

Class 6 • Project Studio

Take-home notes • One workspace with connected tools

What we are building

Project Studio will help you collect possibilities and organize future work. Overview, Idea Lab, and Projects share one backend and database. Task boards, journals, and mind maps have clearly labeled preview pages for later lessons.

Weekly capstone homework begins after winter break in January 2027. You do not need to choose a capstone topic now. An idea can describe a useful problem, its audience, and one possible feature.

Make it mine

Use Exercise Prompt 2 to choose a theme, colors, layout, light/dark mode, or gentle animation. Keep one consistent design across routes. If you add a theme toggle, save its preference after reload; that display preference can use browser storage while ideas remain in D1. Respect reduced-motion preferences and test readable text, keyboard navigation, a narrow window, and existing forms.

The parts of the system

- Route: an address for a screen, such as /ideas. Refreshing that address should still open the correct screen.
- Frontend: the browser page and controls. HTML supplies structure, CSS appearance, and JavaScript behavior.
- Backend: the Worker handles requests, checks data, and talks to the database. An API is the set of requests the frontend can make.
- Database: D1 stores related records. A table groups one kind of record; a row is one record; a column is a field.

A saved idea's journey

1. You submit the form. Browser JavaScript sends a POST request to /api/ideas.
2. The Worker checks the fields and uses a bound SQL INSERT to store a row in D1.
3. The browser requests updated information. Idea Lab and Overview both read the same database through the backend.

A localStorage-only imitation would skip this backend/database journey. In our app, local D1 data survives reload and a Wrangler restart in the same folder. The dashboard refreshes on navigation or reload; it is not live multiplayer synchronization.

IDs connect resources

Each idea has a primary key: ideas.id. A project's source_idea_id is a foreign key referring to that idea. A JOIN follows the ID to show the current source title. Renaming the source keeps the relationship; the project's own title stays independent.

In this starter, one idea can become at most one project. Deleting the idea preserves the project and clears the source link (ON DELETE SET NULL). Future tasks and journal entries will belong to projects by project_id.

Your Cloudflare development tools

Workers runs the backend. D1 supplies the relational database. Wrangler installs inside this project and runs the local development environment. Today's local work needs no Cloudflare sign-in. Node 22+ is required; Node 24 LTS is preferred.

First launch, inside the extracted folder containing package.json. Run each command separately:

```
npm ci
npm run types
npm run db:migrate
npm run db:seed
npm run dev
```

Returning later: run `npm run dev` in the same folder. Keep its terminal open. Use the actual printed URL, normally `http://127.0.0.1:8787`. Stop your own server with `Ctrl+C`.

Local Explorer

Open the server origin plus `/cdn-cgi/local/explorer`. Choose D1 Databases > DB > ideas to browse real rows. The SQL editor can run this read-only query:

```
SELECT id, title, audience FROM ideas;
```

Terminal fallback: `npm run db:list`. Local Explorer can also inspect Workflows when configured. This starter has no Workflow, so an empty Workflows section is expected.

When something goes wrong

- Connection refused: check that Wrangler is running and use its printed address.
- Missing table: check the project folder and run `npm run db:migrate`.
- Wrong or stale result: inspect the browser request, backend response, and actual database row. Write the action, expected result, and actual result.

Local records live under `.wrangler/state`. Keep that folder; Git and the starter ZIP do not copy your personal database. The seed command adds examples and may restore deleted sample ideas. Remote D1 will be a separate database in a later deployment lesson.

Stretch and next steps

Try a linked-project rename test, JSON export, another editable field, or a backend test. Explain how your extension uses the existing system. We will add Git checkpoints and more linked tools in later classes. Continue the standing weekly ChatGPT/Claude update check.

This is a local single-user prototype. Use fictional examples. Public accounts, ownership rules, and shared editing belong to a future lesson.

[Cloudflare reference: Local Explorer](#)

[Cloudflare reference: D1 local development](#)