Dify modes and capability projection

This reference defines how startup descriptors become Dify capabilities in tenant discovery. Use it to predict a manifest, diagnose a missing suffix, or detect a mode that exposes an unintended provider route.

SECTION	Connectors
TYPE	Connector
PROTOCOL	AIP 1.0
SOURCE	docs/connectors/dify/reference/modes-and-capability-projection.md

Projection is static. The connector reads each configured mode at startup and applies compiled rules; it does not ask Dify to infer the app mode during discovery.

Projection identity

The frozen catalogue contains 33 app operations and 46 Knowledge operations. Each configured app publishes one invocation alias plus a mode-dependent subset of the app catalogue.

```
TEXT
cap:dify:<app_id>
cap:dify:<app_id>:<app_operation>
cap:dify:knowledge:<credential_id>:<knowledge_operation>
```

App IDs and Knowledge credential IDs must be unique within their descriptor domain. The app ID `knowledge` is reserved because it would collide with the Knowledge routing prefix.

Map mode to the app alias

The connector accepts six exact mode strings:

Mode	Alias provider route	Alias kind	Connector approval
`workflow`	`POST /v1/workflows/run`	Workflow	Required
`completion`	`POST /v1/completion-messages`	Agent	Not required
`chat`	`POST /v1/chat-messages`	Agent	Not required
`advanced-chat`	`POST /v1/chat-messages`	Agent	Required
`agent-chat`	`POST /v1/chat-messages`	Agent	Required
`agent`	`POST /v1/chat-messages`	Agent	Required

Every alias is medium-risk and requires a tenant-scoped idempotency key. All publish sync, async, streaming, and cancellation support, no safe retry, and rollback unsupported.

The absence of connector-required approval for plain completion and chat does not override tenant policy.

Apply the five operation projection sets

Every app operation belongs to one of five sets.

All six modes

These 18 operations project for every configured app:

Family	Operation suffixes
App metadata and files	`parameters.get`, `meta.get`, `info.get`, `site.get`, `file.upload`, `file.preview`, `end_user.get`
Application feedback	`feedback.list`
Audio	`audio.transcribe`, `audio.synthesize`
Human input	`human_input_form.get`, `human_input_form.submit`
Annotations	`annotation_reply.toggle`, `annotation_reply.status`, `annotation.list`, `annotation.create`, `annotation.update`, `annotation.delete`

Projection does not remove route-specific preconditions. A human-input token, audio feature, annotation setting, file identity, or end user must still satisfy its provider contract.

Completion only

`completion.stop` projects only for `completion`.

The operation uses the completion-message task route. The connector never converts a completion task ID into a chat or workflow stop.

Four chat-like modes

These nine operations project for `chat`, `advanced-chat`, `agent-chat`, and `agent`:

- `chat.stop`
- `message.list`
- `message.feedback.create`
- `message.suggested.get`
- `conversation.list`
- `conversation.delete`
- `conversation.rename`
- `conversation.variable.list`
- `conversation.variable.update`

The connector treats all four as the chat provider family. Route-specific user identity and UUID rules still apply.

Workflow and advanced-chat

`workflow.run.get` and `workflow.log.list` project for `workflow` and `advanced-chat`.

These reads are app-scoped and can expose records from multiple end users. Their availability does not make the advanced-chat alias a Workflow kind.

Workflow only

These three operations project only for `workflow`:

- `workflow.run_by_id`
- `workflow.stop`
- `workflow.event.stream`

Run-by-ID and event stream use the connector's bounded SSE implementation. Workflow stop remains a separate approved mutation.

Calculate per-app capability counts

The app-operation count comes directly from the five sets. Add one for the app invocation alias.

Mode	App operations	Alias	Published capabilities
`workflow`	23	1	24
`completion`	19	1	20
`chat`	27	1	28
`advanced-chat`	29	1	30
`agent-chat`	27	1	28
`agent`	27	1	28

Changing an app name or description does not alter the count. Changing its ID changes every capability identity. Changing its mode changes the alias binding, approval rule, and projected operation set.

Add independent Knowledge capabilities

Knowledge descriptors have no app mode. Every configured Knowledge credential publishes all 46 Knowledge operations with this prefix:

```
TEXT
cap:dify:knowledge:<credential_id>:<operation>
```

An app credential never contributes Knowledge capabilities. A Knowledge credential never contributes an app alias or app operations.

For a host with configured apps and Knowledge credentials, the expected total is:

```
TEXT
sum(per-app published count) + 46 × knowledge credential count
```

One `chat` app and one Knowledge credential therefore publish 74 capabilities. This is fixture arithmetic, not a universal Dify manifest size.

Parser limits allow many descriptors, but global manifest admission can impose a lower practical maximum. Split instances before weakening admission limits.

Interpret operation kind and policy

Six app operations have Workflow kind:

- `workflow.run_by_id`
- `workflow.run.get`
- `workflow.stop`
- `workflow.event.stream`
- `human_input_form.get`
- `human_input_form.submit`

The other 27 app operations have Tool kind. Mode projection does not change an operation's provider method, kind, risk, approval, retry, response type, or compensation contract.

Every GET remains a low-risk retryable read. Every other catalogue method is a mutation requiring approval and a key, with no safe retry. DELETE, workflow run-by-ID, annotation-reply toggle, and annotation delete follow their compiled high-risk rules.

Separate projection from runtime evidence

Evidence	What it proves	What it does not prove
Parsed descriptor	Valid local ID and one supported mode	Provider app really uses that mode
Generated manifest	Compiled capability projection	Tenant authorization for every caller
Registry admission	Accepted identity and artifact boundary	Provider route compatibility
App health probe	App key can read `/v1/parameters`	Invocation or every projected operation
Knowledge health probe	Knowledge key can list one dataset page	Any mutation, indexing, or pipeline run
Successful operation	One route worked for one input	Entire mode or catalogue qualification

Discovery can therefore be internally correct while a provider rejects a route. Treat configuration, discovery, admission, readiness, and qualification as separate gates.

Account for pinned projection mismatches

Two reviewed mismatches require explicit caller handling:

Connector projection	Pinned provider boundary	Decision
Conversation-variable operations include `agent`	Provider accepts chat, advanced-chat, and agent-chat	Do not invoke those two suffixes for `agent`
`workflow.event.stream` is workflow-only	Provider route also accepts advanced-chat	Do not construct the missing advanced-chat capability

The first mismatch is an exposed capability that the provider rejects. The second is a provider route intentionally absent from connector discovery.

Do not patch Action IDs, bypass discovery, or call the provider route directly to compensate for either difference.

Change a projected surface safely

App and Knowledge descriptor files, modes, IDs, and key maps are startup-owned. Editing a file does not change a running manifest.

Use a replacement replica and new configuration or admission revision. Compare the expected count and exact capability set before routing traffic.

An app ID or mode change can invalidate stored caller assumptions, policy rules, task mappings, dashboards, and documentation links. Treat it as a versioned integration change.

Diagnose projection differences

Observation	First decision
Alias is absent	Verify descriptor parsing, unique ID, key source, and admission
Alias route is unexpected	Compare the exact startup mode with the six-mode table
App operation is absent	Find its projection set; do not derive the suffix manually
Knowledge operations are absent	Verify a separate Knowledge descriptor and key file
Count is one below expected	Check whether the app alias was omitted from the calculation
Count differs by 46	Check the Knowledge credential count
Projected route is rejected	Compare the pinned provider mode boundary and known mismatches
Configuration changed without discovery change	Replace the startup-owned host replica
Admission rejects a large manifest	Reduce or split descriptors; parser ceilings are not admission guarantees

Related documentation

- Dify connector overview
- Dify connector configuration
- Dify capability index
- Dify authentication and credential scopes
- Errors and retry decisions