

Module 17

AI & library systems integration

For digital librarians ready to connect AI to the systems they manage (ILS, repositories, discovery layers) without a computer science degree. This module demystifies APIs and shows you what's actually possible.

By Yulia Brusova

15 min read · Reviewed October 2026

WHAT YOU'LL BE ABLE TO DO

- 1 Explain what an API is and how it enables AI integration with library systems
- 2 Identify integration opportunities between your ILS and AI tools
- 3 Evaluate whether to build, configure, or advocate for a vendor feature
- 4 Understand the data pipeline concept in the context of library systems
- 5 Read basic API documentation well enough to evaluate integration possibilities
- 6 Conduct a vendor AI due diligence review covering data privacy, algorithmic transparency, environmental cost, and contract terms, using ALA's AI guidance as a framework

The most common question digital librarians ask about AI integration is whether a given capability is something they can implement themselves or something that requires a developer. The honest answer is that it depends on the specific integration, the library systems involved, and the institution's technical environment. Some AI integrations are accessible today without any programming knowledge. Some are accessible with modest technical help. Some genuinely require developer involvement or a vendor solution. This module provides the grounding to know which category a given integration falls into, and to hold informed conversations with IT departments and vendors.

What an API is and why it matters for AI integration

An API, or Application Programming Interface, is a defined channel through which two software systems exchange data. The practical concept matters more than the acronym. When one system needs to send information to another, or request it, an API is the standardized mechanism that makes the exchange possible. The two systems do not have to be built by the same people or understand each other's internal architecture.

For example, when a patron searches the library's discovery layer and receives results from multiple licensed databases at once, APIs make that coordination possible. The discovery layer sends requests to each database's API, receives responses in a standardized format, and presents the combined results in a single interface. Similarly, when a library website displays current hours pulled automatically from an ILS rather than maintained by hand, an API is typically behind that connection.

This matters for AI integration because AI tools increasingly expose APIs of their own. Claude, ChatGPT, and most other major AI platforms offer APIs that let other systems send them content and receive processed responses automatically. Instead of a staff member copying catalog records into an AI interface and pasting the output into another system, the transfer can happen programmatically, at scale, and on a schedule. The ILS can send records to an AI service for description enhancement, and the discovery layer can request AI-generated summaries. The chat reference tool can use an AI API to draft initial responses. None of this requires manual intervention at each step.

Understanding APIs at a conceptual level, even without being able to build API connections yourself, changes how you evaluate vendor products and frame conversations with IT. A practitioner who knows that an AI integration is technically possible through existing API infrastructure is in a fundamentally different position in those conversations than one who does not.

What AI integration looks like in major ILS platforms

What AI integration is practically possible depends heavily on which integrated library system a library runs. No single answer applies across all platforms. Vendor-built AI features are also changing quickly, so consult current documentation and speak directly with vendors before drawing conclusions about what is or is not possible.

Alma, developed by Ex Libris, has one of the most accessible API ecosystems among commercial ILS platforms and an active developer community. Ex Libris also offers LibOW, a low-code automation platform that connects Alma to external services, including AI tools, without custom code. To evaluate what AI integrations are possible for an Alma installation, examine LibOW's available connectors before commissioning any custom development; the functionality you need may already be built in. For example, automated metadata enhancement on record import, batch processing of records through an external AI service, and reporting workflows fed from Alma data are all achievable through the Alma API ecosystem. The technical support each requires varies.

Sierra and other Innovative platforms use an older API architecture that works but typically requires developer involvement for custom integrations. These platforms are not precluded from AI integration, but the practical path more often runs through vendor-built features than custom development. Koha, as open-source software, offers more flexibility for custom API integration than most commercial platforms, provided the institution has technical staff or community support to implement the connection.

Before investigating any custom integration, the most important first step is a direct conversation with the ILS vendor. Vendors across the ILS market are actively building AI features into their platforms, and the integration a library needs may already be on the roadmap for an upcoming release. That conversation takes fifteen minutes and may save months of custom development. A practitioner who builds a custom metadata enhancement workflow in January may discover in March that the vendor has shipped the same capability as a supported feature. Asking the vendor first is not a sign of limited ambition; it is the operationally sound starting point.

Institutional repositories and AI: accessible entry points

For most digital librarians, institutional repository platforms offer more accessible AI integration opportunities than integrated library systems, for two reasons. First, IR platforms are often structurally simpler than ILS systems, with clearer data models and more straightforward API documentation. Second, the AI use cases for repositories are particularly well-defined. Metadata enhancement, description generation for underdescribed items, subject term suggestion, and discoverability improvement are all tasks where AI assistance produces clear, measurable value.

For DSpace installations, an active community of practitioners and developers is building AI integrations for metadata suggestion and quality control. For example, workflows that send item records to an AI service and get back suggested subject headings, enhanced abstracts, or normalized title fields are in active development and use in the DSpace community. Track these developments through the DSpace community forums and annual user group meetings rather than general search. They move quickly, and general sources often lag behind current practice.

bepress and Digital Commons are vendor-managed platforms with limited custom integration capability. For these installations, the practical path is advocacy with the vendor rather than independent development. These platforms restrict what users can implement outside the vendor's supported feature set, which makes the vendor conversation both more important and more constrained. Elsevier, which acquired bepress in 2017, has been integrating AI features across its research products, and the direction for Digital Commons is worth monitoring directly.

For most IR administrators today, the most practical AI-assisted metadata workflow requires no API integration at all. A practitioner can export a batch of item records as a CSV file and submit batches to Claude or another AI tool with a structured prompt. The prompt can request description enhancement, subject term suggestion, or abstract normalization. The

reviewed results are then imported back into the repository. This workflow is manual rather than automated, but it needs no technical support and meaningfully improves record quality in a fraction of the time needed to work through each record individually. It also serves as a proof of concept when the case for developer time or vendor investment in a more automated integration needs to be made.

Data pipelines: what they are and when they are appropriate

A data pipeline is an automated process that extracts data from one system, applies a transformation or processing step, and loads the result into another system. It runs on a recurring schedule, without manual intervention at each step. The concept is not specific to AI, but AI services fit naturally into pipelines as the processing step between extraction and loading.

For example, a library pipeline for AI-assisted metadata enhancement might pull new catalog records nightly from the ILS. It would submit each record to an AI service with a prompt requesting subject term suggestions and an enhanced description. The enriched records would then go into a review queue for cataloger approval before the final versions are written back to the ILS. The pipeline runs automatically on its schedule, and the cataloger's involvement is limited to the review step rather than the initial processing. The same structure could also handle repository records, chat reference logs that need categorizing, or usage statistics that need pattern analysis.

Data pipelines of this kind are not independent projects for most digital librarians without developer support. They require API access to both the source and destination systems, server infrastructure to run the process, error handling to manage failures without data loss, and monitoring to catch problems before they accumulate. To implement a true pipeline, the practitioner needs developer support, a managed integration platform such as Zapier or Make for simpler cases, or a vendor product that includes pipeline functionality as a supported feature.

Understanding what a pipeline is and what it should do is valuable even if you cannot build one. A digital librarian who can describe a desired pipeline in concrete terms is far better positioned to advocate for it with IT or a vendor than one who can only ask for "better AI integration." Concrete terms means what data should move, from where, to where, on what schedule, and with what transformation in between. That specificity turns a vague request into a scoped project that a developer or vendor can evaluate and implement. The practitioner's role in pipeline development is often less about building than about defining what the pipeline should accomplish, precisely enough for someone else to build it.

Build, configure, or ask the vendor: a decision framework

Every AI integration decision for library systems involves a choice among three paths. You can ask a vendor to provide the capability as a supported feature, configure an existing tool to connect systems without custom code, or build a custom integration with developer support. Each path suits different conditions. The most common error is choosing to build when advocating to the vendor would have worked better.

Asking the vendor is the right first step when the library runs a commercial platform and the use case is general enough that other libraries likely share it. For example, a metadata enhancement feature that thirty libraries would benefit from is something a vendor has a strong incentive to build. One library building it as a custom integration is a poor use of effort compared with a well-articulated feature request submitted through the vendor's formal channel. Such requests work best when they describe the specific workflow, the data involved, and the measurable outcome the library expects. Advocacy through library consortia and professional organizations also amplifies the signal that a feature is broadly needed.

Configuring an existing tool is the right path when a no-code or low-code platform can connect the relevant systems without custom development. For example, Zapier, Make, and Alma's LibOW platform can connect library systems to AI services for a range of workflows without any code. These tools have meaningful limitations: they handle straightforward trigger-action logic more reliably than complex conditional workflows. For the integrations they support, however, they are faster to implement, easier to maintain, and more accessible to practitioners without programming backgrounds than custom development.

Building a custom integration with developer support is appropriate when the use case is specific to the institution's workflow, no vendor product addresses it, and the benefit justifies the ongoing maintenance cost. That decision requires an honest assessment of the maintenance burden. Custom code needs ongoing attention as the systems it connects are updated. A library that builds a custom integration without a maintenance plan is taking on technical debt that may eventually cost more than the integration saves. In a build decision, the practitioner's responsibility is not only to make the case for initial development. It is also to make sure the institution's commitment to maintenance is explicit before work begins.

Vendor AI due diligence: questions worth asking before any

contract is signed

Every path described in this module, whether asking the vendor, configuring an existing tool, or building a custom integration, eventually runs into the same gate. A contract, a license amendment, or a terms-of-service agreement governs what the vendor's AI feature actually does with the library's data. ALA's "Guidance on the Use of Artificial Intelligence in Libraries," adopted at the June 2026 Annual Conference, takes up vendor transparency as one of its central themes. The four areas below (data privacy, algorithmic transparency, environmental cost, and contract terms) are the questions that theme raises for a digital librarian evaluating a specific AI feature. Due diligence is therefore not a separate compliance step that someone else performs after the technical decision. It is part of the same evaluation that determines whether to build, configure, or ask the vendor.

Data privacy is the most immediate due diligence question, and the most concrete. When an AI feature processes patron queries, circulation data, or institutional repository content, the practitioner needs to know whether that data leaves the library's systems at all. If it does, is it retained by the vendor? Is it used to train models that serve other customers? Is it processed by a subcontracted AI provider that the library's contract with the ILS vendor may not name directly? For example, a discovery layer's AI summary feature may call an underlying model from a third-party AI provider. The library's data is then governed by two sets of terms, the ILS vendor's and the AI provider's. A due diligence review that examines only the first has not actually answered the question.

Algorithmic transparency, sometimes called the "black box" problem, is the second area. It is harder to resolve, because vendors often cannot, or will not, fully explain how a proprietary AI feature reaches its outputs. A due diligence review can reasonably ask not for the model's internal workings but for its behavior under specified conditions. On which categories of records or queries does the feature perform poorly? What happens when it is uncertain? Can the feature be disabled or overridden for a specific collection or workflow if its outputs prove unreliable there? A vendor that can answer these behavioral questions, even without disclosing proprietary model details, offers meaningfully more transparency than one that responds only with marketing language about capabilities.

Environmental cost is a newer due diligence category, and one increasingly worth raising directly with a vendor. Training and running large AI models consumes substantial energy, and substantial water for data center cooling. A library's procurement decisions are, in aggregate, part of the demand that shapes how that infrastructure is built and operated. A due diligence review does not need to resolve this question definitively. It is now legitimate, however, to ask a vendor whether it publishes information about the energy sourcing or

efficiency of the infrastructure behind its AI features. The vendor's response, or lack of one, is itself informative.

Contract and licensing questions tie the other three categories together in practical terms. To evaluate a vendor's AI feature responsibly, the digital librarian should confirm several points in the contract. What does it say about data ownership and retention after the contract ends? Can the vendor change the AI feature's behavior or the underlying model without notice? Can the library opt out of an AI feature for specific collections while keeping the rest of the platform? Who bears responsibility if an AI-generated output, such as a suggested subject heading or a generated description, causes a problem after publication? These questions belong with the library's procurement or contracts office, working alongside the digital librarian. The digital librarian, however, is often the only person in that conversation who understands what the AI feature actually does. That makes the practitioner's participation in contract review a genuine professional contribution, not a formality. None of these four categories (privacy, transparency, environmental cost, or contract terms) can be fully resolved by a vendor's answers alone. The final judgment about whether a feature's risk profile is acceptable for a specific library's patrons and collections remains the library's own. The vendor's answers are input to that decision, not a substitute for it.

PRACTITIONER NOTE

The most instructive pattern in library systems integration work is spending weeks building a custom solution before asking the vendor directly. When finally asked, the vendor sometimes says the feature is already on the roadmap for the next release. Other times, a supported connector in an automation platform already handles exactly the workflow in question. That conversation takes fifteen minutes. Asking the vendor before building is not a failure of ambition; it is the operationally correct starting point. If the vendor says yes, the time saved can go to the integrations only the institution can build for itself.

KEY TAKEAWAYS

- An API is a standardized channel through which two software systems exchange data; AI tools increasingly expose APIs that allow library systems to send content for processing and receive results automatically.
- What AI integration is possible depends heavily on which ILS or repository platform the library operates; Alma and open-source platforms offer more accessible API

ecosystems than most commercial systems.

- For institutional repositories, an AI-assisted metadata workflow using batch CSV export and import is accessible today without any API integration and serves as a practical entry point and a proof of concept for more automated approaches.
- A data pipeline automates the movement and transformation of data between systems on a recurring schedule; building one requires developer support, but defining what a pipeline should do with precision is itself a valuable practitioner contribution.
- The three integration paths are vendor advocacy, configuration of existing tools, and custom development; asking the vendor first is the correct default, and building is appropriate only when the use case is institution-specific and the maintenance commitment is explicit.
- The practitioner's most important contribution to AI integration work is often not building but defining: describing what data should move, between which systems, with what transformation, precisely enough that a developer or vendor can evaluate and implement it.
- ALA's Guidance on the Use of Artificial Intelligence in Libraries (adopted June 2026) frames vendor transparency as a baseline expectation; a vendor AI due diligence review should cover data privacy, algorithmic transparency, environmental cost, and contract terms, with the digital librarian's technical understanding of the feature itself as essential input to that review.

References

APA 7TH EDITION

1. Ex Libris. (2025). *Alma library services platform*. Clarivate. <https://exlibrisgroup.com/products/alma-library-services-platform/>
2. Ex Libris. (2025). *Library Open Workflows (LibOW): Automated library workflows for Alma*. Clarivate. <https://exlibrisgroup.com/library-open-workflows/>

ACRL AI Competencies covered

Use & Application

Analysis & Evaluation

