indition™

INDITION AI ASSISTANT

Setup Guide

Install AI Assistant from zero: the AI Core dependency, module enablement, migrations, default quotas and model parameters, and a first assistant.

AI Assistant Module · Version 1.0 · June 2026

Indition AI Assistant — Setup Guide

> An installation guide for administrators standing up AI Assistant from zero: confirming the AI Core dependency, enabling the module, running the migrations, setting default quotas and model parameters, and building your first assistant end-to-end.

This guide is for platform administrators and deployers. To build and run assistants day to day, see the **User Manual**; for architecture and internals, see the **Technical Reference**.

Before you start

- Administrator access to the target site/instance and its databases.
- The ability to deploy the paired React frontend and import route/menu/access metadata.
- **Indition AI Core configured and working** — AI Assistant cannot generate anything without it (see chapter 1).

Install at a glance

1. Configure AI Core: provider, credential, models — including an **embedding** model.
2. Enable the AI Assistant module and run its package migrations.
3. Import route/menu/access metadata, sync OAuth, clear caches, re-login.
4. Deploy the frontend.
5. Set default quotas and model parameters.
6. Build a first collection → assistant → chat app and verify end-to-end.

1. Prerequisite: configure AI Core first

> AI Assistant is a **consumer of the Indition AI Core** infrastructure layer. Every answer, embedding, and document extraction goes through AI Core. If AI Core isn't set up, no assistant can do anything — set it up before installing AI Assistant.

In AI Core (see the **AI Core Administrator Guide**), confirm all of the following:

- **AI enabled** — the master switch `AI_CORE_ENABLED` is on.
- **A provider** — at least one active provider with a correct base URL.
- **A credential** — a valid, active (encrypted) API key for that provider.
- **A generation model** — at least one active model with accurate input/output pricing (cost flows into AI Assistant's quotas and Cost & Usage report).
- **An embedding model** — required for knowledge retrieval (RAG). The model must support embeddings.
- **Sharing** — the provider/model is shared (or default-shared) to the customer that will use AI Assistant.

> **Embeddings note:** retrieval depends on an embedding-capable provider/model in AI Core. If you mix providers (for example, one vendor for answer generation and another for embeddings), make sure the embedding side is covered — an assistant with a generation model but no working embedding model can index sources but won't retrieve them well.

Quick check: from AI Core's request log, confirm a test generation and a test embedding both succeed before moving on.

2. Install the module

Enable AI Assistant

Add the module to the enabled-module list for the target site, following your platform's standard module-enablement process.

Run the database migrations

Run the AI Assistant package migrations to create the runtime tables (tenant-prefixed `{SITE_ID}_aia_*` — assistants, collections, resources, chunks, conversations, messages, feedback, configured responses, quotas, usage buckets, and more). Per-customer quota/usage data is cross-site.

Import metadata & sync access

1. Import the module's route, menu, and access metadata (the `configs/` and `accessRules/` definitions) using your site's metadata-sync tooling.
2. Sync the OAuth access components/groups so the four access scopes resolve: the product menu group, cost/usage reports, model-parameter settings, and owner-quota administration.
3. Clear caches and re-login so new roles, menus, and routes take effect.

Deploy the frontend

Deploy the paired React frontend (`core/app/aiassistant`). Confirm the AI Assistant menus appear for an admin account: Assistants, Conversations/Inbox, Global Pre-configured Q&A, Reports, and the administration items (Quotas, Default Quotas, Model Parameters).

3. Configure defaults

Default quotas

Before customers start creating assistants, set sensible platform-wide defaults under **Default AI Assistant Quotas** — these apply to new customers and can be overridden per customer later under **Manage AI Assistant Quotas**. Quota dimensions (0 = unlimited):

- **Structural**: max assistants, resources, chunks, storage (MB).
- **Daily**: messages, AI requests, tokens, cost (USD).
- **Monthly**: messages, AI requests, tokens, cost, ingestion jobs.
- **Public messages**: the "approaching limit" and "blocked" text end users see.

Model parameters

Set the default model and parameters under **AI Assistant Model Parameters**. Permitted users can override the model per assistant in the workspace. The actual models/providers come from AI Core.

> **Cost guardrails first:** set daily and monthly cost limits in the default quota before opening the module to customers. AI spend accrues per message; the Cost & Usage report breaks it down by model and assistant once traffic starts.

4. Build your first assistant (smoke test)

Walk the full path once to confirm everything is wired up:

1. **Create a collection** in the Collection Builder; give it a name and description.
2. **Add a few sources** — upload a PDF/DOCX/TXT or add a URL.
3. **Wait for indexing** — confirm sources move to *indexed* (not *failed*). A failed source shows its last error; reprocess after fixing.
4. **Create an assistant** with the creator wizard, pointed at the collection; enable citations.
5. **Test chat** — ask a question your sources answer; confirm a grounded reply with citations.
6. **Build a chat app** and copy the embed code from **Settings** → **Widget**.
7. **Embed & verify** — place the snippet on a test page and confirm the widget answers.

5. Verification checklist

Check	Expected
AI Core ready	Master switch on; active provider/credential/generation model/embedding model; shared to the customer.
Migrations applied	`{SITE_ID}_aia_*` tables exist.
Access + re-login	AI Assistant menus visible; admin sees Quotas/Model Parameters; agents see the product menu.
Frontend deployed	Dashboard, workspace, collections, inbox, reports load.
Defaults set	Default quotas (incl. cost limits) and model parameters configured.
Indexing works	A test source reaches *indexed*.
Grounded answer	Test chat returns an answer with citations, not just the fallback.
Embed works	The chat app answers on a test page; request log shows the calls with cost.

Stuck? The **FAQ & Troubleshooting** guide covers the usual first-run issues: "no AI at all" (AI Core disabled or no active provider/model), sources stuck/failed (type, size, or unreachable URL), only-fallback answers (no indexed sources), and quota blocks.