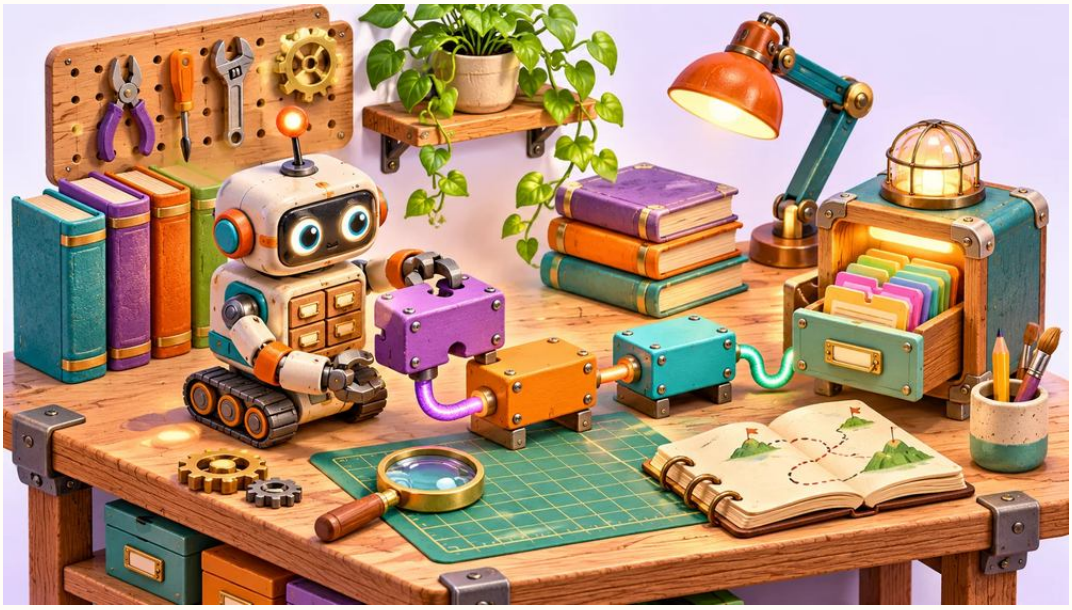
MCP Builder Lab

Build a resource-finder tool, test the protocol and explain your decisions.



A complete local Python project with a fictional learning catalog, transport tests, platform-specific setup examples and teaching resources. Use the worksheet to plan your own extension and capture reproducible evidence.

Use this kit to make a concrete plan, rehearse one workflow and capture what you learn. The detailed article and editable templates accompany this compact guide.

Open the full article

https://codeteachers.com/blog/build-your-first-mcp-resource-finder/

The implementation sequence

01 / Set up the environment

Use the pinned requirements and a Python 3.12 virtual environment. Run the included test before changing the project.

02 / Read the contract

Predict the result for normal, empty and invalid requests. Match those predictions to the input schema and function.

03 / Inspect the server

Follow registration, strict validation, catalog filtering and the structured return. Keep diagnostics off standard output.

04 / Test through MCP

Run discovery and calls through the real stdio client. Separate function errors from transport or environment errors.

05 / Connect and explain

Use the platform-specific VS Code example. Compare the assistant's summary to actual tool inputs and outputs.

06 / Extend with evidence

Write acceptance cases for one extension, implement it and record the change in the project brief and results journal.

A completed example

Learner and task

A beginner has 20 minutes and wants a Python exercise.

Tool name and promise

find_resources searches fictional titles and filters by study time; never enrolls.

Inputs and bounds

topic: nonblank string up to 80 characters; max_minutes: integer 1-120, default 30.

Structured output

demo, count, resources with ID, title, minutes and format.

Empty result behavior

count: 0 and resources: []; no invented recommendation.

Review the example as a learning aid. It is not evidence that a customer agreed, a deployment succeeded or a result was measured.

The example, continued

Invalid request behavior

A blank topic or string time budget returns a tool error.

One next feature

Add an enum filter for exercise, lab or project.

Reproduction evidence

Python 3.12; mcp 1.29.0; run test_resource_server.py; capture final PASS line.

Portfolio explanation

I preserve empty results so the assistant cannot quietly exceed the learner's time budget.

Review the example as a learning aid. It is not evidence that a customer agreed, a deployment succeeded or a result was measured.

Your working worksheet

01 / Learner and task

Describe the person and the decision this tool supports.

02 / Tool name and promise

Use an action name and a precise description.

03 / Inputs and bounds

Include types, limits and defaults.

Your working worksheet

04 / Structured output

List the stable fields the caller can inspect.

05 / Empty result behavior

Explain what a successful search with no match returns.

06 / Invalid request behavior

Choose an input that should fail explicitly.

Your working worksheet

07 / One next feature

Keep the next milestone narrow enough to test.

08 / Reproduction evidence

Record interpreter, dependency version and test command.

09 / Portfolio explanation

Describe a design decision rather than just listing technology.

Use the kit, then verify the result

1. Keep a working copy

Extract kit.zip. Open index.html in your browser, or edit the Markdown templates in any text editor. The interactive worksheet keeps notes in page memory only. Download notes before leaving.

2. Make the example your own

Replace every fictional person, source, date and assumption with your own confirmed context. Record unknowns explicitly instead of filling them with plausible guesses.

3. Rehearse a normal and an awkward case

Try one known input, one missing input and one result that requires a reviewer. Check the delivered artifact against its source. Keep the manual path usable.

4. Record a dated decision

State what passed, what failed and what you will change next. Count setup, checking and rework when reviewing effort. A completed worksheet is preparation, not measured business improvement.

Sources and next steps

MCP architecture

https://modelcontextprotocol.io/docs/learn/architecture

WebMCP browser API (experimental)

https://developer.chrome.com/docs/ai/webmcp/imperative-api

Your detailed companion article

https://codeteachers.com/blog/build-your-first-mcp-resource-finder/

Independent educational materials. Product names belong to their owners. These templates may be adapted for your own projects and client work.