Shared Memory MCP Server - Setup Guide

What Was Wrong in Your Code

1. **Missing** `notifications/initialized` **-** After responding to `initialize`, you must send this notification
2. **Wrong method name -** It's `tools/call` not `tools/execute` in the current MCP spec
3. **Wrong parameter name -** It's `arguments` not `input` in `msg.params`
4. **No UPDATE operation -** Only had Add and Search
5. **No DELETE operation -** Essential for CRUD
6. **No LIST operation -** Needed to browse all memories
7. **Silent failures -** No handling for unknown tools

Two Versions Available

Option 1: Standalone (Recommended for Quick Start)

- No backend needed
- Stores data in `~/.mcp-shared-memory.json`
- Simpler to set up

Option 2: Backend Version

- Requires FastAPI backend
- More scalable for multiple clients
- Better for production use

Setup Instructions

Option 1: Standalone Setup

1. Create package.json

```
json
```

```
{
 "name": "mcp-shared-memory",
 "version": "1.0.0",
 "type": "module",
 "bin": {
  "mcp-memory": "./standalone.js"
 },
 "dependencies": {}
}
```

2. Save standalone code as `standalone.js`

- Make it executable: `chmod +x standalone.js`

3. Configure Claude Desktop

Edit `~/Library/Application Support/Claude/claude_desktop_config.json` (Mac) or `%APPDATA%\Claude\claude_desktop_config.json` (Windows):

```json
{
 "mcpServers": {
  "sharedMemory": {
   "command": "node",
   "args": ["/absolute/path/to/standalone.js"]
  }
 }
}
```

4. Restart Claude Desktop

Option 2: Backend Setup

1. Install Python dependencies

```bash
pip install fastapi uvicorn pydantic
```

2. Start the backend

```bash
```

```
python backend.py
```

3. Create package.json for client

```json
{
  "name": "mcp-shared-memory-client",
  "version": "1.0.0",
  "type": "module",
  "bin": {
    "mcp-memory": "./client.js"
  },
  "dependencies": {
    "axios": "^1.6.0"
  }
}
```

4. Install Node dependencies

```bash
npm install
```

5. Make client executable

```bash
chmod +x client.js
```

6. Configure Claude Desktop (same as above, but point to `client.js`)

Testing Your Setup

Once configured, restart Claude Desktop and try:

```
Can you add a memory for my "cursor-ide" project: "User prefers TypeScript with strict mode enabled"
```

```
Search for memories related to "TypeScript"
```

```
List all memories for the "cursor-ide" project
```

Usage Examples

Adding Memory

```javascript
// From any AI
"Remember for chatgpt project: User prefers concise responses"
```

Searching

```javascript
"Search memories about TypeScript preferences"
```

Listing by Project

```javascript
"Show me all memories for vscode"
```

Updating

```javascript
"Update memory mem_abc123 with new content: User now prefers tabs over spaces"
```

Deleting

```javascript
"Delete memory mem_abc123"
```

Storage Location

Standalone: `~/.mcp-shared-memory.json`
Backend: `~/.mcp-shared-memory.json`

Both use the same file format for easy switching between versions.

Troubleshooting

Claude doesn't see the tools

1. Check config file location is correct
2. Ensure absolute path to script
3. Restart Claude Desktop completely
4. Check Console logs (Help > Show Logs)

"Connection refused" (Backend version)

1. Make sure `python backend.py` is running
2. Check port 8000 is not in use
3. Verify `BASE_URL` in client.js matches backend

Permission denied

```bash
chmod +x standalone.js
# or
chmod +x client.js
```

Next Steps: Multi-AI Integration

To use this with ChatGPT, Gemini, VS Code, etc., you'll need to:

1. **ChatGPT**: Use Custom GPT with Actions (API endpoint needed)
2. **VS Code/Cursor**: Create an extension that calls your backend API
3. **Gemini**: Use Function Calling to connect to your API

Would you like me to create any of these integrations?