# **Redmine MCP Tool Documentation**

# **1. Architecture Overview**

**How It Works**

The Redmine MCP Tool interacts with Redmine’s REST API to fetch, analyze, and visualize project management data.

**Key Components:**

Redmine REST API: **Fetches** project / issue data (JSON responses).

Dashboard Engine: **Generates** HTML/CSS/JS **Dashboards** with Chart.js.

Authentication Handler: Manages **API** key validation. ( frappe ).

**Data Flow:**

User submits a request (e.g., "Generate redmine dashboard for Project <project name").

Tool queries Redmine API for issues/projects.

Data is processed into metrics (status, priority, assignee distribution).

An interactive dashboard is rendered client-side with dynamic charts.

# **2. Dependencies**

Dependency And Purposes

| **Dependency** | **Purpose** |
| --- | --- |
| Frappe | To run bench |
| Requests | API calls to Redmine |
| Python | Runtime environment |
| Chart.js (CDN) | Dashboard visualization |

# 

# **3. Configuration**

Setup Parameters

Configure via environment variables or hardcoded defaults

| **Parameter** | **Description** |
| --- | --- |
| REDMINE\_API\_URL | https://redmine.promantia.in |
| REDMINE\_API\_KEY | Get it from Redmine Website |
| USER\_EMAIL | Your Email |
| USER\_NAME | Your Name |

# 

# **4. Knows Issue**

1. **Hardcoded Credentials:**

API key is embedded in the tool (unsafe for public repos).

**Workaround:**

Migrate to environment variables.

1. **Pagination Limits:**

Fetches only the first 100 issues per request.

**Workaround:**

Implement loop with offset parameter.

# **5. Update** [**Hooks.py**](http://hooks.py)

# Register tools with Frappe Assistant Core

assistant\_tools = [

"redmine\_mcp\_tools.assistant\_tools.mcp\_custom\_tools.RedmineTool",

"redmine\_mcp\_tools.assistant\_tools.mcp\_custom\_tools.RedmineIssueReporterTool",

# Add more tools here as you create them

]

# Optional: Tool-specific configuration overrides

assistant\_tool\_configs = {

"redmine": {

"api\_url": "https://redmine.promantia.in",

"api\_key": "<your\_api\_key>",

"user\_email": "<your\_email>",

"user\_name": "<your\_name>",

"default\_include\_closed": False,

"default\_limit": 100,

"request\_timeout": 30,

"connect\_timeout": 10,

"debug": True

},

"redmine\_issue\_reporter": {

"api\_url": "https://redmine.promantia.in",

"api\_key": "<your\_api\_key>",

"user\_email": "<your\_email>",

"user\_name": "<your\_name>",

"default\_include\_closed": False,

"default\_limit": 100,

"request\_timeout": 30,

"connect\_timeout": 10,

"debug": True

}

}

# **6. Installation Steps**

1. Clone the Github Repo, the branch name is **main**.

<https://github.com/Sayeesh-Naik/redmine_mcp_tools.git>

1. Create a **.env** file and Add Redmine **Credentials**.

REDMINE\_API\_URL=https://redmine.promantia.in

REDMINE\_API\_KEY=<your\_api\_key>

USER\_EMAIL=<your\_email>

USER\_NAME=<your\_name>

1. Create a Python **Virtual Environment**.
2. Active Virtual Environment and Install all required Libraries

( *Registered in:* ***requirements.txt*** )

1. Apply Credentials in the file called: **mcp\_custom\_tools.py**
2. Install the **Redmine MCP Tool** App on your site.
3. Give **Migrate** once to refresh Custom Tools.
4. Run [**build.py**](http://build.py)file ( from Frappe Assistant Core ).
5. Set-Up **Claude Desktop** and Upload Latest .dxt file into MCP Extension.

And Register your frappe credentials in it.

1. **Done** - Use Specific Prompts to access the Redmine tool.