sync groups	✓ 3-tab wizard + multi-dataset + 3 push modes + 4 aggregation strategies	✓ pip install afriannotate	✓ web + native mobile app + desktop	✓ RTL + non-Latin patch set	✓ TM + glossary + TMX + fuzzy
Vanilla Label Studio	✗	✗	✗ (community integrations only)	✓ label-studio-sdk	<i>Partial</i> (reviewer screen)	<i>Partial</i> (RTL via Style-tag CSS workaround)	✗
INCEpTION	✓ deeper on formal typed-layer constraints	✗ (text-only)	✗	– (community pycaprio, not official)	✗	–	✗
ELAN	<i>Partial</i> (typed tiers)	✓ audio + video + tiers under one player (single-user)	✗	– (community pympi, not official)	✗	✓ Arabic keyboard map; font config	✗
Audino	✗	✗	✗	–	–	–	✗
MateCat	✗	✗	✗	✗	–	✓ per-language direction ltr/rtl	✓ specialist
OmegaT	✗	✗	✗	✗ (Java tool)	–	✓ Unicode bidi controls	✓ specialist
Doccano	✗	✗	✗	✓ doccano-client	✓ README: “Mobile support”	–	✗
Argilla 2.x	✗ (metadata, not KB backend)	✗ (text-only)	✓ (text-only import + push)	✓ argilla	✓ responsive; annotate on phone	✓ RTL auto-detection (v1.26.1)	✗
Prodigy (Explosion AI)	✗	✗	✗	✓ (native Python tool)	✓ touch swipe gestures	✓ writing_dir: rtl	✗
Datasaur	<i>Partial</i> (entity dictionaries)	✗ (text-only)	–	✓ (Datasaur Python SDK)	–	✓ Ara-bic/Mandarin/Cyrillic listed	<i>Partial</i> (translation)
tagtog	✓ hierarchical entity types + typed features	✗ (text-only)	–	–	<i>Partial</i> (reviewer screen)	✓ full RTL in editor (archived docs)	✗
CVAT Community	✗	✗	–	✓ cvat-sdk	✗	–	✗
CVAT Online / Enterprise	✗	✗	–	✓ cvat-sdk	–	–	✗
Roboflow	✗	✗	✓ (export to HF)	✓ roboflow	✗	–	✗
Kili	–	–	–	✓ kili-python-sdk	✗	–	✗

Continued on next page

Matrix 2 – continued from previous page

Tool	KB + typed features	Multimodal shared canvas	HF Hub bidirectional	Python SDK	Mobile	Script/keyboard	CAT
Labelbox	<i>Partial</i> (entity ontologies)	–	–	✓ labelbox	✗	–	✗
SuperAnnotate	–	–	–	✓ superannotate	✗	✓ LTR/RTL/ Auto in editor	✗
Encord	<i>Partial</i> (entity ontologies)	–	–	✓ encord	✗ docs: not supported on tablets	–	✗
Dataloop	–	–	–	✓ dtlpy	✗	✓ transliteration input (Indic; no RTL)	✗
V7 Labs	–	–	–	✓ darwin-py	✗	–	✗
Prolific	✗	✗	✗	✗	✓ mobile/tablet (not enforced)	–	✗
Toloka / Mindrift	✗	✗	–	✓ (SDK archived July 2024)	✓ (reviewer screen)	– unverifiable (docs retired)	✗
Amazon Mechanical Turk	✗	✗	✗	✓ (via boto3)	✓ mobile-friendly worker site	–	✗
Scale Rapid	–	–	–	– (REST only)	✓ (reviewer screen)	–	✗
brat (dormant)	✓ normalization to Wikipedia/OBO	✗	✗	– (community pybrat parser only)	✗	✓ Unicode, ~100 scripts (no RTL)	✗
WebAnno (deprecated)	<i>Partial</i> (typed feature layers; no KB)	✗	✗	✗	–	–	✗
GATE Teamware (dormant)	✗	✗	✗	✗	✗	✓ RTL + Unicode (paper)	✗
POTATO	✓ span linking to Wikidata/UMLS/custom	<i>Partial</i> (ELAN tiers: one medium + tiers)	✓ Hub source + annotation push	✗	✓ responsive; works on tablets	–	✗
Anafora (dormant)	<i>Partial</i> (typed properties, no entity linking)	✗	✗	✗	<i>Partial</i> (reviewer screen)	–	✗
TeamTat (dormant)	✓ concept IDs, MeSH/Gene linking	✗	✗	✗	✗	<i>Partial</i> (generic Unicode)	✗
CATMA	✓ hierarchical tagsets, typed properties	✗	✗	<i>Partial</i> (official GitMA, git-install, data-access)	✗ docs: touch annotation not supported	✓ Hebrew (RTL) guide	✗

Continued on next page

Matrix 2 – continued from previous page

Tool	KB + typed features	Multimodal shared canvas	HF Hub bidirectional	Python SDK	Mobile	Script/keyboard	CAT
TextAE	✓ entity linking, PubDictionaries autocomplete, attributes	✗	✗	✗	Partial (reviewer screen)	Partial (UTF-8 any language; limited testing)	✗
PubAnnotation	✓ dictionary / ontology entity linking	✗	✗	✗	Partial (reviewer screen)	–	✗
ezTag (dormant)	✓ links MeSH, NCBI Gene, dbSNP	✗	✗	✗	✗	–	✗
Hypothesis	✗	✗	✗	– (community hypothesis-api / python-hypothesis)	✓ via proxy / bookmarklet	–	✗
SLATE	✗	✗	✗	✗	✗	Partial (Unicode “should” work; untested)	✗
Knowtator 2.0	✓ OWL concept linking, typed relations	✗	✗	✗	✗	–	✗
Universal Data Tool (dormant)	✗	✗	✗	Partial (official pip, dataset-only, unmaintained)	✗	–	✗
Diffgram (dormant)	✓ typed attribute groups on instances	✗	✗	Partial (official SDK stale since 2023)	–	–	✗
UbiAI	✗	✗	✗	✗	✗	✓ Arabic annotation	✗
LightTag (defunct)	✗	✗	✗	✗	–	✓ RTL + Unicode (paper)	✗
LabelMe	✗	✗	✗	✗	✗	–	✗
VGG Image Annotator (VIA)	Partial (typed region attributes, no KB linking)	✗	✗	Partial (official python-via, early-stage 0.0.4)	✓ (reviewer screen)	–	✗
Make Sense (dormant)	✗	✗	✗	✗	✓ (reviewer screen)	–	✗
labellmg (archived)	✗	✗	✗	✗	✗ (desktop Qt application)	–	✗
Supervisely	Partial (typed object tags, no KB)	✗	✗	✓ official pip ‘supervisely’ SDK	✗	–	✗
Datature	✓ typed attributes per annotation	✗	–	✓ official PyPI package	Partial (reviewer screen)	–	✗

Continued on next page

Matrix 2 – continued from previous page

Tool	KB + typed features	Multimodal shared canvas	HF Hub bidirectional	Python SDK	Mobile	Script/keyboard	CAT
Labellerr	<i>Partial</i> (object attributes; no entity linking)	✗	–	✓ official SDK v2, GitHub-installed	–	–	✗
Sama	–	✗	–	<i>Partial</i> (official pip sama; last release 2023)	✗	–	–
Praat	✗	<i>Partial</i> (audio+transcript+pitch; no video)	✗	– (community: praat-parselmouth)	✗	✓ Unicode; Hebrew; font config	✗
SayMore	✗	<i>Partial</i> (media+transcript playhead; no time-series)	✗	✗	✗	–	✗
Audacity	✗	✗	✗	✗	✗ docs: no mobile version	<i>Partial</i> (RTL documented as imperfect; no IME in labels)	✗
TranscriberAG (dormant)	✗	<i>Partial</i> (audio/video+transcript sync; no time-series)	✗	✗	✗	✓ bidi “well handled”; RTL; IME	✗
Appen	–	–	–	– (community: crowdflower, figure-eight-client)	<i>Partial</i> (reviewer screen)	– unverified (source 404)	–
iMerit (Ango Hub)	<i>Partial</i> (region attributes; no entity linking)	✗	✗	✓ official imerit-ango PyPI package	–	–	✗
CloudFactory	<i>Partial</i> (typed attributes; no entity linking)	✗	–	✓ official hasty pip package	–	–	✗
Surge AI	–	–	–	✓ surge-api (official)	– unverifiable (docs gated)	– unverifiable (docs gated)	–
Snorkel Flow	<i>Partial</i> (dictionary entity linking; no KB backend)	–	–	✓ snorkelflow[sdk] (private index)	<i>Partial</i> (reviewer screen)	–	–
Rasa	Different category: conversational-AI framework, not a per-item annotation tool.						

Table 6: Configuration audit. Task and configuration design flexibility across AfriAnnotate and the 59 reviewed tools, with the community screen’s adaptability rating where one was recorded (– = screened without a rating; “not screened” = tool was not in the community screen).

Tool	Task/config model	Configuration surface	Community “easy to adapt” rating
AfriAnnotate	Composable	inherits Label Studio XML tag grammar; adds 5 tags	not screened
Vanilla Label Studio	Composable	XML tag grammar: object/control/visual tags composed into arbitrary interfaces	medium
INCEpTION	Template-bound	Custom span/relation/chain layers and features; text editor interface fixed	medium
ELAN	Fixed-task	Single time-aligned tier interface; tier types and controlled vocabularies only	medium
Audino	Fixed-task	Single audio waveform interface; flexible label creation only	not screened
MateCat	Fixed-task	One hard-wired segment translation editor; config is languages/TM/files	not screened
OmegaT	Fixed-task	Fixed segment-by-segment CAT editor pane; project languages/resources only	not screened
Doccano	Template-bound	Fixed menu of project types; label configuration only	medium
Argilla 2.x	Composable	Settings composes arbitrary fields (incl. CustomField HTML/JS) and questions	easy
Prodigy (Explosion AI)	Composable	Blocks combine view_ids; Mustache HTML, custom JS, Python recipes	easy
Datasaur	Template-bound	Six NLP project templates; label sets/questions config only	not screened
tagtog	Fixed-task	One text editor; entity types, relations, doc labels config	easy
CVAT Community	Template-bound	Fixed CV shape types; label constructor / raw JSON labels only	easy
CVAT Online / Enterprise	Template-bound	Fixed CV shape types; label constructor / raw JSON labels only	not screened
Roboflow	Template-bound	Six fixed project types; configure classes within chosen type	easy
Kili	Template-bound	JSON interface nests four fixed job types per asset type	medium
Labelbox	Template-bound	Ontology config within per-asset-type editors; custom HTML editor sunset	easy
SuperAnnotate	Composable	Drag-and-drop/JSON Form Builder, custom components, Python code editor	easy
Encord	Template-bound	Ontology objects/classifications/attributes; fixed label editor per modality	easy
Dataloop	Composable	Custom studio apps via Python/JS SDK, full UI control	difficult
V7 Labs	Template-bound	Fixed main annotation types; classes, properties, sub-annotations only	easy
Prolific	Bring-your-own	Recruitment layer; studies link to external survey/experiment URL	not screened
Toloka / Mindrift	Composable	JSON component grammar: views, layouts, fields, helpers; arbitrary interfaces	easy
Amazon Mechanical Turk	Composable	HTMLQuestion: arbitrary HTML5/JS forms plus Crowd HTML Elements	easy
Scale Rapid	Template-bound	Pick template task type, configure taxonomy; markdown-only customization	medium
brat (dormant)	Fixed-task	One span/arc text UI; annotation.conf defines ontology only	medium
Continued on next page			

Configuration audit – continued from previous page

Tool	Task/config model	Configuration surface	Community “easy to adapt” rating
WebAnno (deprecated)	Fixed-task	Single text editor; span/relation/chain layers with tagsets only	–
GATE Teamware (dormant)	Composable	JSON widget array: html/text/radio/checkbox/selector + templating, conditional if, raw HTML	difficult
POTATO	Composable	YAML composes radio/checkbox/likert/span/pairwise/text + custom raw-HTML templates	easy
Anafora (dormant)	Fixed-task	XML schema configures entity/relation/property types; fixed text span+relation UI	easy
TeamTat (dormant)	Fixed-task	Configure entity/relation types only; hard-wired BioC text entity+relation interface	easy
CATMA	Fixed-task	Configure tagsets/properties; fixed text markup/tagging interface plus query language	easy
TextAE	Fixed-task	Fixed text-span editor; JSON config only predefines entity/relation/attribute types	medium
PubAnnotation	Fixed-task	Span/relation JSON model edited via TextAE; free label scheme, fixed interface	medium
ezTag (dormant)	Fixed-task	Hard-wired biomedical concept tagging; config is lexicons/pretrained taggers only	easy
Hypothesis	Fixed-task	Fixed highlight-comment-tag web annotation model; no task-type configuration	easy
SLATE	Fixed-task	Terminal text annotator; config file sets labels, link/scope flags only	easy
Knowtator 2.0	Fixed-task	Hard-wired Protege text-span/relation UI; config is OWL ontology only	difficult
Universal Data Tool (dormant)	Template-bound	udt.json interface field: fixed enum of ~18 types	medium
Diffgram (dormant)	Template-bound	Per-media-type interfaces; schema/attributes config, userscripts automate only	–
UbiAI	Template-bound	Fixed project types (NER/relations/classification/OCR/image); entity-relation label config	easy
LightTag (defunct)	Fixed-task	One text UI; schema = span tags plus classes only	–
LabelMe	Fixed-task	Fixed shape UI; .labelmerc YAML options, predefined labels/flags only	easy
VGG Image Annotator (VIA)	Fixed-task	Hard-wired region annotator; attribute schema (checkbox/radio/dropdown) config only	easy
Make Sense (dormant)	Fixed-task	Single image UI: rect/polygon/point/line plus label list	medium
labellmg (archived)	Fixed-task	Bounding boxes only; predefined_classes.txt and export format; archived	easy
Supervisely	Composable	Apps platform: Python SDK plus widgets builds custom labeling interfaces	easy
Datature	Template-bound	Fixed CV project types picked at creation; ontology/attribute config only	–
Labellerr	Template-bound	Pick data type, tools auto-load; configure labels/questions within	–
Sama	Template-bound	15+ predefined annotation types; Sama PM configures the project and client maps JSON	–
Praat	Fixed-task	Single TextGridEditor; interval/point tiers, free-text labels only	medium
SayMore	Fixed-task	Hard-wired BOLD segmenting/transcription workflow, no interface config	easy
Audacity	Fixed-task	Audio editor; label tracks only, freeform text, no task configuration	easy

Continued on next page

Configuration audit – continued from previous page

Tool	Task/config model	Configuration surface	Community “easy to adapt” rating
TranscriberAG (dormant)	Fixed-task	Hard-wired speech segmentation/transcription editor; turns, topics, acoustic conditions	easy
Appen	Composable	CML tag grammar plus custom HTML/CSS/JS job designer	–
iMerit (Ango Hub)	Template-bound	Fixed editors per media type; label-set/ontology config only	–
CloudFactory	Template-bound	Own CV-only editor (Hasty): fixed layout, class/attribute config; workforce otherwise uses client tools	–
Surge AI	Composable	Projects compose 11 question types, conditional logic, handlebars templates	–
Rasa	Different category	Conversational-AI framework	–
Snorkel Flow	–	–	–

Appendix B: Abbreviations

Term	Meaning
CAT / TM / TMX / XLIFF	Computer-assisted translation, translation memory, and the two standard translation exchange file formats (Translation Memory eXchange and XML Localization Interchange File Format).
GDPR	The European Union General Data Protection Regulation.
HF Hub	The Hugging Face Hub, the primary public distribution platform for machine learning datasets and models.
ICU / Intl.Segmenter	International Components for Unicode; Intl.Segmenter is the browser interface to ICU word- and sentence-boundary rules.
KB / NEL / NER	Knowledge base; named-entity linking (connecting an entity to a curated concept); named-entity recognition (marking entity spans in text).
NLP	Natural language processing.
QC	Quality control.
RBAC	Role-based access control: permissions granted by role (for example, reviewer or annotator) and enforced by the platform.
RTL	Right-to-left scripts, such as Arabic script and Ajami writing systems.
SDK	Software development kit: an installable client library for scripting a platform, here from Python.
SPDX	Software Package Data Exchange: a standard set of machine-readable license identifiers.
ASR / TTS	Automatic speech recognition / text-to-speech, the two main speech-corpus use cases.

Appendix C: Evidence References

- **Public landscape documentation:** the five pages listed under Deliverables (overview, at-a-glance matrix, layers, comparisons, and platform-specific features).
- **Open GitHub issues on vanilla Label Studio's RTL gap:** #1888 (2022-01-03), #2653 (2022-07-12), and #6642 (2024-11-13); all are open at the report date, with no merged pull request addressing text-span direction.
- **Toloka Python SDK:** archived July 2024. The Toloka/Mindrift platform itself remains in operation; the archived status applies to the SDK, not to the platform's suspension capability.
- **AfriAnnotate source repository:**
github.com/MasakhaneHubToolingProject/afriannotate,
 production branch, approximately 300 feature commits over the 12 months preceding this report.
- **Cross-referenced landscape statements:** every capability claim in the landscape documentation is traceable to a specific commit hash and file path in the source repository.