

MCP Servers Setup Complete

All MCP Servers Added

File: `.cursor/mcp.json`

Active Servers (No Setup Required)

1. *Filesystem MCP* ■

- File operations
- Read/write files
- List directories
- Search files

2. *SQLite MCP* ■

- `**Database:** `moode-source/var/local/www/db/moode-sqlite3.db``
- Read moOde configuration
- Query system settings
- Update configuration

3. *Puppeteer MCP* ■

- Browser automation
- Test moOde Web UI
- Take screenshots
- Automate browser tasks

4. *Fetch MCP* ■

- HTTP requests
- Test API endpoints
- Check Web UI status

5. *Memory MCP* ■

- Persistent memory
- Remember context
- Store information

Optional Servers (Need API Keys)

6. *GitHub MCP* ■

- Access GitHub repos

- Read source code
- Check for updates
- **Setup:** Add `GITHUB_PERSONAL_ACCESS_TOKEN` to environment

7. Brave Search MCP ■

- Web search
- Find solutions
- Research
- **Setup:** Add `BRAVE_API_KEY` to environment

8. PostgreSQL MCP ■

- PostgreSQL database access
- **Setup:** Add `POSTGRES_CONNECTION_STRING` to environment (if needed)

Most Useful for moOde Project

1. SQLite MCP

Use cases:

- Read moOde database configuration
- Query audio settings
- Check system state
- Update configuration programmatically

Example:

```
# I can now query moOde database directly
# Read configuration
# Check audio settings
# Update display configuration
```

2. Puppeteer MCP

Use cases:

- Test moOde Web UI automatically
- Navigate to <http://192.168.10.2>
- Take screenshots
- Test Room Correction Wizard
- Automate browser testing

Example:

```
# I can now:
1. Navigate to moOde Web UI
2. Take snapshot
3. Click buttons
4. Test functionality
5. Take screenshots
```

3. Fetch MCP

Use cases:

- Test moOde API endpoints
- Check Web UI status
- Make HTTP requests
- Test connectivity

Example:

```
# I can now:  
- Test http://192.168.10.2/api/endpoint  
- Check if Web UI is responding  
- Test API functionality
```

Benefits

1. **Automated Testing** - Puppeteer can test Web UI
2. **Database Access** - SQLite can read moOde config
3. **API Testing** - Fetch can test endpoints
4. **File Operations** - Filesystem MCP for faster operations
5. **Memory** - Remember context across sessions

Next Steps

1. **Use SQLite MCP** when Pi is ready to read moOde database
2. **Use Puppeteer MCP** to test Web UI automatically
3. **Use Fetch MCP** to test API endpoints
4. **Add API keys** if needed for GitHub/Brave Search

Configuration File

The configuration is in `.cursor/mcp.json`. All servers are configured and ready to use.

Note: Some servers (GitHub, Brave Search) require API keys. These can be added to environment variables if needed.