Connect Cursor to the Functionize MCP Server

Model Context Protocol (MCP) · https://docs.functionize.com/hc/en-us/articles/43755032212631-Connect-Cursor-to-the-Functionize-MCP-Server

Cursor connects to the Functionize MCP server directly. It speaks Streamable HTTP and handles OAuth itself, so there is no bridge to install and no Node.js requirement.

One click

If Cursor is installed on this machine, this link adds the server for you:

Add Functionize to Cursor

Cursor opens, asks you to confirm, and writes the entry itself. Then go to Sign in. If you would rather add it by hand, the file-based route follows.

Add the server by hand

Create or edit one of these files:

Just this project	`.cursor/mcp.json` in the project root
Everywhere	`~/.cursor/mcp.json` in your home directory On Windows that is `%USERPROFILE%\.cursor\mcp.json`

Add the server:

```
{
  "mcpServers": {
    "functionize": {
      "url": "https://mcp.functionize.com/mcp"
    }
  }
}
```

A remote server needs only `url`. Cursor works out that the server is remote from its presence, so nothing else is required.

Restart Cursor after editing `mcp.json` by hand. Cursor's own documentation says to, and a server added to the file while Cursor is running will not appear until you do.

You can also add it through the interface: open the sidebar, choose **Customize**, and use the MCP management screen. Servers have an enable/disable toggle; a disabled server will not load or appear in chat.

Optional: pin a team

If you belong to more than one Functionize team and want this connection to act as a specific one:

```
{
  "mcpServers": {
    "functionize": {
      "url": "https://mcp.functionize.com/mcp",
      "headers": {
        "X-Functionize-Team-Id": "12345"
      }
    }
  }
}
```

Replace `12345` with your team's numeric ID. Omit the header to use **your default team**; switching teams inside Studio does not move the MCP connection. Your team ID is not a secret, so it is fine inline — but if you ever add a genuine credential to this file, use Cursor's `${env:NAME}` interpolation rather than writing it in plain text.

Sign in

Cursor marks the server as needing authentication rather than signing in on its own. Open the MCP management screen, find **functionize**, and click the authentication prompt beside it. Your browser then opens.

Sign in with the email and password you use for Functionize Studio. If you registered with Google, set a password via **Forgot password** on the Studio login page first. If your company signs in to Functionize through SSO, contact your Functionize representative.

There is no separate MCP account or password.

Cursor registers itself with our server automatically, so there is nothing to configure on our side and no client ID to request.

The sign-in page is hosted at `functionize.us.auth0.com`. If you are ever asked for Functionize credentials on another domain, do not enter them.

Check it worked

Ask Cursor to list your Functionize agent sessions. If your team's sessions come back, you are connected. The list is team-wide, so on a shared team you will see colleagues' sessions too.

Ten tools should now be available, covering starting sessions, sending messages, streaming events, attaching files and stopping sessions.

If it does not connect

Check the file is valid JSON. A stray comma after adding the entry is the most common cause of a server that never appears.

Restart Cursor. Cursor's documentation says to restart after editing `mcp.json`, and a server added while Cursor is running will not appear until you do.

Check the server is not waiting on you. Cursor flags a server that needs authentication rather than opening the browser by itself. Look for the prompt on the MCP management screen and click it.

If the team header seems ignored, confirm the connection works without it first, then check with a call rather than with `list_agent_teams` — that tool echoes a pinned team ID back without checking it. List sessions and see whether they are the ones you expect for that team.

For anything else, see Troubleshooting the Functionize MCP Connection.

A note on the Windows path

Cursor's documentation describes the global config as living "in your home directory" without giving a Windows path. `%USERPROFILE%\.cursor\mcp.json` is the standard equivalent and is what we have seen work, but if it does not, use a project-level `.cursor/mcp.json` instead, which is documented explicitly.

Related

- Getting Started with the Functionize MCP Server — what the server can do, once connected
- Connect Any MCP Client to Functionize — the endpoint, scopes and transport in one table
- Troubleshooting the Functionize MCP Connection — when it does not connect
- Exercise 7 - Drive Studio From Your Own AI Tool — ten minutes, against the demo store
- The Independent Tester Principle: Never Build and Test in the Same Session — why this belongs in a session of its own