Chatwoot customer portals

Use these five capabilities to list and create help-center portals, update a portal, and create categories or articles inside a portal. The reviewed surface is intentionally smaller than full portal CRUD.

SECTION	Connectors
TYPE	Connector
PROTOCOL	AIP 1.0
SOURCE	docs/connectors/chatwoot/capabilities/customer-portals.md

Portal content can become customer-visible. Review the provider body, intended audience, and publication behavior before approving a mutation.

Family at a glance

Operations	Reads	Mutations	High risk	Deletes
5	1	4	0	0

All five capabilities are Tools. `portal.list` is a low-risk `GET`. The three `POST` operations and one `PATCH` operation are medium-risk mutations.

Every mutation requires approval and an idempotency key. None advertises connector retry support or compensation.

Operations

Capability	Provider request	Purpose	Risk	Approval
`cap:chatwoot:portal.list`	`GET /api/v1/accounts/{account_id}/portals`	List account help-center portals	Low	No
`cap:chatwoot:portal.create`	`POST /api/v1/accounts/{account_id}/portals`	Create a help-center portal	Medium	Required
`cap:chatwoot:portal.update`	`PATCH /api/v1/accounts/{account_id}/portals/{portal_id}`	Update a help-center portal	Medium	Required
`cap:chatwoot:portal.category.create`	`POST /api/v1/accounts/{account_id}/portals/{portal_id}/categories`	Create a category	Medium	Required
`cap:chatwoot:portal.article.create`	`POST /api/v1/accounts/{account_id}/portals/{portal_id}/articles`	Create an article	Medium	Required

The provider path and operation description use help-center portal semantics. These capabilities do not manage the AIP documentation site or connector-host HTTP routes.

Supported and absent operations

Object	Supported	Not present in the reviewed catalogue
Portal	List, create, update	Get by ID, delete
Category	Create	List, get, update, delete
Article	Create	List, get, update, delete

Do not synthesize an absent capability ID or call an arbitrary provider route. Admission applies only to the five exact suffixes above.

The missing delete surface also limits recovery. A corrective provider operation outside this catalogue is not an AIP compensating capability and requires a separate authorized path.

Account and path inputs

The connector inserts its configured `account_id` into every route. It is not an Action input.

`portal_id` is required by:

- `portal.update`;
- `portal.category.create`;
- `portal.article.create`.

The value must be a non-blank string of at most 512 bytes without control characters. The connector inserts it as one encoded path segment.

`portal.list` and `portal.create` use the account collection route. Category and article identities belong to provider responses or bodies, not to an invented path placeholder.

Generic provider input

Every operation accepts the common catalogue envelope:

Property	Connector boundary
`query`	Optional object; at most 128 keys; scalar values or bounded scalar arrays
`body`	Optional object; at most 512 properties and 16 MiB encoded
`multipart`	Optional bounded fields-and-files object instead of `body`

The connector rejects `body` and `multipart` together. It does not validate portal names, domains, locales, visibility, category structure, article content, publication state, or other provider fields.

Use the exact pinned provider schema. Retain one canonical body with the approval record; do not edit customer-visible content after approval.

List portals

This read needs no provider-specific path or body input:

```
SH
aipctl action call \
  "$AIP_URL" \
  cap:chatwoot:portal.list \
  --action-id act_chatwoot_portal_list_001 \
  --input '{}'
```

The connector sends:

```
TEXT
GET /api/v1/accounts/{configured_account_id}/portals
```

The read needs no capability-level approval and advertises safe retry. Its provider body remains restricted and PII-present under the common contract.

The list result is the only portal read in this family. Confirm that its provider shape contains the fields required for review before relying on it as an authoritative snapshot.

Review a portal mutation

Intent	Review before approval	Verify after completion
Create a portal	Exact provider body, account, audience, and expected visibility	Find the returned portal in the list
Update a portal	Current list evidence, portal ID, changed fields, and visibility impact	List portals and compare the target
Create a category	Portal identity, category body, hierarchy, and audience	Inspect the provider result and available portal state
Create an article	Portal identity, article body, links, restricted data, and audience	Inspect the provider result and available portal state

The source does not state whether a created object becomes visible immediately. Treat publication behavior as provider-owned and verify it through the pinned Chatwoot surface.

Do not approve a guessed body. The generic schema proves only that the body is a bounded JSON object.

Shared mutation contract

All four mutations declare:

Contract field	Value
Approval	Required; tenant-policy selector
Approval validity	900,000 ms
Approval evidence	Reason, input snapshot, policy decision
Idempotency	Required; external-account scope; 86,400,000 ms published TTL
Connector retry	Not supported; unsafe
Side effects	`write`, `external_network`
Data	Restricted; contains PII; redaction required
Execution	Synchronous; no async, streaming, or cancellation support
Compensation	Rollback not supported

The connector forwards the idempotency key to Chatwoot. The contract does not prove provider deduplication or content-level rollback.

Result contract

Every successful operation returns:

```
JSON
{
  "http_status": 200,
  "body": {},
  "provider_request_id": null
}
```

`body` contains parsed JSON, `null` for an empty response, or a `text` wrapper for non-JSON bytes. `provider_request_id` reads `x-request-id` first and `x-runtime` second.

The output schema does not freeze a portal, category, or article shape. Validate required provider fields and preserve unknown fields.

Failures and recovery

Failure code	Typical cause	Safe response
`connector.chatwoot.invalid_action`	Unknown or unadmitted ID, missing portal ID, or malformed generic input	Refresh discovery or correct input before dispatch
`connector.chatwoot.missing_idempotency_key`	Mutation lacks a valid key	Restore the key for the original content intent
`connector.chatwoot.authentication`	Provider returned `401` or `403`	Verify account route and credential revision
`connector.chatwoot.remote_rejected`	Provider rejected fields or current state	Correct authoritative input before a new Action
`connector.chatwoot.remote_temporary`	Provider returned `429` or a server error	Retry the list read within policy; reconcile a mutation
`connector.chatwoot.transport`	Provider exchange failed	Retry the list read within policy; treat mutation outcome as uncertain
`connector.chatwoot.response_too_large`	Response crossed the active bound	Preserve evidence and reconcile a mutation

All connector failures set `retryable: false`. Only `portal.list` advertises safe retry.

For an uncertain create, list portals and inspect the original provider result evidence before another Action. Duplicate portal, category, or article creation cannot be repaired through a delete capability in this catalogue.

For an uncertain update, compare the target in the list when its shape permits. A corrective update needs new approval against current provider state.

Verify an integration

Require all of the following:

- discovery admits one of the five exact portal capabilities;
- the task does not depend on an absent get, update, or delete surface;
- `portal_id` belongs to the configured account;
- provider body fields and publication behavior come from the pinned API;
- approval binds the exact customer-visible content and intended audience;
- each mutation retains one Action, canonical input, and idempotency key;
- available provider state confirms the completed change;
- portal content and results follow restricted-data policy;
- uncertain mutations are reconciled instead of repeated.

These checks validate application handling. They do not establish publication state, provider payload compatibility, or live qualification.

Related documentation

- Chatwoot capability index
- Automation, macros, and canned responses
- Chatwoot connector configuration

- Approvals and policy
- Error reference