

Project templates

Start the next, similar project from the structure of one you already ran - milestones, tasks, dependencies, subtasks and their timing - instead of planning it from an empty page.

Overview

No two projects are identical, and a template does not pretend they are. It is a **starting skeleton** for work that comes back in a similar shape: a customer roll-out, a release, a yearly audit, an onboarding, an event. You start from it, delete the steps that do not apply and add the ones that do. What you save is the *forgetting*: the steps that are always there.

A template keeps	It does not keep
Project settings (name, description, type) and duration Milestones, with their dates as days from the project start , and which milestone blocks which Tasks with title, description, type, estimate, story points, priority, milestone, due and deadline as day offsets Dependencies and subtasks Custom field values Each task's first external comment, as its brief	Costs and budget Owner, project manager, customer, solution, program Assignees and the project team's members Other comments, booked time, attachments Status and done flags Private tasks, deleted tasks, reservations, scheduled (recurring) tasks

A template is a snapshot taken when you save it; it does not follow the project afterwards. It cannot show up in project lists, the Gantt chart, capacity or reports.

1. Save a project as a template

- 1 Open the project and click **Save as template** (next to Report). You see it if you may start projects and may change this project.

- 2 The dialog proposes the project's name. Change it and say what the template is for.
- 3 It shows what the template will hold, for example *3 milestones · 12 tasks · 2 subtasks · 7 dependencies*, and warns when the project has no dates (the template then cannot shift in time), no tasks yet, or private tasks that were left out.
- 4 Click **Save**.

A finished project is the best source: its structure is proven. All its tasks are copied; status and booked time are not.

2. The template list

Open **Project templates** from **My Projects**. Everyone who may start a project sees every template. **View** shows the milestones and tasks with their day offsets and the teams involved. **Start a project** opens the new-project page with that template. **Rename** and **Delete** are for the person who saved it, Project Admins and RC Admins. Deleting a template does not touch the projects made from it.

A template's contents cannot be edited. To change one, change a project and save it again (under the same or a new name), then delete the old one.

3. Start a project from a template

On the **New Project** page, the card **Start from a template** is at the top.

- 1 Choose the template. The page shows what it holds and how many days it spans, takes over its project type, and fills in an empty description from it.
- 2 Enter the name, the **project code** (5 characters, not used by another project) and the **period** as always. **The start date decides every date**: a task that was 9 days after the old project's start is 9 days after the new one's. The end date you enter is used as typed.
- 3 **Include the tasks** is on by default. Switch it off to get the settings and the milestones only.

- 4 **Move dates that fall on a weekend to Monday** is on by default. A date computed as Saturday or Sunday becomes the Monday after.
- 5 **Team mapping** (with tasks): for each team the template's tasks were in, choose which of *your* teams takes them over. A team you can reach is pre-selected. The old project's own team can become **the new project team** (agile projects only). Every row must be answered.
- 6 Create the project. You land on its page.

What you get: a project in **Planning**, owned by you, with the milestones (and their blockers) on the project's epic. With tasks: each in its milestone, with dependencies and subtasks; unassigned in the team's pool, or with the team lead where the team has no pool; **inactive until you kick the project off**, as for any project task. Each task's brief becomes an **internal** comment (never customer-visible, because the customer is a different one now). Custom field values are filled in where the receiving team has a field with the same key; what cannot be placed is counted in the project log. The log also says which template the project came from.

Costs are not part of a template: enter budget and costs on the project itself.

It is all or nothing. If anything is wrong - a team without an answer, a code in use, a date that does not exist - you get the reason and nothing has been created.

4. Making good templates

- **Clean up before saving.** Delete the one-off tasks; keep the steps that always come back. The template is exactly the project's structure at that moment.
- **Name teams the same way** in the projects you want to template. The mapping is per team, so fewer, clearer teams make it quicker.
- **Dates are relative.** Give the source project a start date and put due dates on its tasks and milestones; a template without dates only has the order and the dependencies.
- **Custom fields** travel as values, so the receiving team needs the same field keys (Team Settings > Custom fields) to receive them.

5. REST API and MCP

Call	What it does
GET /project-templates	the templates, without contents
GET /project-templates/{id}	the contents with day offsets, and the teams that need an answer
POST /projects/{id}/save-as-template	save a project: {"name": "...", "description": "..."}
PUT / DELETE /project-templates/{id}	rename / delete
POST /projects with template_id	make a project from a template

```
curl -u USER:TOKEN -X POST https://YOUR-INSTANCE/api/v1/projects \
-H 'Content-Type: application/json' \
-d '{"name": "Roll-out Acme", "template_id": 4, "start_date": "2027-09-10",
    "project_code": "acme1", "solution_id": 3,
    "team_map": {"tm12": 7, "tm15": "project_team"}}'
```

The keys in `team_map` (`tm12`) are the template's team keys from `GET /project-templates/{id}` ; the values are team ids you may use, or `project_team` . Optional: `end_date` , `type` (`agile` or `waterfall`), `members` , `include_tasks` (default true), `weekend_roll` (default true), `custom_fields` for the project. The answer is the project plus `from_template` : what was made and which custom field values were left out. The hosted MCP endpoint has the same as tools (`list_project_templates` , `get_project_template` , `save_project_as_template` , `update_project_template` , `delete_project_template` , and the template arguments of `create_project`). See the MCP guide. A read-only token can read templates but not save one or make a project.

Rules and limits

- Who: the roles that may start projects (Project Lead, Project Supervisor, Project Admin, RC Admin) use every template; there is no per-team scoping. To save a project you also need write access to it.
- A template holds at most 2,000 tasks. Epics other than the project's own, and releases, are not copied.
- The project type decides whether a project gets a project team of its own: **Agile** does, **Classic** works with existing teams. You can choose it when you start from a template.
- Templates are not part of reports or exports.