

Week 3: MCP Integration - Connecting to External Services

01. Session Goals

- Understand MCP (Model Context Protocol) architecture
 - Configure MCP servers for various services
 - Connect Claude to CRM, databases, and APIs
 - Build practical integrations for GTM workflows
-

02. Block 1: Theory - What is MCP? (30 min)

The Problem MCP Solves

Before MCP, connecting AI agents to external services required:

- Custom API integrations for each service
- OAuth flows and token management
- Tool definitions for every endpoint
- Maintenance as APIs change

MCP provides a universal protocol - one standard way to connect Claude to any external service.

The Story Behind MCP

MCP started as an internal project at Anthropic. Engineers were building the same integration patterns over and over, connecting Claude to different tools and data sources. Every team reinvented the wheel.

At an internal hackathon, every single entry was built on an early version of MCP. It went viral inside the company. That's when Anthropic knew they had something worth sharing.

November 2024: Anthropic open-sourced MCP with SDKs for Python and TypeScript.

March 2025: OpenAI adopted MCP across their Agents SDK, Responses API, and ChatGPT desktop. Sam Altman posted: "People love MCP and we are excited to add support across our products."

April 2025: Google DeepMind confirmed MCP support in upcoming Gemini models.

December 2025: Anthropic donated MCP to the Linux Foundation, establishing the Agentic AI Foundation.
Co-founders: Anthropic, Block, and OpenAI. Supporters: Google, Microsoft, AWS, Cloudflare, Bloomberg.

Why this matters:

MCP went from internal experiment to industry standard in one year. All major AI labs now back it. When you learn MCP, you're learning infrastructure that works across Claude, ChatGPT, Gemini, and Microsoft Copilot.

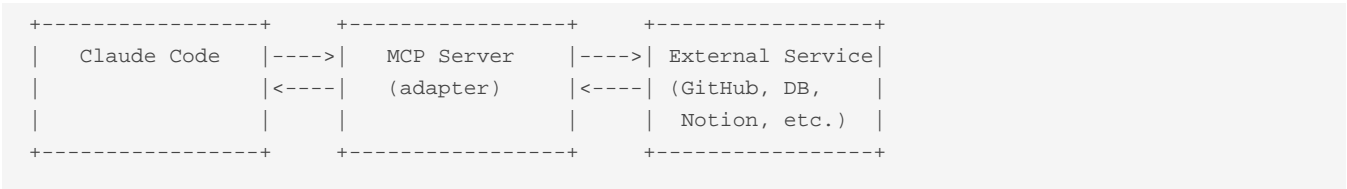
The protocol re-uses ideas from the Language Server Protocol (LSP) that powers code editors. If you've used VS Code's autocomplete for multiple languages, you've benefited from a similar standardization effort. MCP does the same for AI tool integrations.

Current scale: Over 10,000 public MCP servers. 97 million monthly SDK downloads. Enterprise deployment support from AWS, Google Cloud, and Azure.

References:

- Anthropic: Introducing the Model Context Protocol
(<https://www.anthropic.com/news/model-context-protocol>)
- Anthropic: Donating MCP to the Agentic AI Foundation
(<https://www.anthropic.com/news/donating-the-model-context-protocol-and-establishing-of-the-agentic-ai-foundation>)
- GitHub Blog: MCP Joins the Linux Foundation
(<https://github.blog/open-source/maintainers/mcp-joins-the-linux-foundation-what-this-means-for-developers-building-the>)

MCP Architecture



MCP Server: A lightweight adapter that:

- Exposes tools Claude can use
- Handles authentication
- Translates requests to service-specific APIs

Transport Types

Transport	Use Case	Command
HTTP	Cloud-hosted services	<code>`claude mcp add --transport http`</code>
SSE	Real-time services (deprecated)	<code>`claude mcp add --transport sse`</code>
stdio	Local tools, CLI wrappers	<code>`claude mcp add --transport stdio`</code>

Scopes

Scope	Location	Shared?	Use Case
local	`.claude.json` (project path)	No	Personal, sensitive credentials
project	`.mcp.json` in repo	Yes (git)	Team-shared services
user	`.claude.json` (global)	No	Personal tools across projects

Demo

Live demo: Add a GitHub MCP server and query issues.

```
claude mcp add --transport http github https://api.githubcopilot.com/mcp/
```

03. Block 2: Lab 1 - Your First MCP Integration (30 min)

Task: Connect to GitHub

Step 1: Add the GitHub MCP server

```
claude mcp add --transport http github https://api.githubcopilot.com/mcp/
```

Step 2: Authenticate

In Claude Code:

```
> /mcp
```

Select GitHub and follow the OAuth flow.

Step 3: Test the integration

```
> List my open pull requests

> Show me issues labeled "bug" in this repo

> What PRs have been merged this week?
```

Step 4: Verify with `/mcp`

Check that GitHub tools are available.

Success Criteria

- ☐ GitHub MCP server added
- ☐ Authentication complete
- ☐ Can query PRs and issues

- [] Understand available tools

04. BREAK (10 min)

05. Block 3: Theory - MCP for GTM Workflows (30 min)

Common MCP Servers for GTM

Server	Use Case	What It Enables
Notion	CRM, databases	Query contacts, deals, update records
PostgreSQL	Analytics DB	Run queries, explore schemas
Gmail	Email	Read threads, draft responses
Slack	Team comms	Post updates, read channels
HubSpot	CRM	Lead data, deal pipeline
Airtable	Structured data	Query bases, update records

Adding Database Connections

```
# PostgreSQL via dbhub
claude mcp add --transport stdio db -- npx -y @bytebase/dbhub \
  --dsn "postgresql://user:pass@host:5432/database"
```

Now Claude can:

```
> What tables are in the database?
> Show me the schema for the leads table
> Find all leads created this month
```

Adding Notion

```
claude mcp add --transport http notion https://mcp.notion.com/mcp
```

After OAuth:

```
> Search for opportunities in the CRM
> Update the status of deal "Acme Corp" to "Proposal Sent"
```

Environment Variables

For servers requiring API keys:

```
# Set via --env flag
claude mcp add --transport stdio airtable \
  --env AIRTABLE_API_KEY=your_key \
  -- npx -y airtable-mcp-server

# Or use shell expansion in .mcp.json
{
  "mcpServers": {
    "api": {
      "type": "http",
      "url": "${API_BASE_URL}/mcp",
      "headers": {
        "Authorization": "Bearer ${API_KEY}"
      }
    }
  }
}
```

MCP Resources and Prompts

Beyond tools, MCP servers can expose:

Resources - Data you can reference with `@`:

```
@github:issue://123
@notion:page://abc123
@postgres:schema://users
```

Prompts - Pre-built commands:

```
/mcp_github_pr_review 456
/mcp_notion_create_page "Meeting Notes"
```

06. Block 4: Lab 2 - Building a GTM Integration Stack (45 min)

Task: Connect Multiple Services for Lead Research

Build an integration that enables Claude to:

1. Query leads from a data source
2. Research companies on the web
3. Store findings in a structured format

Option A: Using Sample Data (No API Keys Required)

Create a local MCP server wrapper:

Step 1: Create `scripts/mock-crm-server.ts`:

```
// A simple stdio MCP server that reads from mock-crm.json
import { readFileSync } from 'fs';

const crm = JSON.parse(readFileSync('./data/mock-crm.json', 'utf-8'));

// Handle tool calls from stdin
process.stdin.on('data', (data) => {
  const request = JSON.parse(data.toString());

  if (request.tool === 'search_contacts') {
    const results = crm.contacts.filter(c =>
      c.name.toLowerCase().includes(request.query.toLowerCase())
    );
    process.stdout.write(JSON.stringify({ results }));
  }
  // ... more tool handlers
});
```

Step 2: Add as MCP server:

```
claude mcp add --transport stdio mock-crm -- npx ts-node scripts/mock-crm-server.ts
```

Option B: Using Real Services (API Keys Required)**Step 1: Add Notion for CRM:**

```
claude mcp add --transport http notion https://mcp.notion.com/mcp
```

Step 2: Connect and test:

```
> Search for contacts at technology companies
> Find all deals in "Negotiation" stage
> What's the total pipeline value?
```

GTM Workflow Test

Once connected, test a full workflow:

```
> I need to prepare for a call with Acme Corp.
> Find their contact info, recent activity, and any open deals.
> Then search the web for their latest news.
```

Deliverable

Screenshot showing:

1. `/mcp` output with connected servers
2. Successful query results
3. Multi-step workflow execution

07. Wrap-Up (15 min)

Key Takeaways

1. MCP = Universal Protocol - One way to connect to any service
2. Three transports - HTTP (cloud), stdio (local), SSE (deprecated)
3. Three scopes - local, project, user (precedence matters)
4. GTM power - Connect CRM, databases, email for complete workflows

Homework

Set up your integration stack:

1. Add at least 2 MCP servers relevant to your project:

Project Domain	Useful MCP Servers
GTM/Sales	CRM (HubSpot, Salesforce), Email (Gmail), Notion
Developer Tools	GitHub, PostgreSQL, Sentry, Linear
Content/Marketing	Notion, Google Docs, social APIs
Customer Support	Zendesk, Intercom, Slack
Operations	Google Sheets, Airtable, databases

2. Document:
 - Which servers you added
 - How you configured them
 - 3 useful queries you can now run
3. Create a `.mcp.json` for your project (if team-shareable)

Next Week Preview

Week 4: Agent Skills - Teaching Claude new capabilities with SKILL.md files

08. Facilitator Notes

Common Issues

1. OAuth failures: Ensure browser allows popups, try incognito
2. stdio server not starting: Check Node version, npm permissions
3. Connection timeouts: Increase with ``MCP_TIMEOUT=10000 claude``
4. "No such tool" errors: MCP server may have disconnected, re-add it

API Key Options for Participants

For live services, participants can:

- Use free tiers (Notion, Airtable have free plans)
- Use mock data servers (provided in repo)
- Pair up if some have accounts

Timing Adjustments

- If OAuth issues: Skip to Option A (mock data)
- If ahead: Explore additional MCP servers from registry
- Lab 2 can extend into homework if needed