Setting up the AI assistant

The **AI assistant** is a chat in the side panel of Webdesk. It answers questions of the logged-in user and can call predefined **tools** for this purpose. The tools read data from Webdesk (persons, groups, time and attendance, project time) and, in one case, also write data (attendance booking). The answers are generated by an external language model that the administrator connects through a **model** (credentials and endpoint). Optionally, **agents** combine a model, additional instructions and permitted tools.

This page is intended for administrators and consultants. It describes the setup, the order in which key, model and endpoint are determined, the tool catalog, the authorization checks and the data flows to the external provider. The feature is mentioned briefly in New Features[1]; the general handling of system parameters is described in System parameters[2].

Warning: Once the assistant is configured, texts entered by the user, information about the user and the results of the tools that were called are transmitted to the configured external provider (details in the section *Data protection*). Before activating it, clarify whether this is permitted for the company.

Prerequisites

- **Licence:** The licence definition ai-impl.ldef.xml defines the licence ai (module ai) with the description "Number of users allowed to chat with custom AI agents". The licence assigns the module its Spring contexts and all of its actions (no action is excluded). The licence usage can be viewed under *System* > *Current licence* (see Licence information[3]).
- **Network:** The Webdesk server must be able to reach the provider endpoint (default: https://api.openai.com/v1) via HTTP(S).
- **Credentials:** An API key of the provider, the model name (for Azure OpenAI: the deployment name) and, for providers other than OpenAI, the base URL.
- **Permissions:** Administrators need the actions ai_showModels.act and ai_showAgents.act (editing via ai_editModel.act and ai_editAgent.act); users need the action ai_assistant.act. Permissions are granted like for any other action.

Menu and actions

The administration menu contains three entries under *Modules* > *AI Administration*:

Menu entry	Action	Purpose
AI Models	ai_showModels.act (editing: ai_editModel.act)	List of models (columns Name, Provider, Model ID, Base URL, Client); button "New AI Model".
AI Agents	ai_showAgents.act (editing: ai_editAgent.act)	List of agents (columns Name, Description, Model ID, Client); button "New AI Agent".
Assistant	ai_assistant.act	The chat itself (description "AI Assistant chat"). Grants access to the REST interface /rest/ai/... for GET and POST.

The lists show entries of the accessible clients as well as the *public* entries (no client set, displayed as "Public"). The link for editing public entries is only shown to users who may view the data of all clients.

Icon in the side panel

The assistant icon (tooltip "Assistant") appears in the header only if the action ai_assistant.act exists and the user is authorized for it. Clicking it opens or closes the side panel, which can be pinned (from a window

width of 1200 pixels), minimized and opened in full screen. If the icon is missing, the permission or the licence ai is missing.

Setup

1. **Create a model:** *AI Models* > "New AI Model" (fields see next section) and save.
2. **Optionally create an agent:** *AI Agents* > "New AI Agent", select the model and, if required, define the system prompt and tools.
3. **Alternatively set global defaults:** Store the API key, model and base URL as environment variable, JVM parameter or system parameter AiOptions (order see below). These defaults apply if no agent or no model is used, and they replace an empty API key in the model.
4. **Grant permissions:** Give the users or roles the action ai_assistant.act as well as the actions of the modules whose tools are to be used (see tool catalog).
5. **Test:** Open the panel as a user and ask a question, e.g. about the remaining vacation. The replies for error cases are listed in the section *Troubleshooting*.

Model ("AI Model")

A model describes how Webdesk addresses a language model of a provider. Name, Provider and Model ID are required fields.

Field	Description	Possible value(s)
Name	Designation of the model in lists and selection dialogs.	text, required
Client	Binds the model to a client. Without a client the model is public and available to all clients. An agent can only use models that are public or belong to its own client.	client or empty (Public)
Provider	Type of interface. "OpenAI-compatible" (stored as OpenAi) uses the OpenAI chat interface, i.e. also other services with a compatible interface (also via http://; HTTP/1.1 is used). "Azure OpenAI" (stored as AzureOpenAi) uses the Azure interface. If "OpenAI-compatible" is selected but the base URL contains openai.azure.com or /openai/deployments/, Webdesk automatically switches to Azure OpenAI.	OpenAI-compatible / Azure OpenAI, required
Model ID	Identifier of the model at the provider. For Azure OpenAI this is the deployment name; if the base URL contains a section /openai/deployments/<name>, this name takes precedence over the Model ID (warning in the log).	text, required, e.g. gpt-4o

Base URL	Address of the service. Empty: the global default applies (see order), finally https://api.openai.com/v1. A trailing /chat/completions is removed. For Azure OpenAI the base URL is mandatory; the API version is taken from the api-version parameter of the URL, and a host ending in .cognitiveservices.azure.com is rewritten to .openai.azure.com.	URL (default: empty)
Routing preference	Request to routing services, sent as parameter preference. It only takes effect if the base URL contains cortecs.ai; for other endpoints the value is ignored (warning in the log). For Cortecs endpoints Webdesk additionally sends include_metadata and disables parallel tool calls.	Default (empty) / Balanced (balanced) / Cost optimized (cost) / Speed optimized (speed)
API Key	API key of the provider, entered in a password field. The value is used directly as the key (it is not a reference to an external vault). Empty: the global default applies (see order).	text, e.g. <your-api-key>

Note: In the code, the model field "API Key" is not encrypted when saved. The system parameter AiOptions.defaultAiApiKey, in contrast, is stored encrypted and displayed as a password in the mask. For installations with a single provider it is therefore advisable to store the key there (or as an environment variable) and leave the model field empty.

Agent ("AI Agent")

An agent defines which model the assistant works with, which additional instructions it receives and which tools it may use. Name and Model are required fields.

Field	Description	Possible value(s)
Name	Display name. It appears in the agent selection of the chat and determines the order (alphabetical).	text, required
Client	Binds the agent to a client. Without a client the agent is public. A user only sees and uses agents of his or her client and public agents. The selection dialog for the model shows public models and, if a client is selected here, the models of that client; without a client only public models.	client or empty (Public)

Description	Free text for documentation (maximum 1024 characters). It is shown in the agent list but not sent to the language model.	text
Model	The "AI Model" to use. If the model belongs to a different client than the user, the global default is used instead (warning in the log).	selection, required
System Prompt	Additional instructions for the language model (maximum 4000 characters). They are appended to the standard prompt generated by Webdesk and do not replace it.	text
Allowed Tools	Two-column selection "Available tools" / "Selected tools" containing all tools registered in the system. The selection restricts the tools offered to the model. If the selection is empty, there is no restriction: all tools that the user may use according to his or her permissions are offered. The tool names are stored comma-separated (maximum 4000 characters).	tool names, empty = all permitted

Which agent is used?

- If the user selects an agent in the agent selection of the chat, it is used provided it is public or belongs to the user's client. Otherwise Webdesk falls back to the *default agent*: the alphabetically first agent of the client, otherwise the alphabetically first public agent, otherwise the first accessible agent.
- If the selection stays at "Default agent (auto)" (empty value), Webdesk uses an internal default assistant without a model and without tool restriction. Model, base URL and key then come exclusively from the global defaults, and all tools are available that the user may use according to his or her permissions. If the call fails in this mode with "model not found" (HTTP 404), Webdesk repeats it once with the default agent of the client.

Warning: The tool selection of an agent is not a security boundary. Any user can choose "Default agent (auto)" in the agent selection, which removes the restriction of the agent. The action permissions are decisive for security (see authorization checks).

Order of the settings (fallback)

Webdesk determines API key, model name and base URL for every request. The first non-empty value of the respective list applies. Regardless of the provider, the environment variables and JVM parameters are called OPENAI_API_KEY, OPENAI_MODEL and OPENAI_BASE_URL, and they take precedence over the system parameter AiOptions.

Setting	Order (first non-empty value wins)	Result if nothing is set
API key	1. Field "API Key" of the agent's model 2. Environment variable OPENAI_API_KEY	No call to the provider. The chat replies "AI chat fallback is active; no API key configured."

	3. JVM parameter -DOPENAI_API_KEY=<your-api-key> 4. System parameter AiOptions.defaultAiApiKey	
Model name	1. Field "Model ID" of the agent's model (for Azure with a deployment in the URL: the name from the URL) 2. Environment variable OPENAI_MODEL 3. JVM parameter -DOPENAI_MODEL=... 4. System parameter AiOptions.defaultAiModel (default: gpt-4o) 5. built-in value gpt-4o	gpt-4o
Base URL	1. Field "Base URL" of the agent's model 2. Environment variable OPENAI_BASE_URL 3. JVM parameter -DOPENAI_BASE_URL=... 4. System parameter AiOptions.defaultAiBaseUrl (default: empty) 5. built-in value https://api.openai.com/v1	https://api.openai.com/v1

The field of the model is only considered if the chat uses an agent with an accessible model. In all other cases ("Default agent (auto)", model of a different client) step 1 is skipped and the provider "OpenAI-compatible" and the global defaults are used. If the model contains a model name but no key, the key is determined by steps 2 to 4.

System parameter AiOptions

The bean AiOptions contains the global defaults. They only apply if neither the model nor an environment variable or JVM parameter supplies a value.

Property	Description	Possible value(s)
defaultAiApiKey	Default API key used as fallback when no environment variable or JVM parameter is configured. Stored encrypted and displayed as a password in the mask.	text (default: empty)
defaultAiModel	Default model identifier used as fallback when no environment variable or JVM parameter is configured.	text (default: gpt-4o)

defaultAiBaseUrl	Default base URL used as fallback when no environment variable or JVM parameter is configured. Empty means https://api.openai.com/v1.	URL (default: empty)

Tool catalog

Tools are defined in the program and cannot be created in the user interface. They are provided by the modules Portal & Organization (PO), Time & Attendance (TA) and Project Time Management (PTM). All tools run on behalf of the logged-in user: the "getMy..." tools return only his or her own data, the other tools only data that he or she may see according to the checks described below. Tools that change data are marked "Writing".

Portal & Organization

Tool	Purpose	Required action (one is sufficient)
searchColleagues	Searches colleagues in the client by search term (name, employee ID, short name, organizational unit).	po_showPersons.act, po_searchPerson.act
findPersonsByName	Searches persons by full or partial name.	po_showPersons.act, po_searchPerson.act
getAllGroups	All groups of the client (organizational units, cost centers, locations, billing circles).	po_showGroups.act
searchGroups	Searches groups by name or short name.	po_showGroups.act
getPersonsAge	Age of a person (parameter personUid).	po_showPersons.act
getPersonsManager	Manager of a person (default role Manager).	po_showPersons.act
getPersonsCompetence	Persons in the area of responsibility of a person (also via their groups).	po_showPersons.act
getPersonsGroups	Groups a person is assigned to.	po_showPersons.act, po_showGroups.act, po_searchPerson.act
getPersonsOfGroup	Members of a group (parameter groupUid).	po_showGroups.act, po_showPersons.act, po_searchPerson.act

Time & Attendance

Tool	Purpose	Required action (one is sufficient)
getMyOpenVacationDays	Open vacation days of the current vacation year.	ta_getJournalV2.act

getMyYearlyVacationDaysEntitlement	Yearly vacation entitlement in days.	ta_getJournalV2.act
getMyPreviousVacationIntervals	Vacation intervals of the last 12 months.	ta_getJournalV2.act, ta_doBooking.act
getMyWeeklyWorkingTime	Hours already worked in the current week.	ta_getJournalV2.act
getMyWeeklyTargetTime	Weekly target hours according to the time model.	ta_getJournalV2.act
getBookingsInTimeRange	Own bookings of one day (parameter day).	ta_getJournalV2.act
getMyTodayBookings	Own bookings of today.	ta_getJournalV2.act
getAbsenceIntervals	Approved past and planned own absences in a period (parameters from, to).	ta_getJournalV2.act, ta_getCalendar.act, ta_yearlyCalendar.act
listUpcomingAbsences	Own approved future absences from today (parameter daysAhead, default 90, maximum 3650 days).	ta_getJournalV2.act, ta_getCalendar.act, ta_yearlyCalendar.act
getTimeEntryReasons	All time entry reasons of the client that can be used for booking (work and absence).	ta_doBooking.act, ta_getAttendanceTableau.act
getAttendanceStateOfColleague	Attendance status of a person (absent, present, absent with reason, not booking) including the last booking message (parameter personUid).	ta_getAttendanceTableau.act
doBooking (Writing)	Performs an automatic attendance booking (clock in/out) for the user with the chosen time entry reason (parameter timeEntryReasonId). After a successful booking the page reloads in the background. Webdesk does not ask for confirmation before the call.	ta_doBooking.act

Project Time Management

Tool	Purpose	Required action (one is sufficient)
getMyMonthlyEffortSummary	Monthly summary of own project times per project (parameters month, year).	ptm_editMonthlyTimeEfforts.act, ptm_editTimeEfforts.act
listMyOpenProjectAssignments	Projects on which the user may book and/or change bookings.	ptm_editTimeEfforts.act, ptm_doProjectTimeBooking.act, ptm_editMonthlyTimeEfforts.act
searchProjects	Searches projects by short name, name or code and returns	ptm_showProjects.act, ptm_chooseProjectDialog.act, ptm_editTimeEfforts.act

	only projects the user may use (optionally only bookable ones).	

The only writing tool is doBooking. All other tools are read-only.

Authorization checks for tools

1. **Action permission (offering):** Each tool names one or more required actions. A tool is only offered to the language model if the user is authorized for at least one of them; permissions on derived configurations of the action count as well. If a tool refers to an action that does not exist in the system, that action is skipped (warning in the log); if no known action remains, the tool is blocked. Tools of modules whose actions are not present or not authorized are therefore not available.
2. **Restriction by the agent:** The list is then reduced to the agent's "Allowed Tools" (only if an agent is selected and the list is not empty).
3. **Target check (execution):** When a tool is called, Webdesk checks all arguments whose name ends in uid or uids (e.g. personUid, groupUid) for permission on the object concerned. For every UID passed, the user must be authorized for at least one of the required actions for exactly this object (permission on object level or a matching configuration of the action for this person). Otherwise the model receives the error message "Tool not permitted for requested target." and the tool is not executed.

The tools are thus bound to the permissions of the user. Errors and rejections are part of the tool display in the chat (Arguments / Result).

Interfaces (REST)

The chat user interface communicates with endpoints under /rest/ai. They always work on behalf of the logged-in user.

Endpoint	Purpose	Notes
POST /rest/ai/chat	Sends a message (optionally with agent, conversation and page context) and returns the reply, tool calls and UI effects.	Empty messages are rejected.
GET /rest/ai/agents	Agents of the client and public agents, alphabetically.	Fills the agent selection.
GET /rest/ai/mentions	Persons of the client for mentions via @username.	Only user name, name and UID.
GET /rest/ai/conversations	Own conversations (parameters offset, limit; default 50, maximum 200).	Own conversations only.
GET /rest/ai/conversations/{id}/messages	Messages of an own conversation (default 200, maximum 1000).	HTTP 404 if the conversation does not belong to the user.
POST /rest/ai/conversations/{id}/title	Renames an own conversation (maximum 255 characters).	
GET /rest/ai/mcp/tools	List of the tools (name, description, parameters) that the logged-in user may use according to his or her action permissions.	Agent restrictions do not apply here.
POST /rest/ai/mcp/tools/{name}	Executes a tool directly. Body: {"args": {...}} or the arguments	HTTP 404 (unknown tool), 403 (action or target permission missing), otherwise 400.

	directly as a JSON object. Response: {"tool": ..., "result": ...}.	

The mcp endpoints are a simple MCP-style interface and use the same authorization checks as the chat, but without agents and without a language model: doBooking can also be called there. For internal calls from the user interface, the REST access rights of the action ai_assistant.act (/rest/ai/*, GET and POST) are used; external callers (e.g. with an API key) need role-based REST access rights.

Conversation history

- Each conversation belongs to exactly one user and his or her client; other users cannot retrieve it via the interfaces. There is no administration mask for conversations.
- Stored are the conversation (title, owner, client, selected agent and model, timestamps) and, per message, role (user, assistant, tool result), text, tool name, tool arguments, tool result (JSON), error text, agent, model and timestamp (table AiConversationMessage). Tool results therefore also contain the data that was queried, e.g. names and e-mail addresses of colleagues.
- The title is built from the first 80 characters of the first question and can be renamed.
- For every message the chat calls the language model with the previous history (last 30 messages; older parts are removed when an estimated size of about 3500 tokens is exceeded). If an existing conversation is restarted with the flag "fresh" (menu "Regenerate fresh" in the chat), its stored messages are deleted.
- The code has no automatic deletion or retention period and no function to delete whole conversations. Cleanup, if required, must be done in the database.
- Error replies of the chat (see *Troubleshooting*) are also stored as assistant messages in the history.

Data protection: what leaves the system?

With every chat request Webdesk sends the following to the configured endpoint (base URL of the model or of the global default):

- The **standard prompt** with the current time, the user name and rules for using the tools, extended by the agent's **System Prompt**.
- **Information about the logged-in user:** UID, user name, first and last name, employee ID, validity (from/to), download address of the person photo, language, TA ID, time zone, e-mail address, active flag and short name of the organizational unit.
- The **page context** of the currently opened page: action name, menu path, page title, full page URL, action type, process definition as well as the object ID and URL parameters of the page. The user can display this context in the panel via the eye icon ("Show shared context").
- The **messages of the user** including the conversation history (see above) and the **results of all tools called** (e.g. personal data of colleagues, bookings, absences, vacation days, project times) as well as the **descriptions and parameters of the tools offered**.

The API key serves exclusively as credentials for the configured endpoint. Processing at the provider is governed by the provider's terms and lies outside Webdesk; this page makes no statement about it.

In the server log (logger of the class at.workflow.webdesk.ai.impl.AiChatServiceImpl) the user name, the chosen provider with model, base URL and agent name, as well as the tool calls with arguments and results (each truncated to 500 characters) appear at level INFO. The API key is not logged.

Ways to restrict

Measure	Effect
Do not grant the action ai_assistant.act	The user does not see the icon and has no access to the chat interface. Most effective control per user group.

Do not purchase or install the licence ai	The module is not available.
Do not store an API key (neither in the model nor as environment variable, JVM parameter or system parameter)	No call to an external provider takes place; the chat only replies with the error message.
Withdraw the action permissions of the tools (e.g. ta_doBooking.act, po_showPersons.act)	The corresponding tools are not offered to the model and are also blocked via the interface. This is the binding restriction.
Use agents with a tool selection	Reduces the tools offered and thus the data transmitted for users who use the agent. Does not apply if the user chooses "Default agent (auto)".
Client binding of models and agents	Limits which clients use which provider or access.
Data cleanup	Stored conversations remain until they are removed in the database.

Troubleshooting

Symptom	Cause and check
No assistant icon in the header	Permission for ai_assistant.act is missing, or the licence ai is missing or the module was not loaded.
"AI chat fallback is active; no API key configured."	No API key is set at any stage of the order. Note that the environment variable and JVM parameter come before the system parameter, and that in the mode "Default agent (auto)" the model field is not considered.
"AI chat fallback is active; error during model invocation."	The call to the provider failed (wrong key, model name, base URL or deployment, network, missing base URL for Azure OpenAI). The cause is in the server log ("AiChatService error"); the line "AI model resolved" shows provider, model, base URL and agent.
"Model returned no text and no tool calls..."	The model or provider does not support tool calls for this request. Choose a different model or reduce the tools in the agent.
An expected tool is not used	The action permission for the tool is missing, the tool is not among the "Allowed Tools" of the selected agent, or the target check rejects the UID ("Tool not permitted for requested target.").
Routing preference has no effect	The base URL does not contain cortecs.ai; the value is ignored.

1. /daisy/personalwolke-admin/11418-dsy.html?language=4
2. /daisy/personalwolke-admin/5122-dsy.html?language=4
3. /daisy/personalwolke-admin/5524-dsy.html?language=4