Move the Workflow, Not Just the Content

A practical guide to moving Confluence workflows from Data Center to Cloud

Aura Apps

JULY 2026

Executive Summary

Moving Confluence from Data Center (DC) to Cloud involves more than transferring pages, attachments, and users. For teams that rely on workflow and approval apps, it also means moving the processes built around that content.

That difference matters. A migration can appear successful while an important review process no longer works as expected. Reviewer assignments may behave differently. Publishing controls may have changed. A familiar app may offer different features in Cloud, require its own migration process, or no longer make sense for the organization.

Atlassian recommends reviewing Marketplace apps early in the Cloud journey and provides migration tools that help teams see which apps are installed, whether Cloud versions are available, how features differ, and what migration options are available. That information is useful, but it does not make the decision. Each organization still needs to decide what must be preserved, what can be simplified, and how it will know the new approach works.[1]

A practical workflow migration follows five stages:

1. **Discover** the workflows, safeguards, owners, and records the organization actually relies on.
2. **Classify** the best Cloud path for each one: retain, replace, consolidate, use native functionality, redesign, or retire.
3. **Compare** realistic options based on how well they support the work, the people involved, the migration, and the budget.
4. **Prove** the choice with representative content, real users, and the situations most likely to expose a problem.
5. **Transition** through a confirmed migration path, with a plan for checking results, fixing exceptions, training users, and supporting the new setup.

Native Approvals in Confluence, introduced in open beta for Premium and Enterprise customers in July 2026, gives teams a genuine option for page-level sign-off.[2] For straightforward approvals, it may be enough. More involved processes may still require custom workflow steps, specialized reviewer rules, additional actions, cross-space publishing, support on another Confluence plan, or a proven replacement path from an existing workflow app.

Aura Workflow is one option for those more involved needs. The useful question is not whether Aura has more features. It is whether Aura fits the workflows the organization needs to preserve and provides a migration path the team can confidently follow.

The pages may be what moves to Cloud, but the workflows are what help people trust the content once it arrives.

1. The migration inside the migration

Atlassian has set a clear deadline for most DC customers. Affected Data Center products and related Marketplace app licenses are scheduled to reach end of life on March 28, 2029. After that date, they become read-only.[3]

March 2029 may still sound distant, but complex Confluence environments need time for discovery, vendor coordination, trial migrations, remediation, and user preparation. For teams that have not started, the practical time to begin planning is now.

Teams need to understand the current site, clean up what they no longer need, choose a migration method, move the content, check the results, and prepare users for launch.

All of that is necessary, but it is not the whole story.

A well-established Confluence environment is more than its pages. Over time, teams add Marketplace apps that turn Confluence into a publishing platform, policy library, controlled document system, or operational workspace. Those apps often contain important business rules that are easy to overlook because they sit so close to the content.

A policy may require Legal approval before publication. Editing a procedure may send it back for review. Approved content may expire after twelve months. Authors may need to work on a new draft while readers continue to see the last approved version. The organization may also need a record of who approved a document, when they approved it, what they reviewed, and why something was rejected.

These are more than app features. They are safeguards that help the organization work consistently and demonstrate that the right decisions took place.

The Confluence Cloud Migration Assistant can help administrators see which Marketplace apps are installed, whether Cloud versions exist, how features differ, and which migration paths are available.[1] App data moves only when the vendor supports a migration path, and both built-in and Marketplace macros may behave differently after the move.[4]

Even when a technical path exists, one larger question remains: will the new Cloud setup still support the way the organization works?

That is the migration inside the migration. Spaces, pages, attachments, users, and groups make up the visible move. Underneath is a second project involving workflow steps, reviewers, permissions, notifications, integrations, publishing, and approval records.

Workflow tools should not be treated as cargo that simply follows Confluence to Cloud. Important processes need an owner, clear requirements, a chosen destination, and a test plan of their own.

The first step is not choosing a replacement app. It is understanding what the current workflow does and deciding which parts are still worth carrying forward.

2. Discover: preserve outcomes, not complexity

An app inventory tells you what is installed. It does not tell you what the organization needs.

In a Data Center environment that has evolved over time, different teams may use the same app in very different ways. One configuration may contain workflow steps nobody understands, while a seemingly simple approval may be the only documented safeguard supporting an audit.

Before comparing Cloud products, look at both the technology and the business process.

Inventory the technology

Start with the Confluence Cloud Migration Assistant or the relevant administrative tools. Record whether each workflow or approval app is active, where it appears, which spaces and teams depend on it, and whether a suitable Cloud version exists. Note important feature differences, the available migration method, current licensing costs, the expected Cloud user tier, and any integrations or customizations that also need attention.

Atlassian recommends considering app usage, cost, importance, Cloud availability, feature differences, security, and migration options during this assessment.[5] That creates a useful technical map. The next step is to add the human and business context.

Trace the business process

Talk with the people who own the process and the people who use it. Find out what starts the workflow, what it is meant to protect, and who authors, reviews, approves, publishes, or administers the content. Ask what happens when someone rejects a request, misses a deadline, leaves the company, or edits content that was already approved.

It is also important to understand what readers should see during a review, what proof must remain afterward, when approved content becomes outdated, and what could happen if the workflow fails. These conversations often reveal that the current configuration and the real business need have drifted apart.

Separate the requirement from the current setup

Suppose a Data Center workflow includes a step called “Legal Gate 2,” uses a particular macro, and sends every approval to a fixed reviewer group.

The underlying need may be much simpler:

Legal must approve important policy changes before broad publication, and the organization must keep a record of that decision.

That requirement should survive the migration. The step name, macro, and reviewer setup should survive only if they are still the best way to meet it.

This distinction helps avoid two common mistakes. The first is losing an important safeguard because nobody realized what an awkward configuration was doing. The second is rebuilding every old step and workaround simply because it already exists.

Cloud migration is a good opportunity to remove duplicate workflows, retire unused processes, standardize common patterns, and document requirements that previously lived only in someone’s head.

The goal is not to reproduce DC perfectly. It is to preserve what the organization needs in a form it can manage confidently in Cloud.

3. Classify: choose the right Cloud path

Once the required outcome is clear, each workflow needs a destination. There are six reasonable options.

1. Retain

Move to the current app’s Cloud version when it meets the requirement, satisfies the organization’s security and governance needs, and offers an acceptable migration path.

Keeping the same app may reduce disruption, but it should not be automatic. A Cloud edition can differ from its DC counterpart in features, administration, architecture, or price. Test the behavior that matters instead of relying on the shared product name.

2. Replace

Choose another Marketplace app when the existing solution does not offer a suitable Cloud version, lacks an important capability, adds too much cost or complexity, or no longer fits the organization’s plans.

3. Consolidate

Move several capabilities into a smaller group of apps when one product or product family can meet the requirements without giving up necessary depth.

Consolidation may make vendor management, security reviews, training, and administration easier. It still needs to be tested one capability at a time. Fewer apps help only when the remaining apps do the required work.

4. Use native functionality where it fits

Use Confluence's own capabilities when they meet the need.

This option became more relevant in July 2026, when Atlassian began rolling out Approvals in Confluence as an open beta for Premium and Enterprise customers. The current documentation includes formal approval requests on pages, notifications, visible progress, decision history, required approvers, and a space-level option to require approval before publishing.[2]

The beta announcement also describes automation triggers and actions, along with expiration windows that can prompt teams to renew approvals. Beta features can change, so teams should check what is available in their own site and review the latest documentation before making a production decision.

For straightforward page approvals, Atlassian's native capability may be enough. Teams with more complex requirements should compare it with app-based workflows before deciding which approach fits best.

5. Redesign

Use the migration to simplify or improve the process. A five-step approval process shaped by DC limitations may become a clearer three-step process in Cloud. A fixed approver may become a role or variable. Several departmental workflows may become one shared template.

Simplifying a workflow does not mean removing the safeguards that matter. Any changes should be agreed with the process owner and tested against the same requirements.

6. Retire

Remove workflows that are unused, duplicated, outdated, or no longer worth the effort required to maintain them.

Retirement may be the least glamorous choice, but it can be one of the most valuable. It reduces the amount of data, configuration, training, and support that must make the move.

These strategies can coexist. Within a single organization, that might mean retaining a complex policy workflow, replacing a departmental app, using native Approvals for straightforward sign-off, and retiring unused templates. The destination for each workflow should be explicit. An undocumented assumption is not a migration plan.

4. Compare: look beyond the feature list

After classifying the possible paths, teams can compare realistic candidates. This is where a simple feature checklist can become misleading.

Two products may both claim to provide approvals, notifications, and history. But those labels may not tell the whole story. The more useful questions are practical ones. Who can review the content? Can the workflow require several people to approve? What happens if a reviewer leaves during an active request? Does rejection require an explanation? Does editing a page cancel its previous approval? Can readers continue to see the last approved version while authors revise the next one? What proof of the decision remains afterward?

A helpful comparison looks at four areas.

Workflow fit

Start with the situations identified during discovery. Check whether each option supports the required workflow steps, reviewer choices, approval rules, rejections, revisions, reminders, recurring reviews, permissions, publishing, history, and automated follow-up actions.

Do not stop at asking whether a feature exists. Ask what happens when a review is late, rejected, edited, reassigned, or only partly completed. Those are often the moments when differences between products become clear.

Platform and governance fit

Consider how the app fits into the wider Cloud environment. Review its hosting and data flows, security and privacy information, reliability, compliance support, data residency, retention options, permissions, administration, integrations, documentation, vendor support, and fit with the organization's Confluence plan.

Atlassian sets requirements for Cloud apps and provides information to help customers evaluate them. Even so, each organization must decide whether an app meets its own security, legal, regulatory, and operational needs.[1]

Migration fit

A product can meet the functional requirement and still be a poor migration choice if the path to it is unclear.

Determine whether the move is automated, manual, supported by a vendor, or split across several stages. Confirm exactly what comes across, including templates, pages, workflow steps, history, metadata, reviewers, and attachments. Ask how users and groups are matched, whether active approvals can move, how exceptions are reported and repaired, what recovery choices exist, and how much lead time the vendor needs.

Marketplace Partners have an important role because app data can move only through the migration paths they provide. Atlassian recommends contacting vendors early and sharing the planned schedule, scale, security concerns, and support needs.[5]

Human and economic fit

Finally, consider the experience of the people who will live with the decision. Authors should understand what to do next. Reviewers should be able to respond without specialist training. Readers should be able to tell whether content is a draft, approved, or outdated. Administrators should be able to maintain workflows and diagnose problems without creating unnecessary risk.

When comparing costs, make sure you are comparing like with like: the same user tier, contract length, features, and level of support. Look beyond the license price and include migration services, manual cleanup, training, and ongoing support. A cheaper app may cost more once the conversion work is included, while a more expensive app may save money if it replaces several others. Run the numbers rather than assuming either outcome.

The goal is not to collect the largest number of features. It is to find the simplest solution that reliably supports the work the organization needs to do.

The comparison produces a promising answer. The pilot shows whether that answer holds up.

5. Prove: test the workflow, not just the data transfer

Atlassian strongly recommends a trial migration, data checks, and user acceptance testing before production.[6] Workflow migrations need the same care because the most serious problems are often about behavior, not page counts.

A page count can match perfectly while the process around those pages no longer works.

Build the pilot around representative content and the people who actually use it. Test the normal approval path, but also include rejection, revision, resubmission, several reviewers, missing reviewers, notifications, publishing controls, content updates, approval history, and at least one failed or incomplete migration item. If the process uses expiring approvals, recurring reviews, group reviewers, or draft restrictions, test those too.

Include process owners and end users, not only administrators. An administrator can confirm that a page moved to the next step. A policy owner can confirm that the move represents the right business decision. A reviewer can reveal that the new experience is confusing. An audit or governance stakeholder can determine whether the recorded evidence is sufficient.

Write down the expected result before each test. Then record what actually happened, who owns any follow-up, what evidence supports the result, and whether the scenario passed, failed, or exposed a limitation the organization is willing to accept.

Check more than totals. Where the source and target allow it, compare workflow templates, pages in scope, current steps, reviewer assignments, history, metadata, permissions, published copies, and known exceptions.

Testing does not remove all migration risk. It turns unknowns into observed behavior, documented limitations, and decisions with clear owners.

The pilot should end with an explicit choice: approve the target, approve it after specific fixes, revise the design, or choose another path. “The demo looked good” is not an acceptance decision.

6. Transition: make the migration path clear

Once a target passes the pilot, turn the decision into a practical transition plan.

Start by confirming the current path with both the source and target vendors. Do not assume that a Cloud replacement can receive data directly from a DC app. Product compatibility and migration compatibility are different questions.

Aura’s migration path from Comala Document Management provides a useful example of a direct, supported route. Aura supports migrations from both Comala Data Center and Comala Cloud through a script-based process:

1. Recreate the required workflows in Aura, matching the source workflow and state names where possible.

2. Run Aura's migration script to extract the active Comala workflow assignments and current page states.
3. Review the generated mapping and identify the corresponding Aura workflow name and scope.
4. Upload the migration file to Aura and monitor which pages were successfully applied or could not be migrated.

When a Comala state matches a state in the target Aura workflow, the page retains its current position in the lifecycle. If no matching state exists, the page enters the target workflow's initial state. The route is direct, but it still requires workflow preparation, mapping, testing, and review of exceptions.[7]

AppFox illustrates a different pattern for source apps without a direct Data Center-to-Aura path. For supported AppFox workflows, the replacement may happen in two stages:

1. Move from AppFox Data Center to AppFox Cloud using Atlassian's migration tools.
2. Convert from AppFox Cloud to Aura Workflow through Aura's supported Cloud conversion process.[8]

These examples show why teams must confirm the complete route for their specific source app. A migration may be direct or may include intermediate stages, and each stage needs its own scope, validation, and recovery plan.

A useful transition plan covers ten points:

1. **Scope**: Identify the templates, pages, spaces, users, groups, history, and metadata included.
2. **Requirements**: Confirm versions, permissions, licenses, target settings, and vendor availability.
3. **Sequence**: Put every migration and conversion stage in the right order.
4. **Protection**: Prepare appropriate backups and decide when recovery or rollback would be necessary.
5. **Practice**: Run a representative test and update the instructions based on timing and problems observed.
6. **Checks**: Decide how the team will compare the source and target results.
7. **Fixes**: Assign owners, priorities, and deadlines for exceptions.
8. **Change control**: Limit or track changes in the source during the final migration window.

9. **Preparation**: Train administrators, authors, reviewers, readers, and support teams for the new process.
10. **Follow-up**: Monitor failures, incomplete items, support requests, and user frustration after launch.

Pay particular attention to people and groups. Workflow templates often refer to specific identities. In Aura Workflow, for example, an exported template contains the workflow design, but it does not include a page's current workflow step, pending approvals, history, or page-level metadata. Fixed account and group identifiers may also need to be selected again when a template moves between sites.[9]

The transition plan needs to account for those limits. A useful playbook makes clear who takes each action, what success looks like, and what to do when reality differs from the plan.

7. Where Aura Workflow fits

Native Approvals can change the starting question for Cloud customers.

The question is no longer simply, "How can we add approval to Confluence?" It is, "How much workflow control does this process need?"

For Premium or Enterprise customers that need straightforward page sign-off, notifications, visible status, decision history, required approvers, and a publishing gate, native Approvals may be enough. Teams should test it alongside their actual requirements.

Aura Workflow becomes relevant when a process goes beyond that baseline or needs a different way of working.

Configurable content lifecycles

Aura Workflow allows teams to design workflow steps and transitions around their process instead of relying on one standard approval cycle. A page might move from Draft to Subject-Matter Review, Legal Review, Approved, Published, and eventually Expired. A transition can require approval, allow a manual choice, begin after an edit, or happen after a set period.[10]

This can help when governance requires more than a simple approved or not-approved result.

Flexible approval rules

Reviewers can be users, groups, or workflow variables. Teams can set minimum approval and rejection thresholds, require everyone to agree, require a comment with a rejection, send

reminders, and optionally require a two-factor authentication check before recording a decision.[11]

Variables allow one shared template to find the right role or group for an individual page. That reduces the need to place every person directly into the workflow design.

Actions and useful context

Workflow steps can trigger actions such as sending email, changing restrictions, applying labels, updating a title, creating an official version, or publishing a page. Workflow metadata can also store structured information connected with the content.[10]

These features can connect an approval decision to what needs to happen next, reducing manual follow-up for authors and administrators.

Working and published content

When readers should see only approved content, Aura documents a two-space publishing approach. Authors draft and review material in a restricted working space. Once the workflow reaches the approved step, an action publishes that content to a reader-facing space. Authors can then revise the working copy while readers continue to see the last approved version.[12]

This approach may suit policies, controlled procedures, customer documentation, and similar content where work in progress should not replace the trusted version.

Broad application

A workflow can apply to one page, an entire space, or pages with a particular label.[10] This allows teams to manage a category of content consistently while using variables and metadata to adjust the shared template for individual pages.

Plan and migration considerations

Aura Workflow is available across Confluence plans, including Free and Standard, while native Approvals is currently documented for Premium and Enterprise.[2][13] That difference may matter, especially for teams on lower-tier plans, but availability alone should not determine the choice. Compare the full cost and capabilities of each option at your actual user tier.

The migration path also needs to be confirmed for the specific source app. Aura offers supported conversion paths for certain workflow products, but a route may include several stages and may not carry over every active item or historical detail. Confirm the scope, exceptions, identity matching, and available support before committing.

Aura belongs on the shortlist when a team needs a custom multi-step lifecycle, more specialized reviewer rules, approval thresholds, two-factor authentication for decisions, automated follow-up actions, structured metadata, cross-space publishing, broad governance across a space or label, support on a Confluence plan without native Approvals, or a supported path away from an existing app.

It may not be the right fit when native functionality already meets the need, an essential integration is unsupported, the migration route remains unclear, or the work required to manage the solution outweighs its value.

The best next step is therefore not a generic product demonstration. It is an assessment built around one representative workflow. Bring the current workflow, the outcome it protects, the organization's Confluence plan, and the scenarios the new solution must pass. Then compare native Approvals, Aura Workflow, the existing app's Cloud version, and any other realistic option.

Move the control, not just the content

Cloud migration creates both a deadline and an opportunity to decide which safeguards, decisions, and records must remain dependable in Cloud, and what the simplest sustainable approach looks like.

The sequence is straightforward. Discover what the organization relies on. Classify the possible Cloud paths. Compare the realistic choices. Prove the preferred option with real scenarios. Transition through a confirmed plan with clear owners.

Apply that method equally to the existing app's Cloud version, native Confluence Approvals, Aura Workflow, and any other candidate. The purpose is not to justify a product that has already been chosen. It is to preserve trust in the processes connected to the content.

Start with one workflow that matters. Name its owner. Write down the outcome, reviewers, safeguards, evidence, and consequences of failure. Then use it to test both the proposed solution and the migration method.

The pages may be what moves to Cloud, but the workflows are what help people trust the content once it arrives.

Workflow migration readiness checklist

Discover

- [ ] We have an inventory of workflow and approval apps.
- [ ] We know which spaces, pages, macros, and teams depend on them.
- [ ] Every important workflow has an owner.
- [ ] We have documented the required outcome separately from the current setup.
- [ ] We understand usage, importance, cost, integrations, and record-keeping needs.

Classify

- [ ] Every workflow has a proposed path: retain, replace, consolidate, native, redesign, or retire.
- [ ] We have checked whether native Confluence Approvals meets the need.
- [ ] Process owners have approved any redesigns or retirements.
- [ ] Every important workflow has a documented destination.

Compare

- [ ] We have compared how each option supports the workflow, platform, migration, people, and budget.
- [ ] We have reviewed security, privacy, compliance, hosting, and data needs.
- [ ] Cost comparisons use the actual Cloud plan, user tier, migration effort, and support model.
- [ ] We have separated vendor statements from behavior we tested ourselves.

Prove

- [ ] We completed a representative pilot with real content and roles.
- [ ] We tested approval, rejection, revision, permissions, publishing, history, and exceptions.
- [ ] We recorded both expected and actual results.
- [ ] Administrators, process owners, and end users took part in acceptance testing.
- [ ] Every known limitation has an owner and an agreed response.

Transition

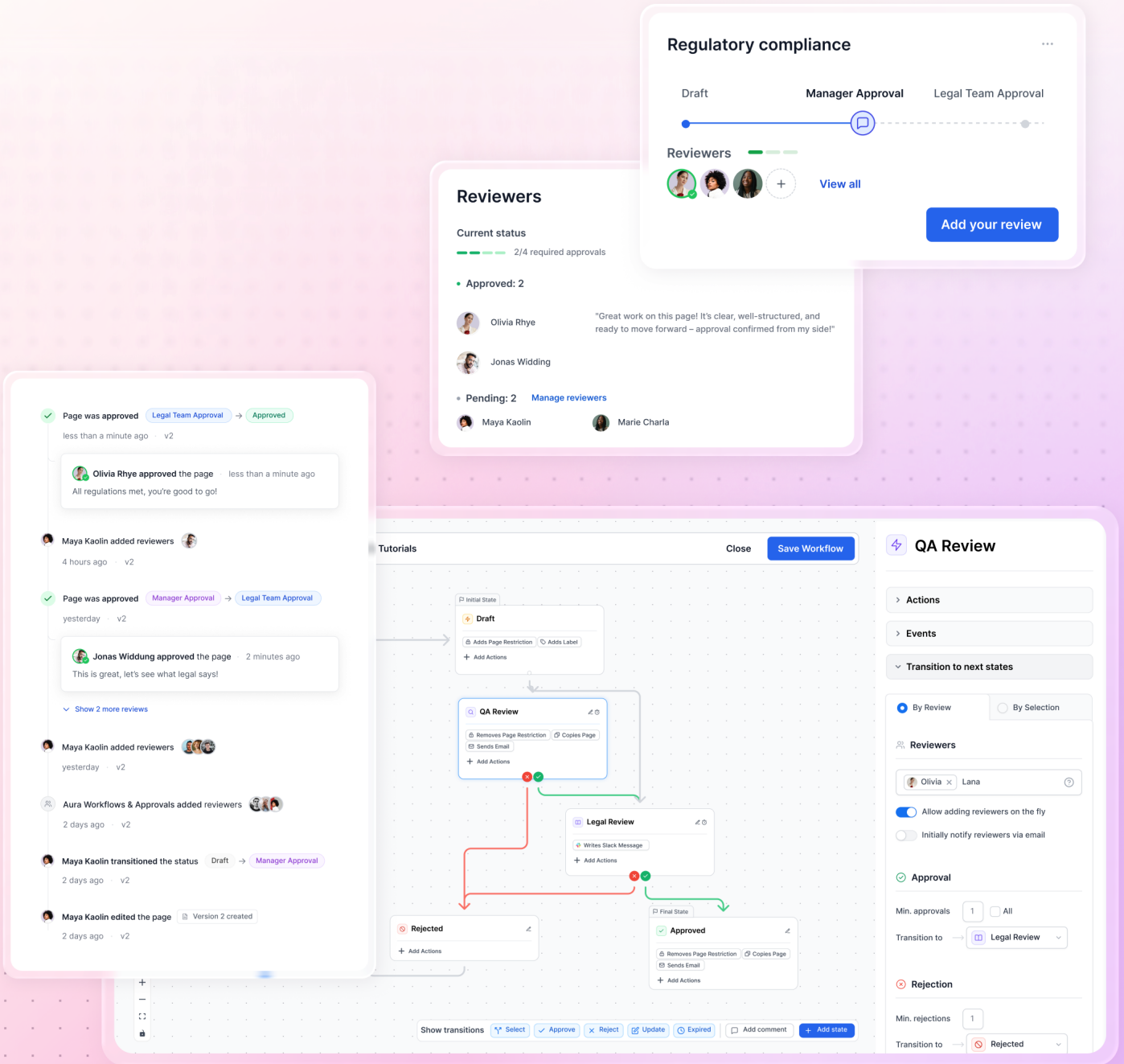
- [ ] Source and target vendors confirmed the migration stages and supported scope.
- [ ] The plan includes every intermediate step.
- [ ] Backup, result checking, repairs, and recovery rules are documented.
- [ ] We tested how users and groups will be matched.
- [ ] Training and post-launch support have owners.
- [ ] Monitoring covers failures, incomplete items, manual fixes, and user frustration.

Start with one workflow that matters

You do not have to rebuild every process at once. Tell us which approvals have to survive the move, and we will show you how they map to Aura Workflow.

Plan your migration

Try Aura Workflow



References

[1] Atlassian, "Cloud migration app assessment," accessed July 13, 2026. https://www.atlassian.com/migration/assess/app-assessment

[2] Atlassian, "Request and manage approvals in Confluence," accessed July 13, 2026. https://support.atlassian.com/confluence-cloud/docs/request-and-manage-approvals-in-confluence/ The open beta date and rollout description come from the Atlassian Team's July 10, 2026 Community announcement, "Approvals in Confluence, now on Open Beta."

[3] Atlassian, "Data Center End of Life," accessed July 13, 2026. https://www.atlassian.com/licensing/data-center-end-of-life

[4] Atlassian, "What migrates with the Confluence Cloud Migration Assistant," accessed July 13, 2026. https://support.atlassian.com/migration/docs/what-migrates-with-the-confluence-cloud-migration-assistant/

[5] Atlassian, "Audit Marketplace apps and decide on the strategy," accessed July 13, 2026. https://support.atlassian.com/migration/docs/audit-apps-for-your-migration-to-cloud/

[6] Atlassian, "Test your migration to Cloud," accessed July 13, 2026. https://support.atlassian.com/migration/docs/test-your-server-to-cloud-migration/

[7] Aura Apps, "Migrating from Comala to Aura Workflows," accessed July 13, 2026. https://docs.aura-workflows.app/admin-guide/migration/

[8] Aura Apps, "How to migrate AppFox workflows to Aura: Confluence approval workflow guide," March 27, 2026. https://aura-apps.io/blog/appfox-workflows-to-aura-workflow-migration-guide/

[9] Aura Apps, "Import and export," Aura Workflows documentation, accessed July 13, 2026. https://docs.aura-workflows.app/admin-guide/import-export/

[10] Aura Apps, "Overview," Aura Workflows documentation, accessed July 13, 2026. https://docs.aura-workflows.app/getting-started/overview/

[11] Aura Apps, "Approvals," Aura Workflows documentation, accessed July 13, 2026. https://docs.aura-workflows.app/admin-guide/approvals/

[12] Aura Apps, "Document lifecycle and publishing," Aura Workflows documentation, accessed July 13, 2026. https://docs.aura-workflows.app/using-aura/document-lifecycle/

[13] Aura Apps, product website, accessed July 13, 2026. https://aura-apps.io/