Workflow

Complete PDF Manual

Guide for Magento administrators, technical project owners, and implementation teams.

LAST UPDATED
2026-08-25

SOURCE
Open complete documentation online

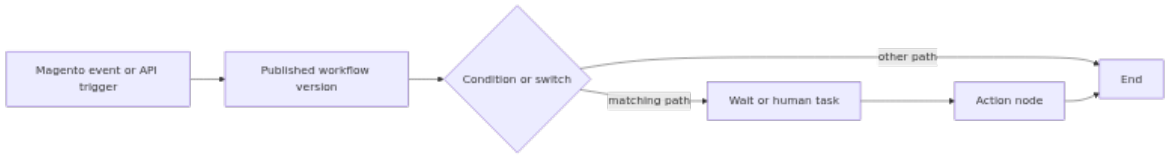
Table of Contents

Workflow · workflow

Workflow

MageB2B Workflow runs repeatable Magento processes from a visual definition. An event, schedule, API call or webhook starts a run. Conditions and switch nodes choose a path; action nodes update the run payload or perform a controlled side effect; waits and approval tasks pause the run until it can continue.

The workflow definition is not the run itself. Editing a draft changes the next published version, while an existing run keeps the version with which it started.



Start here

1. Install the module
2. Review the global settings
3. Build a first approval workflow

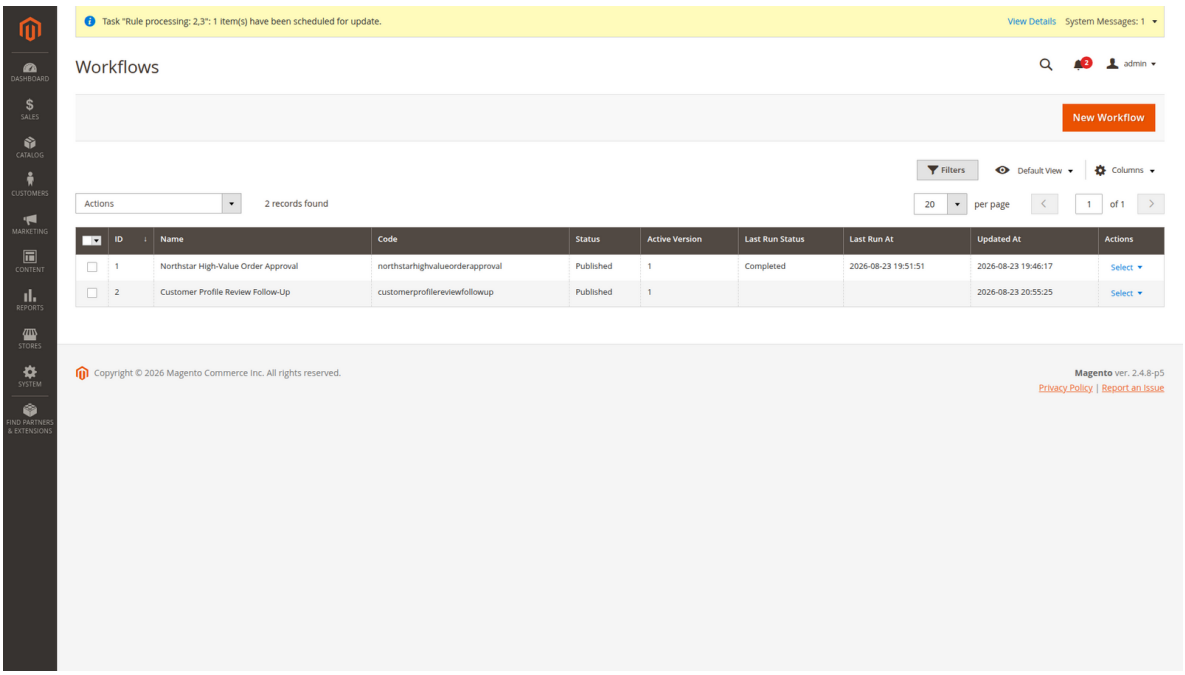
Build and route workflows

- Workflow concepts
- Builder and connections
- Triggers and event payloads
- Conditions and switch routing
- Action nodes
- Human tasks and decisions

Release and operate workflows

- Validation and simulation
- Publishing and versions
- Import and export
- Runs and tasks
- Queues and cron
- API integration
- Permissions and security

Workflow definitions are under **Content > Workflow > Workflow Definitions**. Human decisions are under **Content > Workflow > Workflow Tasks**.


Task "Rule processing: 2,3": 1 item(s) have been scheduled for update.
View Details
System Messages: 1
Workflows
admin
New Workflow
Filters
Default View
Columns
Actions
2 records found
20
per page
1
of 1
ID
Name
Code
Status
Active Version
Last Run Status
Last Run At
Updated At
Actions
1
Northstar High-Value Order Approval
northstarhighvalueorderapproval
Published
1
Completed
2026-08-23 19:51:51
2026-08-23 19:46:17
Select
2
Customer Profile Review Follow-Up
customerprofilereviewfollowup
Published
1
2026-08-23 20:55:25
Select
Copyright © 2026 Magento Commerce Inc. All rights reserved.
Magento ver. 2.4.8-p5
Privacy Policy | Report an Issue

Installation

Install the package through the same authenticated SoftwareSilo Composer repository used for your other modules. Composer installs the required Extension Manager package with it.

```
composer require mageb2b/workflow
php bin/magento module:enable MageB2B_Workflow
php bin/magento setup:upgrade
php bin/magento cache:flush
```

In production mode, finish with your normal dependency-injection compilation and static-content deployment process.

```
php bin/magento setup:di:compile
php bin/magento setup:static-content:deploy
```

Verify the installation

1. Sign in to Magento Admin.
2. Open **Content > Workflow > Workflow Definitions**.
3. Confirm that **New Workflow** is available to a role with workflow management permission.
4. Open **Stores > Configuration > General > Workflow Engine** at **Default Config** scope.
5. Confirm that **Enable Workflow Engine** is set to **Yes**.

The module stores definitions, immutable versions, runs and tasks in Magento. Take a database backup before installing it on an existing store, and deploy the code to every application, cron and queue-worker node before running `setup:upgrade`.

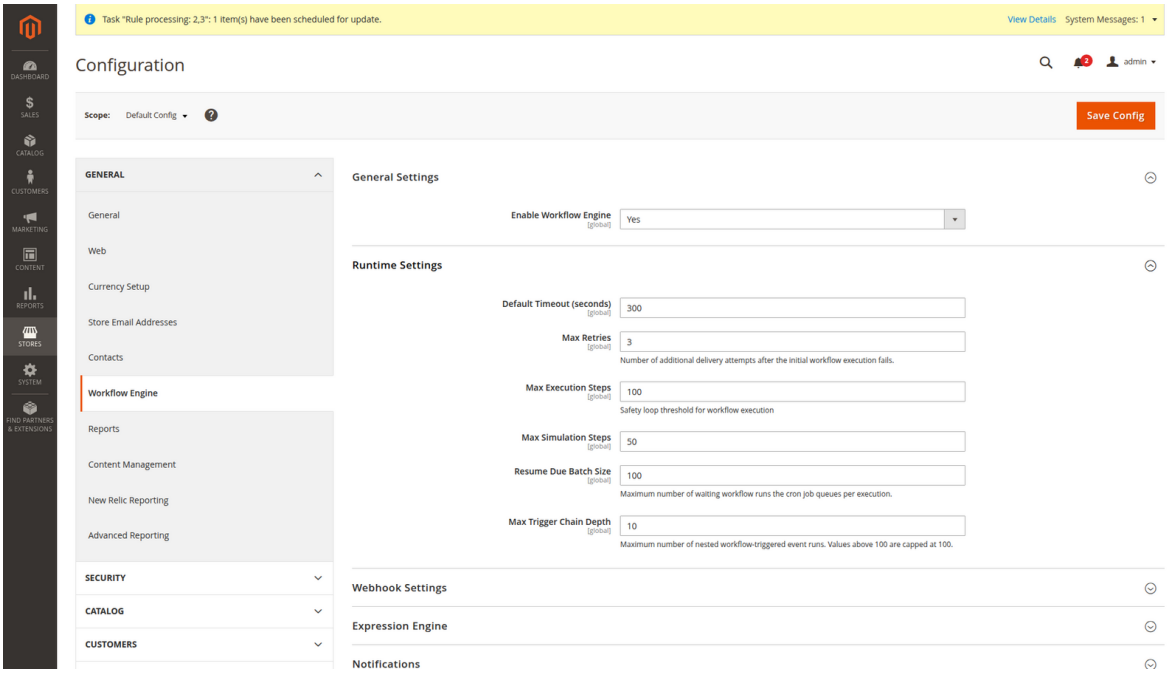
Runtime prerequisites

The engine dispatches event triggers and run execution through Magento message queues. A successful Admin installation does not prove that live workflows will run. Configure both consumers described in Queues and cron, and make sure Magento cron runs every minute.

Do not publish a business-critical definition until validation, simulation and a controlled end-to-end run have all passed.

Configuration

Open **Stores > Configuration > General > Workflow Engine**. These settings are global; the module does not expose website- or store-view-specific values.



General and runtime settings

Enable Workflow Engine is the master runtime switch. Disabling it prevents workflows from being started. It does not delete definitions, versions, runs or tasks.

Default Timeout is the fallback request timeout used by webhook execution. **Max Retries** controls additional run-consumer attempts after a failed execution. **Max Execution Steps** stops an accidental loop in a live run, while **Max Simulation Steps** applies the same kind of limit to previews. **Resume Due Batch Size** limits how many waiting runs the cron job queues at once. **Max Trigger Chain Depth** prevents workflows from starting an unbounded chain of further workflow-triggering events.

Keep the defaults for a first rollout. Raise a limit only after checking the run history and worker capacity; a larger number can turn a malformed workflow into more queue work rather than fixing it.

Webhook settings

The request and response byte limits bound outbound webhook traffic. The method list restricts which HTTP verbs a webhook action can use. The allowed-host list is the important security control: an empty list denies every host. Add only the public hosts that workflows genuinely need. `*` permits every public host and should not be a routine production setting.

Inbound signed triggers use the signature lifetime, clock-skew allowance and nonce

lifetime. A short clock-skew allowance is safer, but all participating systems need synchronized clocks.

Expressions, notifications and condition models

Enable Expressions permits expression-based conditions. Turning it off affects definitions that depend on expression evaluation.

Enable Task Notifications controls notifications for human tasks. Notification delivery still depends on the store's email and queue setup.

Flat condition models expose supported Magento data fields in the builder. The default prefixes cover catalog, sales, customer, quote and EAV tables. Restricting the list removes fields from the selectable model surface; widening it exposes more database-backed fields to workflow authors.

Import and export

Imported packages are limited by **Max Workflow Import Size**. **Allow Overwrite on Import** is off by default. Leave it off unless replacing an existing definition code is an intentional deployment step and the incoming package has been reviewed.

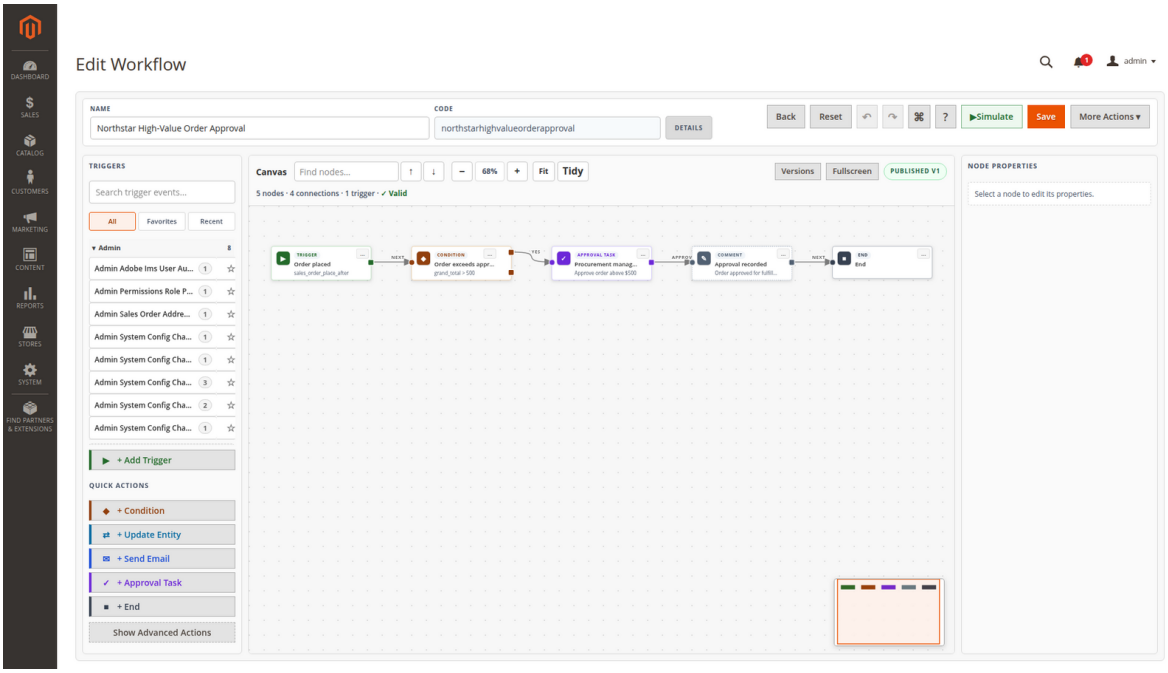
First workflow

This example sends orders above a chosen total to a procurement manager before fulfillment continues. Build it in a staging store first, using a product and customer created for testing.

Create the definition

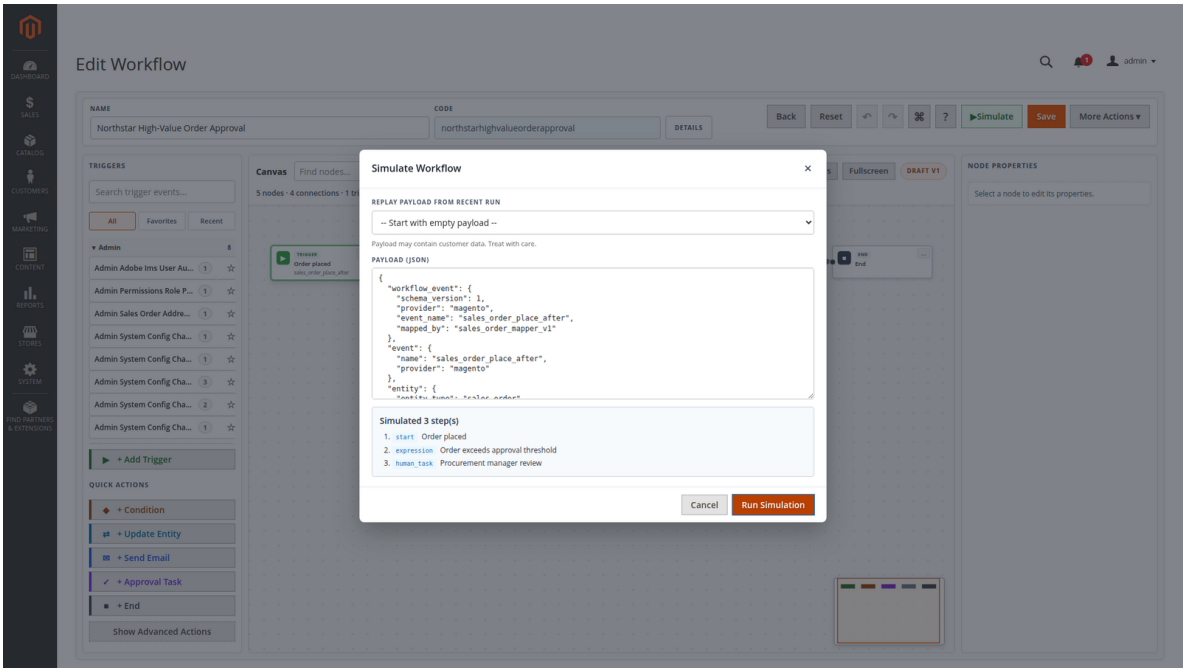
1. Open **Content > Workflow > Workflow Definitions**.
2. Click **New Workflow**.
3. Enter a clear name and a stable code. The code cannot be changed after the definition is created.
4. Add the **Sales Order Place After** trigger.
5. Add a condition named `Order exceeds approval threshold`.
6. Set the rule to compare the event entity's `grand_total` with the threshold that matches the test order.
7. Add an approval task named `Procurement manager review` and keep only the decisions your process supports.
8. Add a Comment action that records the approval in the workflow audit trail, then add an End node.

Connect the trigger's **NEXT** port to the condition. Connect the condition's matching branch to the approval task, the task's decision port to the follow-up action, and that action to End. A condition branch without a connection stops at the condition; use an explicit End node when the canvas should name that outcome.



Validate and simulate

The builder shows **Valid** only after the graph, node settings and connections pass validation. Click **Simulate** and provide an event payload whose order total exceeds the threshold. Simulation should reach the approval task and stop there. It does not create a real task or send side effects.



Publish and run the real path

1. Save the definition.
2. Use **More Actions > Publish** and confirm the prompt.
3. Place a controlled Magento order above the threshold.
4. Wait for the event-trigger and run-start consumers to process their queues.
5. Open **Content > Workflow > Workflow Tasks**.
6. Review the order context, add a useful decision comment and approve the task.
7. Confirm that the task closes and the workflow's latest run becomes **Completed** after the resumed run is processed.

Publishing makes the version executable. Saving a draft alone does not.

Workflow concepts

A workflow has four layers that are easy to confuse during troubleshooting:

- A **definition** is the named, editable container identified by its code.
- A **version** is a saved graph of nodes and connections. Publishing makes one version active.
- A **run** is one execution of one published version with its own payload, status and current node.
- A **task** is a manual decision created when a run reaches a Human Task node.

Changing a draft does not rewrite an earlier run. This separation lets an operator inspect what actually ran even after the workflow has been edited.

Nodes and connections

A trigger starts the graph. Conditions and switches choose a labelled output. Actions perform work or update the run payload. Wait and Human Task nodes pause execution. An End node completes that route.

Connections carry control between nodes. Labels matter: a condition uses **YES** and **NO**, an approval task uses its enabled decisions, and a normal action uses **NEXT**. A node shown on the canvas is not part of the executable path until it is connected.

Payload and variables

Every run carries a JSON payload. Magento event triggers add normalized `workflow_event`, `event`, `entity` and `data` sections. The exact entity fields depend on the event mapper. A Set Variable node writes a workflow value into the current run payload; it does not by itself change a Magento order, customer or product.

Entity updates, comments, email and webhooks are separate action nodes because they have effects outside the in-memory workflow state.

Synchronous design, asynchronous execution

The graph reads left to right, but live event execution is queued. The event consumer matches a published definition and creates work; the run consumer advances the workflow. A wait is resumed by cron, and a human task resumes after a decision. For that reason, a successful save or publish is not evidence that the queues are healthy.

Builder and connections

The builder has three working areas: triggers and quick actions on the left, the graph canvas in the middle, and the selected node's properties on the right.

Add and configure a node

1. Add a trigger from the event catalogue or use **Add Trigger** for another supported trigger type.
2. Add actions from **Quick Actions**. Expand advanced actions for Set Variable, Webhook, Comment and Wait.
3. Select a node on the canvas.
4. Complete its properties on the right.
5. Click outside the field or move focus before saving so the last edit is committed by the browser control.

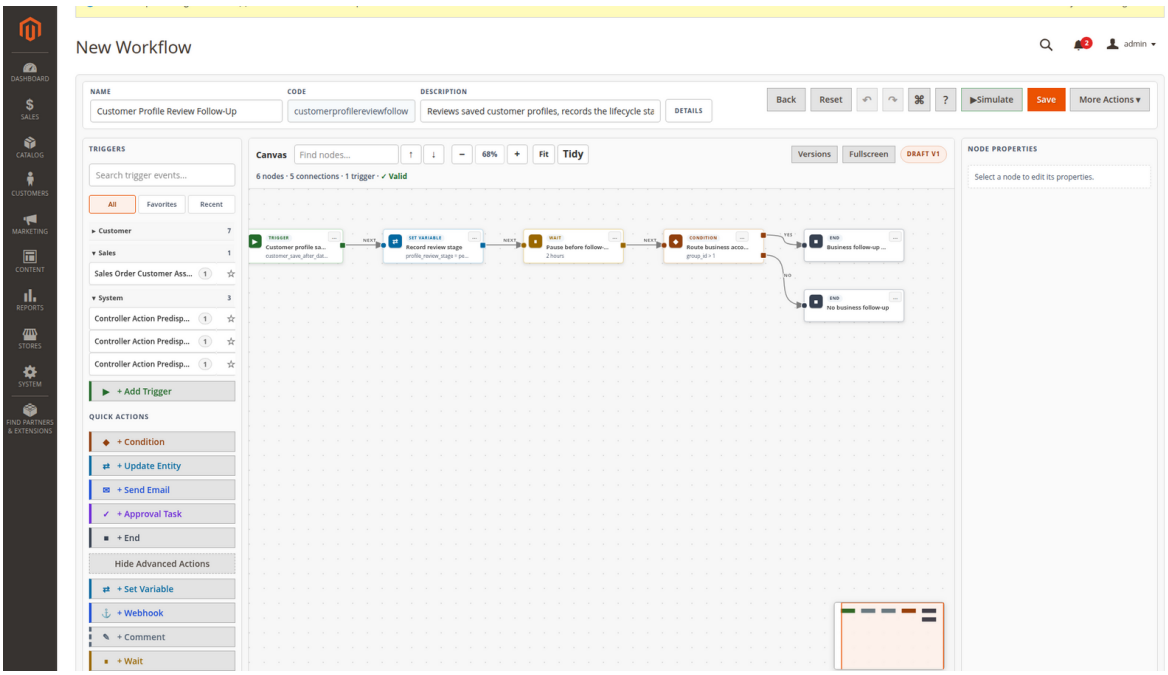
Use names that state the business step, such as `Route business account`, rather than names that repeat the node type.

Connect the graph

Hover over a source node, drag from its labelled output port and drop on the destination's **IN** port. Normal nodes expose **NEXT**. Conditions expose **YES** and **NO**. Switch nodes expose their configured cases plus a default route. Human tasks expose only the enabled decisions.

Select a connection when you need to remove it. The node menu can duplicate or delete a node, but deleting a node also removes its connections.

Fit centers the complete graph. **Tidy** arranges it automatically. The minimap helps with a graph larger than the visible canvas. Search, undo, redo, keyboard shortcuts and the command palette are available from the builder toolbar.



Connection checks before saving

- One supported trigger starts the intended flow.
- Every executable non-End node has an outgoing route.
- Every condition or switch outcome is connected to its intended destination or deliberately stops there.
- Decision ports match the decisions enabled on the Human Task node.
- No detached node is being mistaken for part of the flow.
- The builder reports **Valid**.

Long graphs are easier to review when paths stay left-to-right and alternate branches are placed above or below the main line.

Triggers and event payloads

The trigger defines when a published workflow may start and what initial context it receives.

Magento event triggers

The builder catalogue lists registered Magento events by area. Search for the exact event name, select it and confirm the event identifier on the trigger card. Event coverage depends on the installed Magento modules and the events they dispatch.

Mapped events produce a normalized payload. For a customer-save event, the payload includes a customer entity with fields such as customer ID, email, first name, last name and group ID. Sales-order events expose an order entity with fields such as increment ID, state, status, grand total and customer reference.

Do not assume that an arbitrary Magento model field is present in every event payload. Use the trigger's mapped entity fields or inspect a privacy-safe recent run before writing a condition.

Other trigger types

The runtime also supports manual, schedule, API, webhook, provider-event and raw trigger nodes. API and provider triggers have authenticated REST endpoints. Webhook triggers can require a timestamped signature and nonce. Schedule triggers depend on Magento cron.

Choose one trigger that owns the business start. Two published definitions may listen to the same event, but each creates its own independent run.

Payload structure

Magento-mapped events use the following broad structure:

```
{
  "workflow_event": {
    "schema_version": 1,
    "provider": "magento",
    "event_name": "customer_save_after_data_object"
  },
  "event": {
    "name": "customer_save_after_data_object",
    "provider": "magento"
  },
  "entity": {
    "entity_type": "customer",
    "customer_id": 24,
```

```
    "group_id": 2
  },
  "data": {}
}
```

Treat run payloads as business data. Access to payload display is protected by a separate Admin permission, and API credentials should receive only the permissions their integration needs.

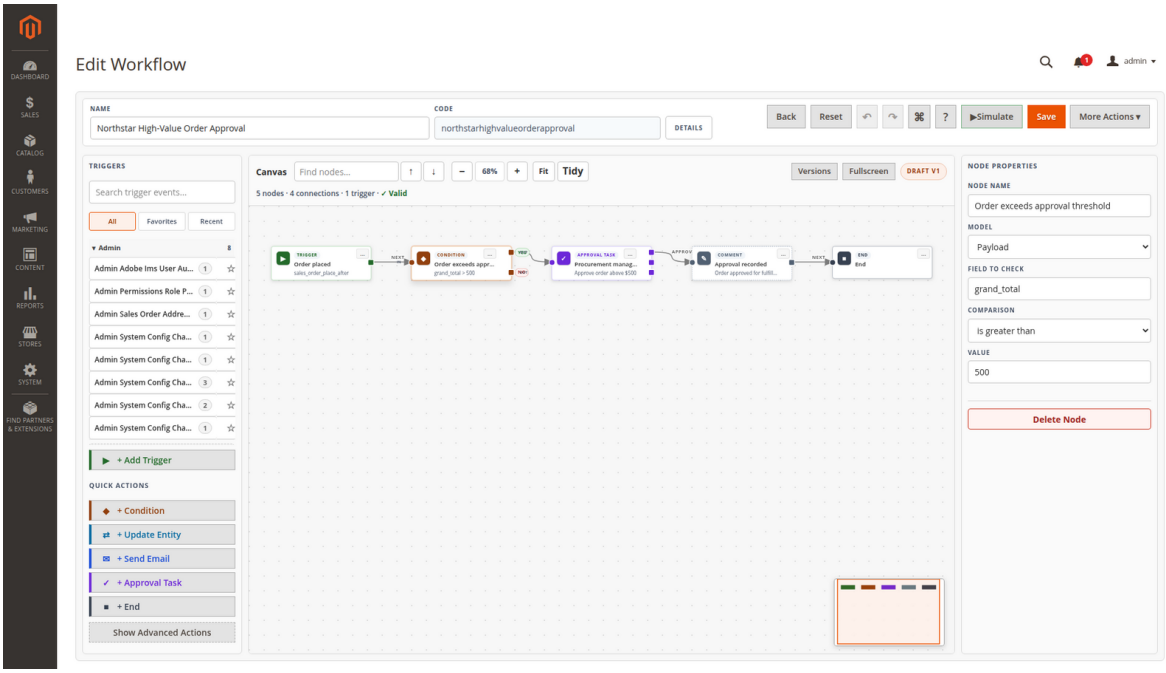
Conditions and switch routing

Use a Condition node for a yes/no decision. Use a Switch node when one value needs several named routes plus a default.

Field-based conditions

Select the payload or a supported Magento condition model, then choose a field, comparison and value. The builder presents known fields for the selected model. A saved card shows the concise rule, which makes a final visual review worthwhile.

For a high-value order, a condition can compare the event entity's grand total with the approval threshold. Connect **YES** to the approval task. The **NO** branch may go to a named End node, or remain unconnected when stopping at the condition is the intended result.



Expression conditions

Expression mode evaluates a rule against the current payload. It is useful when the decision spans nested values or cannot be expressed by one field comparison. Expressions run only while the Expression Engine setting is enabled.

Keep an expression small and use payload keys that the trigger actually supplies. For example:

```
payload["entity"]["grand_total"] > 500
```

A missing key is not the same as a value of zero. Simulate both the matching and non-matching path when optional event data is involved.

Switch routing

A Switch node evaluates one value against its configured cases. Each case becomes a connection port, and the default port catches values that did not match. Case labels should be stable identifiers rather than display prose because connections depend on them.

Practical routing review

Test each reachable branch, including the default or **NO** path. A simulation that proves only the happy path can still leave a disconnected rejection, fallback or low-value route. When business rules overlap, document which condition runs first; the graph order is the rule order.

Action nodes

Action nodes either change the workflow payload or perform work in Magento or an external system. Simulation suppresses side effects, so a successful preview should be followed by a controlled live run for any action that changes data or sends a message.

Set Variable

Set Variable writes a named value into the current run payload. Use it to carry a decision, stage or calculated marker to later nodes. It does not update a Magento entity.

Update Entity and entity state

Entity-update actions write supported fields or state to the configured Magento entity. Verify the entity type, identifier source, target field and allowed value before publishing. Run the flow first against a disposable record and confirm the saved Magento entity afterward.

Email

Email actions use Magento's email infrastructure. Confirm the template, recipient source and sender configuration. A simulation records the step without sending the message. Live delivery still depends on Magento email transport and queue operation.

Webhook

Webhook actions call an allowed public host with a permitted HTTP method. The global request, response, retry and timeout limits apply. An empty allowed-host list blocks all destinations. Do not put credentials directly into a screenshot, workflow name or documentation example.

Comment

Comment actions write a rendered message to the workflow audit trail and the run's node log. They do not add a Magento sales-order status-history comment. Use a message that will still make sense when an operator reviews the workflow run later.

Wait

Wait pauses a live run for a configured duration and records when it may resume. Magento cron queues due runs in batches, so the actual continuation also depends on cron and the run-start consumer. In simulation, Wait is shown as the last executed step and no resume job is scheduled.

End

End finishes the current path. Give alternate End nodes clear names when the completion reason matters, such as `Business follow-up queued` and `No business follow-up`.

Human tasks and decisions

A Human Task pauses a run and creates an Admin task. The workflow continues only after an allowed decision is submitted and the resumed run is processed.

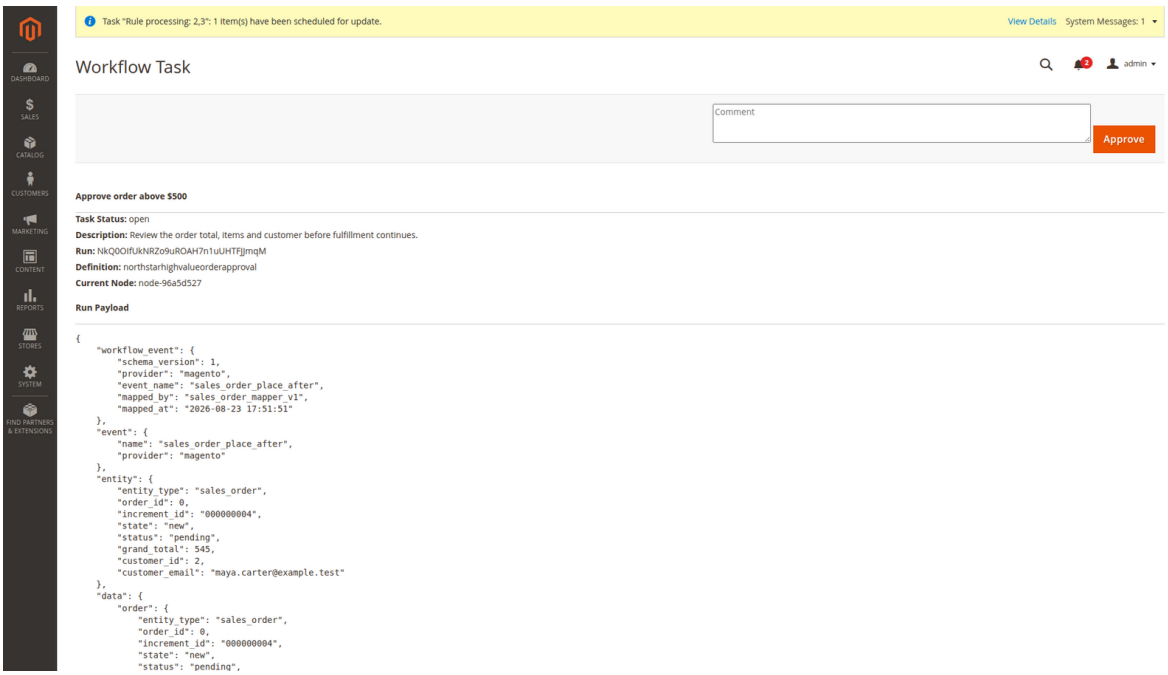
Configure the task

Select an Approval Task node and set a useful title and description. The title should tell the reviewer what is being decided; the description should name the evidence to check. Enable only the outcomes the process genuinely supports. Each enabled outcome becomes a connection port on the node.

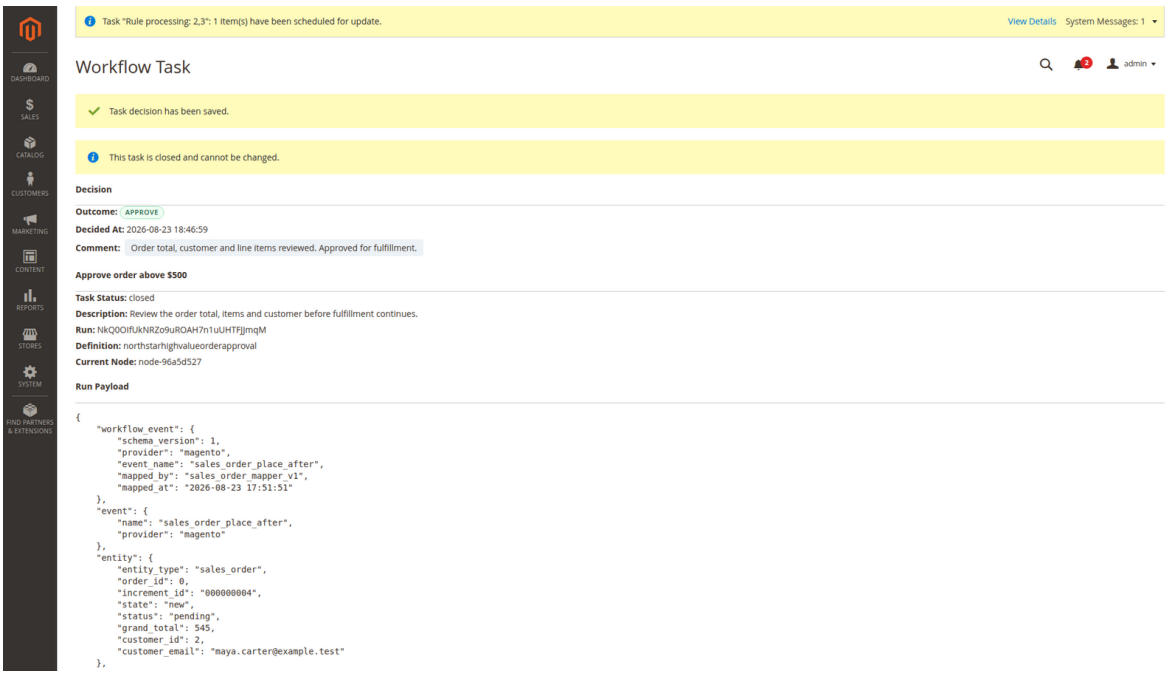
For a one-way release gate, keep only **Approve** and connect its port to the next action. For a review that can be rejected or returned, connect every enabled outcome to its own meaningful route.

Decide a task

1. Open **Content > Workflow > Workflow Tasks**.
2. Open an active task.
3. Review the definition, current node and permitted run context.
4. Enter a decision comment that records what was checked.
5. Click an available decision button once.
6. Confirm that the task is closed and shows the stored outcome, time and comment.



The decision screen renders only the outcomes allowed by the published task node. Access to the full run payload is controlled separately because it can contain customer or order data.



What happens next

Saving the decision closes the task and requests continuation. The run-start consumer must then resume the run. If the task is closed but the definition grid still shows a waiting run, check the consumer before trying to decide the task again.

A closed task cannot be changed. Correct a mistaken business decision through the owning Magento process rather than editing workflow tables.

Validation and simulation

Validation checks whether a definition can be executed. Simulation follows one route with a supplied payload while suppressing external effects.

Validate the graph

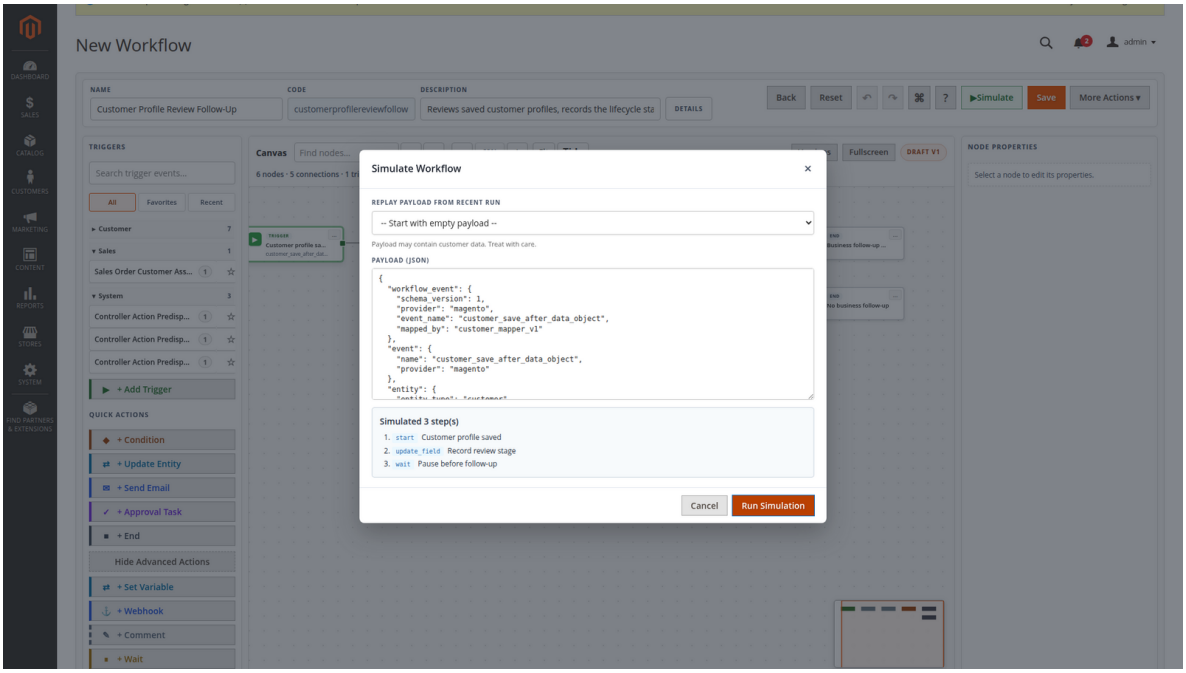
The builder validates while you work and shows either **Valid** or an issue count. Resolve issues before publishing. Common causes include a missing trigger, an incomplete node configuration, a route without a destination, an unsupported decision connection or a graph that cannot reach an End node.

Validation proves structure and configuration. It cannot prove that a real Magento event will carry the values assumed by a rule or that an external endpoint will be available.

Run a simulation

1. Save the current definition.
2. Click **Simulate**.
3. Start with an empty payload, select a privacy-safe recent run, or paste JSON that matches the trigger contract.
4. Run the simulation.
5. Compare the reported steps with the intended route.

Recent-run payloads may contain customer or order data. Use them only with the separate payload-view permission and avoid copying them into tickets or public documents.



Expected stopping points

A Human Task stops simulation at the task node without creating an Admin task. A Wait records its resume information and stops the simulated path without scheduling a live continuation. Email, webhook, entity-update and comment actions do not perform their external side effect during simulation.

This behavior lets you inspect routing safely, but it also means simulation cannot replace an end-to-end test.

Minimum pre-publish check

Simulate every condition, switch and decision route with representative values. Then publish and run one controlled Magento event through the real queue, task and cron path. Record the definition code, event, expected route and resulting run status so a later failure can be compared with a known-good run.

Publishing and versions

Saving stores the editable definition. Publishing creates the executable contract for new runs.

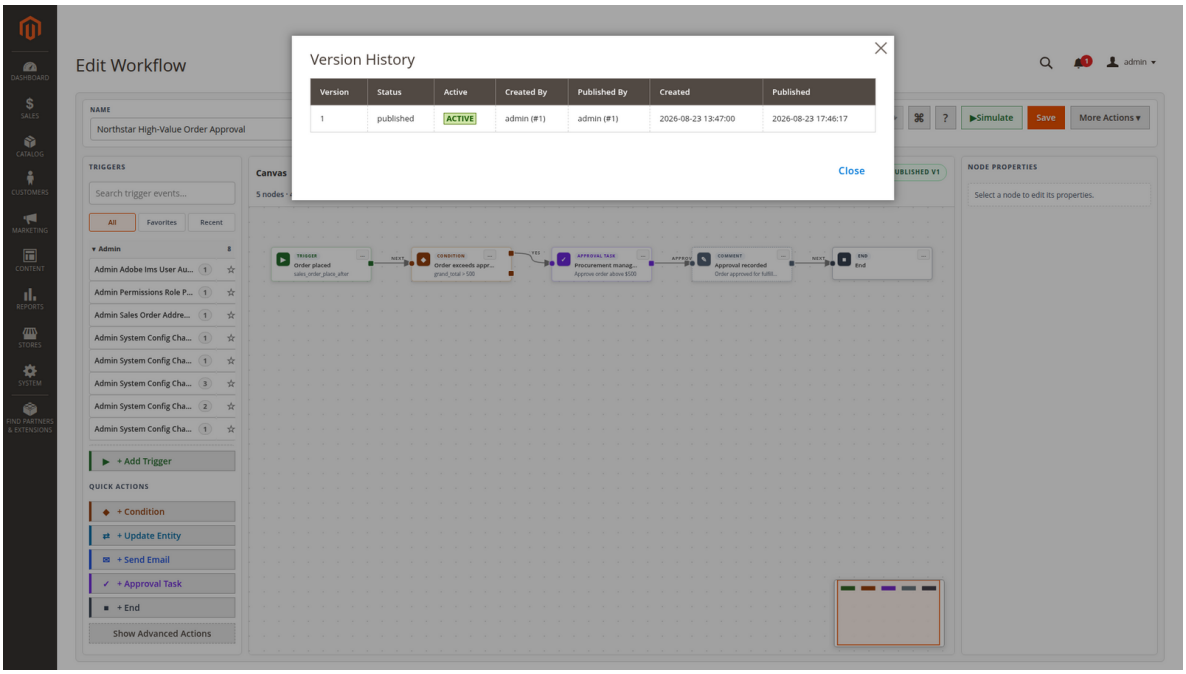
Publish a definition

1. Confirm that the builder reports **Valid**.
2. Save the latest changes.
3. Open **More Actions** and choose **Publish**.
4. Confirm the publication prompt.
5. Return to the definition grid and verify **Published** and the active version number.

Only users with the publish permission can make a version live. A definition can also be archived and later restored through the supported management actions or API.

Version history

Click **Versions** in the builder to see each stored version, its status, creator, publisher and timestamps. One published version is marked active.



Runs retain their definition version. If a workflow changes while an approval is open or a wait is pending, the existing run does not silently switch to the new graph.

Safe change process

For a production change, export the current definition, edit the draft, validate it, simulate every changed route, and publish during a controlled window. Trigger a known test case

and confirm the active version and final run state.

Do not use a display-name edit as a substitute for a new business-process review. Changes to event names, expressions, switch cases, side-effect targets and task decisions can all change runtime behavior even when the canvas still looks similar.

Import and export

Export packages a definition as JSON for review, backup or transfer between Magento environments. Import validates the package before saving it.

Export

Open the definition and choose **More Actions > Export JSON**. Store the file with the deployment change that owns it. Review exported files as configuration code: they contain graph structure, triggers, expressions and action settings.

Import

1. Open **More Actions > Import JSON** from the builder.
2. Select the reviewed package.
3. Confirm the imported definition code and graph.
4. Validate and simulate it in the target environment.
5. Publish it only after the target store's events, email identities, entity fields, webhook allowlist and worker setup have been checked.

The maximum package size comes from **Stores > Configuration > General > Workflow Engine > Import / Export**. Overwrite is disabled by default. With overwrite disabled, an imported code that already exists is rejected instead of replacing the definition.

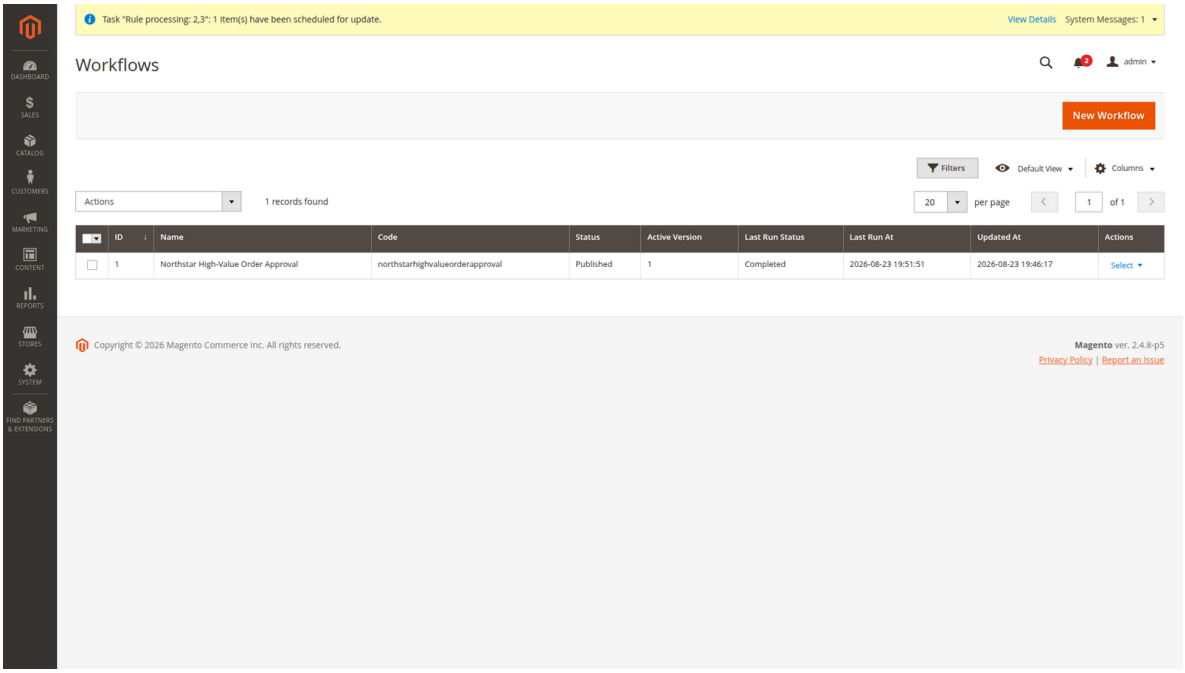
Environment-specific checks

An export can describe a valid graph while still referring to a template, event, customer group, status, URL or field that differs in the target store. Import does not make those dependencies portable. Keep secrets out of export files, and configure sensitive integration values through the supported secure mechanism for the target environment.

An imported definition is not automatically a proven live process. Treat validation, simulation and publication as separate steps.

Runs and tasks

The definition grid shows the most recent run status and timestamp. It is the quickest place to confirm whether a published workflow has executed.



Run states

A new trigger creates queued work. The run may then be running, waiting for a delay or task, completed, or failed. Exact timing depends on the queue consumers and cron. A missing latest-run value usually means no matching event has produced a run for that definition.

Use the run API when an integration or support tool needs a list or detailed status. Detailed payload access can reveal customer and order data, so the Admin UI protects it with a dedicated permission.

Task states

An open task accepts one of the decisions enabled on its Human Task node. After a decision, it is closed and cannot be changed. The decision records the outcome, time and optional operator comment. The run then needs the run-start consumer to continue from the corresponding decision port.

Diagnose a run that does not finish

1. Confirm that the definition was published before the event occurred.
2. Check whether the grid shows a run and note its status and time.
3. If no run exists, check the event-trigger consumer and whether the trigger name

matches the Magento event.

4. If a run waits at a Human Task, open Workflow Tasks and review the task state.
5. If a closed task or elapsed Wait has not continued, check cron and the run-start consumer.
6. If the run failed, inspect the Magento exception and system logs around the run time.

Do not repair runs by editing workflow database tables. Preserve the run ID and definition version when escalating the issue.

Queues and cron

Live workflows rely on two Magento queue consumers and one cron job.

Component	Purpose
`workflow.event.trigger.consumer`	Processes registered event messages and matches them to published definitions.
`workflow.run.start.consumer`	Starts or resumes workflow runs and advances their nodes.
`workflow_resume_due_runs`	Runs each minute and queues waiting runs whose resume time has arrived.

Run the consumers through the same supervised process used for the store's other Magento consumers. For a manual diagnostic, Magento can start an individual consumer with `queue:consumers:start`, but a long-running production setup needs process supervision and restart handling.

```
php bin/magento queue:consumers:start
workflow.event.trigger.consumer
php bin/magento queue:consumers:start
workflow.run.start.consumer
```

Health check

1. Confirm that Magento cron runs successfully.
2. Confirm that both consumers are configured and remain alive.
3. Publish a small controlled workflow.
4. Trigger its Magento event.
5. Verify that the definition grid receives a latest-run status.
6. For a Human Task, decide it and verify that the run continues.
7. For a short Wait in staging, verify that cron queues it after the due time and the run consumer completes it.

Queue backlog explains several misleading symptoms: a real order exists but no task appears; a decision is saved but the run stays waiting; or a wait passes without continuation. Check the correct stage before changing the definition.

The runtime retry and batch limits are global settings. Increasing them is not a substitute for fixing a repeatedly failing node or undersized worker pool.

API integration

The module exposes authenticated Magento REST services for definition management, runtime inspection, external triggers and task decisions. Use an integration token whose role contains only the required workflow resources.

Definition management

Supported operations include:

- list and retrieve definitions;
- save and validate a definition;
- simulate and publish a definition;
- archive and restore a definition;
- list templates;
- export and import a definition package.

The routes are under `/V1/workflow`. Definition codes identify individual resources. Validate and simulate before publishing an API-supplied graph just as you would in Admin.

Start a workflow

`POST /V1/workflow/triggers/api/{code}` accepts a `payload` object for a definition with a compatible API trigger. The provider route also accepts the event name, event data and provider. A successful trigger response confirms that work was accepted; the run is still asynchronous.

Example request body:

```
{
  "payload": {
    "source": "procurement_portal",
    "request_id": "REQ-1048",
    "amount": 789.00
  }
}
```

Inspect runs and decide tasks

`GET /V1/workflow/runs` supports limit and offset values. `GET /V1/workflow/runs/{runId}` returns one run. Task decisions use `POST /V1/workflow/tasks/{taskId}/decision` with a `decisionData` object containing an allowed decision and any supported comment data.

Do not retry a task decision blindly after a timeout. Check the task in Magento Admin first because the original request may already have closed it.

Integration safeguards

Keep API-trigger permission separate from workflow publication and task operation. Validate payload size and field count at the sending system as well as in Magento. Use idempotent external reference values where the calling process may retry, and record the returned run identifier for support tracing.

Permissions and security

Open **System > Permissions > User Roles**, edit a role and expand the Workflow resources.

Permission	Grants
Manage Workflows	Definition grid, builder and management API access.
Publish Workflows	Publish, archive and restore operations.
Simulate Workflows	Simulation in Admin or through the management API.
View Workflow Run Payloads	Access to stored run payload details.
Advanced Workflow Features	Global Workflow Engine configuration and advanced features.
Operate Workflow Tasks	Task grid, task decisions and task-decision API.
Trigger Workflows via API	API and provider trigger endpoints.

Separate authoring, publishing and task operation for processes that affect orders or customer data. A reviewer who approves a purchase does not automatically need permission to edit or publish its workflow.

Payload privacy

Event and run payloads can contain names, email addresses, order values and internal identifiers. Grant payload viewing only to roles that need it. Redact payloads before sharing diagnostics outside the authorized support path.

Webhook and expression controls

Use a narrow webhook host allowlist. An empty allowlist denies all hosts; `*` allows all public hosts. Private and unsafe destinations remain subject to runtime protections, but a broad allowlist still enlarges the outbound integration surface.

Limit who can enable expressions, expose additional flat-model prefixes or import definitions. Imported JSON and expressions are executable business configuration and should receive the same review as a code deployment.

Inbound signed webhook triggers depend on a current timestamp and unused nonce. Keep system clocks synchronized and never expose signing secrets in workflow names, exports or screenshots.

Common issues

The definition cannot be published

Open the builder's issue list. Check for a missing trigger, incomplete required fields, detached paths, missing destinations or a branch that cannot reach an End node. Save the corrected graph, confirm **Valid**, then publish again with a role that has Publish Workflows permission.

A Magento event did not create a run

Confirm that the definition was published before the event and that the engine is enabled. Compare the trigger's exact event identifier with the source event. Then check `workflow.event.trigger.consumer`. Saving or simulating a definition does not start the live event consumer.

The task list is empty after a matching order

First check whether the definition grid has a latest run. No run points to event dispatch or event-consumer processing. A running workflow that has not reached the task points to the run-start consumer or an earlier condition. Simulate the same payload shape and check that the approval branch is selected.

A task is closed but the run still waits

The decision and the resumed run are separate steps. Confirm that the task shows the stored decision, then check `workflow.run.start.consumer`. Do not submit the decision again or edit the task record.

A Wait node never resumes

Magento cron must execute the minute-based `workflow_resume_due_runs` job, and the run-start consumer must process the queued continuation. Check both. Also confirm the store and server clocks when the due time looks wrong.

Simulation stops before the end

This is expected at Human Task and Wait nodes. Simulation does not create a real approval task or schedule a delayed continuation. The displayed steps should still prove that routing reached the intended pause.

A webhook is rejected

Check the global allowed-host list, HTTP method, request and response size limits, retry cap and node timeout. An empty allowed-host list denies every destination. For signed

inbound triggers, also check timestamp age, allowed clock skew and nonce reuse.

An import reports that the code already exists

Overwrite is disabled by default. Either import under a new reviewed code or explicitly enable overwrite at Default Config scope for the controlled replacement. Turn it off again if overwriting is not part of the normal deployment policy.

An expression does not take the expected route

Confirm that expressions are enabled and that the payload contains every referenced key with the expected type. Test matching, non-matching and missing-value payloads. A numeric string, number and absent field may not behave as the same business value.

What to include in a support request

Provide the definition code, active version, trigger type and identifier, approximate event time, run ID if available, run status, current node, expected route and exact error. Include sanitized payload fragments only when they are needed to reproduce the rule. Never send credentials, webhook signatures or full customer payloads.

FAQ

Does saving a workflow make it live?

No. Saving stores the draft. Use **More Actions > Publish** to make a validated version active for new runs.

What happens to an open run when I publish a change?

It continues with the version with which it started. New runs use the newly published active version.

Does simulation send emails or webhooks?

No. Simulation suppresses email, webhook, entity-update and comment side effects. It also does not create a real human task or scheduled wait continuation.

Why does simulation stop at an approval or wait?

Those nodes require time or an external decision. The simulation reports the route up to the pause so you can verify that the correct node was reached.

Can a workflow update an order or customer?

Supported entity-update actions can change configured Magento entity fields or state. A Set Variable node changes only the workflow payload. Test any entity update with a disposable record before publishing it.

Can one event start several workflows?

Yes. Each published definition listening to that event can create its own run. They do not share a payload after the runs start.

Which decision buttons appear on a task?

Only decisions enabled on the published Human Task node should be available. Connect every enabled decision port to a valid route before publishing.

Can a closed task be changed?

No. It records one final decision. Handle a mistaken approval or rejection through the business process that owns the affected Magento record.

Why is there no latest-run status?

The definition has not yet produced a processed run. Check its published state, trigger match and event-trigger consumer.

How do waits continue?

Magento cron finds due runs every minute and queues them in configured batches. The run-start consumer then resumes execution.

Can I transfer a workflow between stores?

Yes, through JSON export and import. Review target-store dependencies such as events, fields, templates, statuses, URLs and allowlists before publishing the imported definition.

Is import allowed to replace an existing definition?

Not by default. **Allow Overwrite on Import** must be enabled at Default Config scope for an intentional replacement.

Can an external system start a workflow?

Yes, for compatible API, provider or webhook triggers. Use an authenticated integration with the dedicated API-trigger permission and keep the returned run reference.

Where should I look first when a workflow appears stuck?

Check the definition grid for a run, then its current stage: event consumer before run creation, run-start consumer during execution, Workflow Tasks for a manual pause, and cron plus the run-start consumer for a Wait.