eyal-gor / p_69_branch_monkey_mcp Public

Code Issues 1 Pull requests 5 Actions Projects Security and quality Insights

main 9 Branches 0 Tags Go to file Go to file Code

2 people and Eyal Goren Relay: retry transcript push on 404 / 5xx / network with bounded backoff 69bc718 · last week

.branch	ve data.db fro acking (already in ….	4 months ago
br	transcri sh on 404 / 5xx / n…	last week
	nat edup into _post_tr…	last week
	_version.py so isol…	2 weeks ago
	n branch-monkey-…	2 months ago
	ompute stats, and …	2 months ago
	ate olling GitHub for ne…	2 weeks ago
in	eway UR gateway.cerver.ai	3 weeks ago
install.sh	.sh: pass --refl to uvx so restarts …	2 weeks ago
pyproject.toml	Relay: bake version into wheel via hatch bu…	2 weeks ago
setup-infisical.sh	setup-infisical.sh: read from /dev/tty so curl-…	last week
uv.lock	Refresh uv lockfile	2 months ago

README

Kompany MCP Server

MCP (Model Context Protocol) server for Kompany - connecting Claude Code to your task management and team collaboration platform.

Quick Start

Add this to your project's `.mcp.json`:

```
{
  "mcpServers": {
    "kompany-cloud": {
      "command": "uvx",
      "args": ["--from", "git+https://github.com/gneyal/p_69_branch_kompany_mcp.git", "branch-monkey-mcp"],
      "env": {
        "BRANCH_MONKEY_API_URL": "https://kompany.dev"
      }
    }
  }
}
```

Restart Claude Code. On first use, a browser will open for you to log in and approve the device.

Using It With Codex

Codex does not consume MCP servers the same way Claude Code does, so the integration path is different:

- Use the local bridge and workflow CLI instead of `.mcp.json`
- Set the default CLI provider to `codex`
- Authenticate Codex via `OPENAI_API_KEY` or `codex login`
- Keep the MCP server as an optional adapter for Claude Code

In this repo, the non-MCP path already exists:

- The Kompany API logic is plain Python/HTTP, not MCP-specific
- The local agent server supports `cli_tool: "codex"`
- `kompany-workflow llm` can route prompts through the selected CLI provider

That means the practical Codex adaptation is: treat MCP as one frontend, not the core architecture.

Features

- **No API key needed** - Authenticates via browser approval
- **Automatic token storage** - Saved in `~/.branch-monkey/token.json`
- **Task management** - Create, update, track tasks
- **Team collaboration** - Share tasks with your team
- **Prompt tracking** - Associate prompts with tasks

Available Tools

Status & Auth

- `kompany_status` - Get connection status
- `kompany_login` - Force re-authentication (use if having auth issues)
- `kompany_logout` - Clear auth token

Projects & Organizations

- `kompany_project_list` - List all projects in your organization
- `kompany_project_focus` - Set focus to a specific project
- `kompany_project_clear` - Clear project focus
- `kompany_org_list` - List organizations you have access to

Tasks

- `kompany_task_list` - List all tasks
- `kompany_task_create` - Create a new task
- `kompany_task_update` - Update a task
- `kompany_task_delete` - Delete a task
- `kompany_task_work` - Start working on a task
- `kompany_task_log` - Log progress on a task
- `kompany_task_complete` - Mark task as complete
- `kompany_task_search` - Search tasks
- `kompany_get_recent_tasks` - Get recently worked tasks
- `kompany_auto_resume` - Auto-detect related tasks

Versions

- `kompany_version_list` - List versions
- `kompany_version_create` - Create a version

Team

- `kompany_team_list` - List team members
- `kompany_team_add` - Add team member

Machines

- `kompany_machine_list` - List machines
- `kompany_machine_create` - Create a machine

Local Server Identification

When you run the relay client (`branch-monkey-relay`), your machine connects to Kompany Cloud and becomes available for executing tasks.

How Machine Identification Works

Each local machine is identified by a unique **machine ID** composed of:

- **Hostname** (e.g., `my-macbook-pro`)
- **Process ID** (e.g., `12345`)

Combined format: `my-macbook-pro-12345`

This ensures that:

- Each machine has a globally unique identifier
- Multiple relay instances on the same machine get different IDs
- The system can route requests to the correct machine

How the "Local" Button Maps to a Computer

Important: "Machines" in the API (`/api/machines`) are business automation processes, not physical computers. Local computers are tracked separately as **compute nodes** in the `compute_nodes` table.

When you click "Local" on a task card, the system routes the request to a compute node:

1. **Compute nodes are per-user** - Each user's connected computers are tracked by `user_id`
2. **Currently: first available** - The system uses the user's connected compute node (if multiple are online, behavior depends on frontend implementation)
3. **Future: explicit selection** - The UI could show a dropdown to select which computer to use

Note: The task's `machine_id` field refers to business process machines, NOT local computers.

Multiple Computers as "Local"

If you have multiple computers signed in as the same user:

1. **Each registers as a compute node** - Stored in `compute_nodes` table with unique `machine_id` (hostname-pid)
2. **Same Supabase channel prefix** - But each has a unique channel: `relay:{user_id}:{machine_id}`
3. **Routing requires machine_id** - The cloud must specify which compute node to send requests to
4. **Status tracking** - Heartbeats every 25 seconds update `last_heartbeat` and `status`

Compute Node Data

Each connected computer is tracked with:

- `machine_id`: Unique identifier (hostname-pid)
- `user_id`: The authenticated user
- `name`: Machine hostname
- `node_type`: "local"
- `status`: "online" or "offline"
- `last_heartbeat`: Timestamp of last heartbeat
- `capabilities`: e.g., `{"claude": true}`

Connection Behavior

- **Same machine_id reconnects**: Upserts (updates existing row)
- **Different process on same host**: Creates new entry (different PID = different machine_id)
- **Machine goes offline**: Status updated to "offline" on graceful shutdown

Current Limitation

The frontend needs to handle the case where a user has multiple computers connected. Options:

1. Show a compute node selector before running on "Local"
2. Default to most recently active compute node
3. Allow users to set a "primary" compute node

Troubleshooting

Authentication Issues

If you're having trouble connecting:

1. **Use `kompany_login`** - Forces re-authentication via browser
2. **Check the token file** - Stored at `~/.branch-monkey/token.json`
3. **Clear and retry** - Use `kompany_logout`, then restart Claude Code
4. **Check network** - Ensure you can access https://kompany.dev

First-Time Setup

On first use:

1. A browser window opens automatically
2. Log in to Kompany (or create an account)
3. Select the organization you want to connect to
4. Approve the device when prompted
5. Return to Claude Code - you're connected!

After connecting, use `kompany_project_list` to see available projects and `kompany_project_focus <id>` to select one.

Requirements

- Python 3.10+
- uv package manager

License

Releases

No releases published

Packages

Contributors 2


claude Claude


eyal-gor Eyal

Languages

Python 85.6% **HTML** 13.4% **Shell** 1.0%