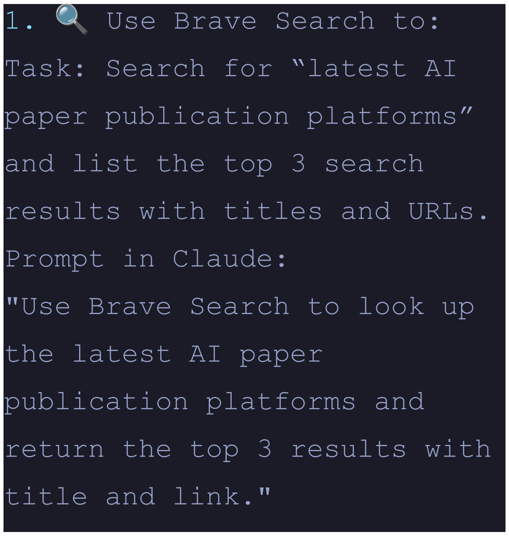
Module 1 Homework
Kevin Le - Fall Cohort 2025

Part 1 Using MCP to build Agent-like work-flow

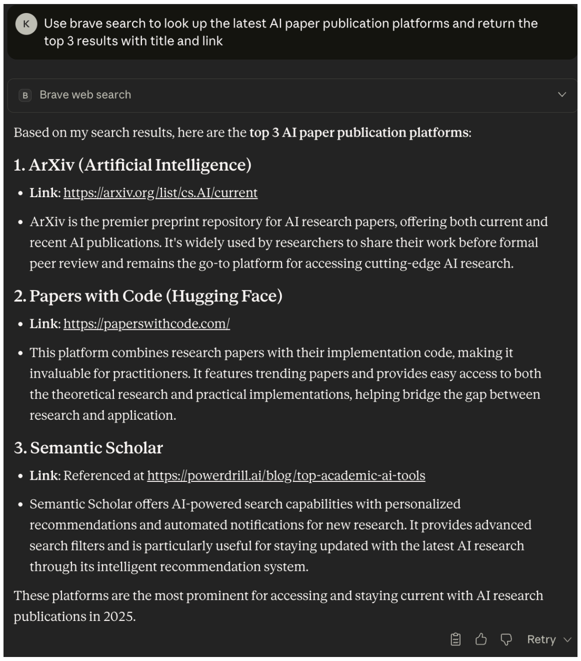


1. 🔍 Use Brave Search to:

Task: Search for “latest AI paper publication platforms” and list the top 3 search results with titles and URLs.

Prompt in Claude:

"Use Brave Search to look up the latest AI paper publication platforms and return the top 3 results with title and link."



K Use brave search to look up the latest AI paper publication platforms and return the top 3 results with title and link

Based on my search results, here are the **top 3 AI paper publication platforms**:

1. ArXiv (Artificial Intelligence)

- **Link**: https://arxiv.org/list/cs.AI/current
- ArXiv is the premier preprint repository for AI research papers, offering both current and recent AI publications. It's widely used by researchers to share their work before formal peer review and remains the go-to platform for accessing cutting-edge AI research.

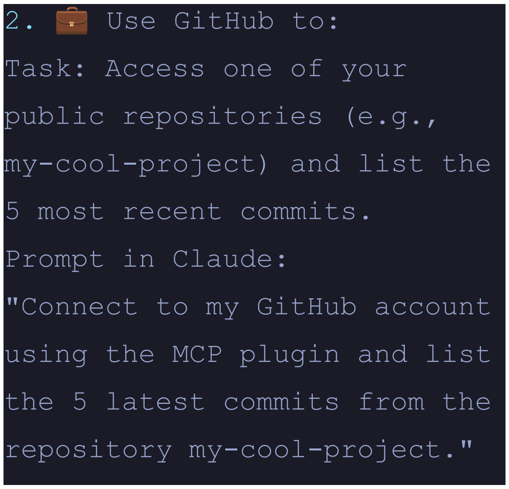
2. Papers with Code (Hugging Face)

- **Link**: https://paperswithcode.com/
- This platform combines research papers with their implementation code, making it invaluable for practitioners. It features trending papers and provides easy access to both the theoretical research and practical implementations, helping bridge the gap between research and application.

3. Semantic Scholar

- **Link**: Referenced at https://powerdrill.ai/blog/top-academic-ai-tools
- Semantic Scholar offers AI-powered search capabilities with personalized recommendations and automated notifications for new research. It provides advanced search filters and is particularly useful for staying updated with the latest AI research through its intelligent recommendation system.

These platforms are the most prominent for accessing and staying current with AI research publications in 2025.

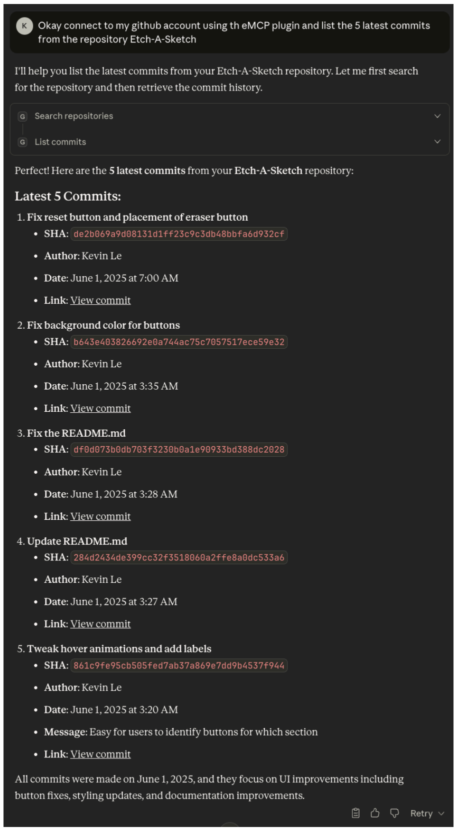


2. 💼 Use GitHub to:

Task: Access one of your public repositories (e.g., my-cool-project) and list the 5 most recent commits.

Prompt in Claude:

"Connect to my GitHub account using the MCP plugin and list the 5 latest commits from the repository my-cool-project."



K Okay connect to my github account using th eMCP plugin and list the 5 latest commits from the repository Etch-A-Sketch

I'll help you list the latest commits from your Etch-A-Sketch repository. Let me first search for the repository and then retrieve the commit history.

Perfect! Here are the **5 latest commits** from your **Etch-A-Sketch** repository:

Latest 5 Commits:

1. **Fix reset button and placement of eraser button**
    - **SHA**: `de2b069a9d08131d1ff23c9c3db48bbfa6d932cf`
    - **Author**: Kevin Le
    - **Date**: June 1, 2025 at 7:00 AM
    - **Link**: View commit
2. **Fix background color for buttons**
    - **SHA**: `b643e403826692e0a744ac75c7057517ece59e32`
    - **Author**: Kevin Le
    - **Date**: June 1, 2025 at 3:35 AM
    - **Link**: View commit
3. **Fix the README.md**
    - **SHA**: `df0d073b0db703f3230b0a1e90933bd388dc2028`
    - **Author**: Kevin Le
    - **Date**: June 1, 2025 at 3:28 AM
    - **Link**: View commit
4. **Update README.md**
    - **SHA**: `284d2434de399cc32f3518060a2ffe8a0dc533a6`
    - **Author**: Kevin Le
    - **Date**: June 1, 2025 at 3:27 AM
    - **Link**: View commit
5. **Tweak hover animations and add labels**
    - **SHA**: `861c9fe95cb505fed7ab37a869e7dd9b4537f944`
    - **Author**: Kevin Le
    - **Date**: June 1, 2025 at 3:20 AM
    - **Message**: Easy for users to identify buttons for which section
    - **Link**: View commit

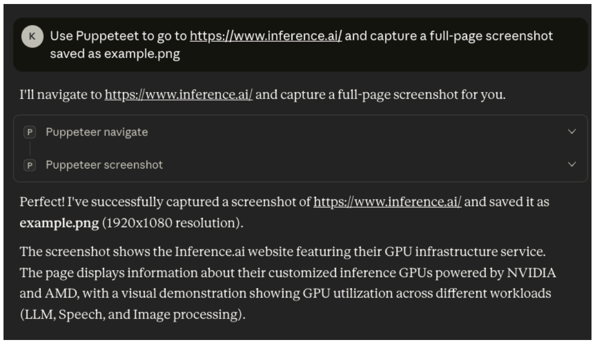
All commits were made on June 1, 2025, and they focus on UI improvements including button fixes, styling updates, and documentation improvements.

3. 🤖 Use Puppeteer to:

Task: Visit https://www.inference.ai/, take a full-page screenshot, and save it as example.png.

Prompt in Claude:

"Use Puppeteer to go to https://www.inference.ai/ and capture a full-page screenshot saved as example.png."

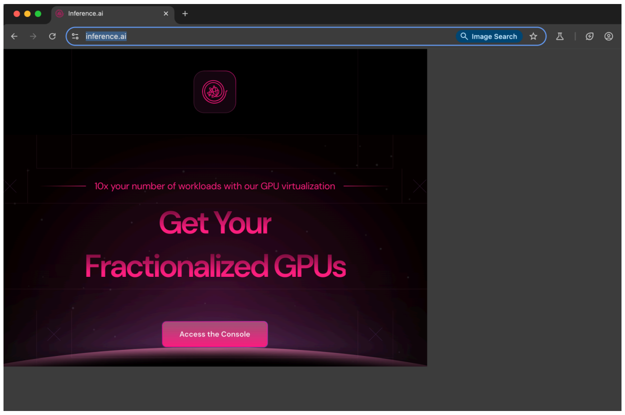


K Use Puppeteet to go to https://www.inference.ai/ and capture a full-page screenshot saved as example.png

I'll navigate to https://www.inference.ai/ and capture a full-page screenshot for you.

Perfect! I've successfully captured a screenshot of https://www.inference.ai/ and saved it as **example.png** (1920x1080 resolution).

The screenshot shows the Inference.ai website featuring their GPU infrastructure service. The page displays information about their customized inference GPUs powered by NVIDIA and AMD, with a visual demonstration showing GPU utilization across different workloads (LLM, Speech, and Image processing).

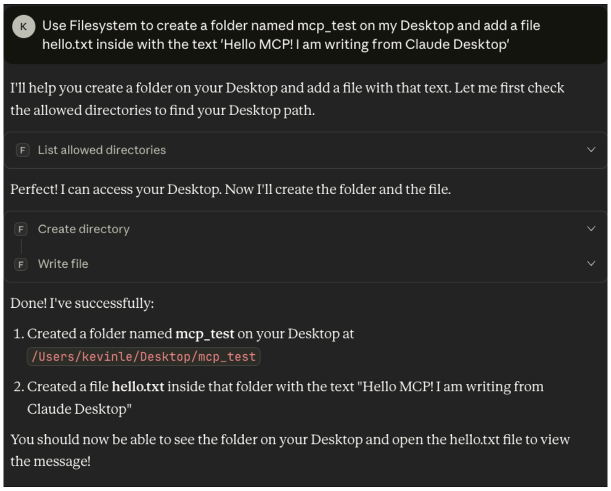

Inference.ai
inference.ai
Image Search
10x your number of workloads with our GPU virtualization
Get Your Fractionalized GPUs
Access the Console

4. 💾 Use Filesystem to:

Task: Create a new folder on your Desktop named mcp_test, and inside it, create a text file hello.txt containing “Hello MCP!”.

Prompt in Claude:

"Use Filesystem to create a folder named mcp_test on my Desktop and add a file hello.txt inside with the text 'Hello MCP!'."



K Use Filesystem to create a folder named mcp_test on my Desktop and add a file hello.txt inside with the text 'Hello MCP! I am writing from Claude Desktop'

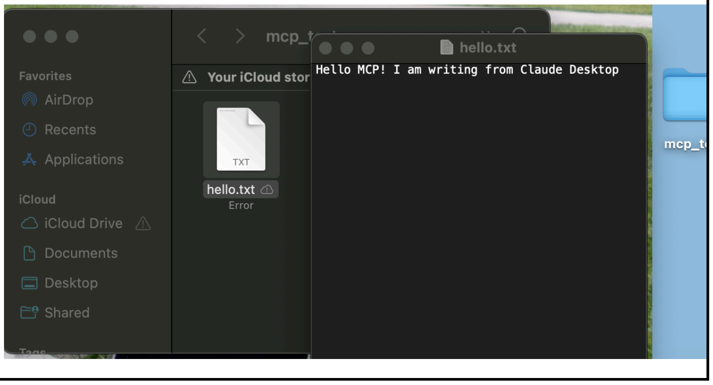
I'll help you create a folder on your Desktop and add a file with that text. Let me first check the allowed directories to find your Desktop path.

Perfect! I can access your Desktop. Now I'll create the folder and the file.

Done! I've successfully:

1. Created a folder named **mcp_test** on your Desktop at `/Users/kevinle/Desktop/mcp_test`
2. Created a file **hello.txt** inside that folder with the text "Hello MCP! I am writing from Claude Desktop"

You should now be able to see the folder on your Desktop and open the hello.txt file to view the message!



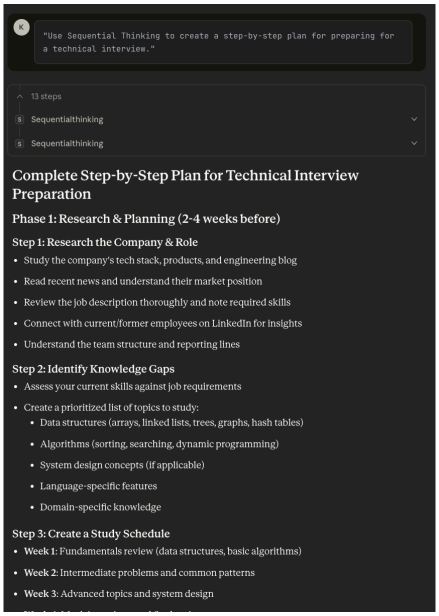
```
Hello MCP! I am writing from Claude Desktop
```

5. 🧠 Use Sequential Thinking to:

Task: Think step-by-step about how to prepare for a technical interview and generate a preparation plan.

Prompt in Claude:

"Use Sequential Thinking to create a step-by-step plan for preparing for a technical interview."



"Use Sequential Thinking to create a step-by-step plan for preparing for a technical interview."

Complete Step-by-Step Plan for Technical Interview Preparation

Phase 1: Research & Planning (2-4 weeks before)

Step 1: Research the Company & Role

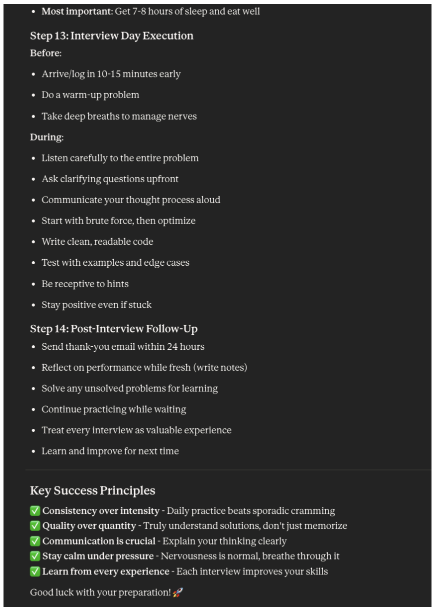
- Study the company's tech stack, products, and engineering blog
- Read recent news and understand their market position
- Review the job description thoroughly and note required skills
- Connect with current/former employees on LinkedIn for insights
- Understand the team structure and reporting lines

Step 2: Identify Knowledge Gaps

- Assess your current skills against job requirements
- Create a prioritized list of topics to study:
  - Data structures (arrays, linked lists, trees, graphs, hash tables)
  - Algorithms (sorting, searching, dynamic programming)
  - System design concepts (if applicable)
  - Language-specific features
  - Domain-specific knowledge

Step 3: Create a Study Schedule

- **Week 1:** Fundamentals review (data structures, basic algorithms)
- **Week 2:** Intermediate problems and common patterns
- **Week 3:** Advanced topics and system design



  - **Most important:** Get 7-8 hours of sleep and eat well

Step 13: Interview Day Execution

Before:

- Arrive/log in 10-15 minutes early
- Do a warm-up problem
- Take deep breaths to manage nerves

During:

- Listen carefully to the entire problem
- Ask clarifying questions upfront
- Communicate your thought process aloud
- Start with brute force, then optimize
- Write clean, readable code
- Test with examples and edge cases
- Be receptive to hints
- Stay positive even if stuck

Step 14: Post-Interview Follow-Up

- Send thank-you email within 24 hours
- Reflect on performance while fresh (write notes)
- Solve any unsolved problems for learning
- Continue practicing while waiting
- Treat every interview as valuable experience
- Learn and improve for next time

Key Success Principles

✅ **Consistency over intensity** - Daily practice beats sporadic cramming
✅ **Quality over quantity** - Truly understand solutions, don't just memorize
✅ **Communication is crucial** - Explain your thinking clearly
✅ **Stay calm under pressure** - Nervousness is normal, breathe through it
✅ **Learn from every experience** - Each interview improves your skills

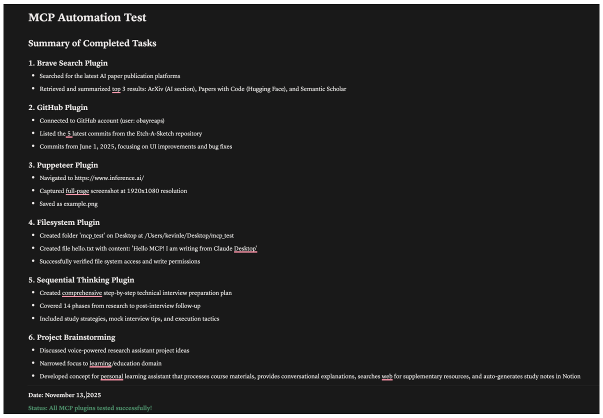
Good luck with your preparation! 🚀

6. 📝 Use Notion to:

Task: Create a new Notion page titled “MCP Automation Test” and log the results of all the tasks above.

Prompt in Claude:

"Use the Notion plugin to create a new page titled 'MCP Automation Test' and write a summary of the tasks I just completed using each plugin."



MCP Automation Test

Summary of Completed Tasks

1. Brave Search Plugin

- Searched for the latest AI paper publication platforms
- Retrieved and summarized top 3 results: ArXiv (AI section), Papers with Code (Hugging Face), and Semantic Scholar

2. GitHub Plugin

- Connected to GitHub account (user: obayreaps)
- Listed the 5 latest commits from the Etch-A-Sketch repository
- Commits from June 1, 2025, focusing on UI improvements and bug fixes

3. Puppeteer Plugin

- Navigated to https://www.inference.ai/
- Captured full-page screenshot at 1920x1080 resolution
- Saved as example.png

4. Filesystem Plugin

- Created folder 'mcp_test' on Desktop at /Users/kevinle/Desktop/mcp_test
- Created file hello.txt with content: 'Hello MCP! I am writing from Claude Desktop'
- Successfully verified file system access and write permissions

5. Sequential Thinking Plugin

- Created comprehensive step-by-step technical interview preparation plan
- Covered 14 phases from research to post-interview follow-up
- Included study strategies, mock interview tips, and execution tactics

6. Project Brainstorming

- Discussed voice-powered research assistant project ideas
- Narrowed focus to learning/education domain
- Developed concept for personal learning assistant that processes course materials, provides conversational explanations, searches web for supplementary resources, and auto-generates study notes in Notion

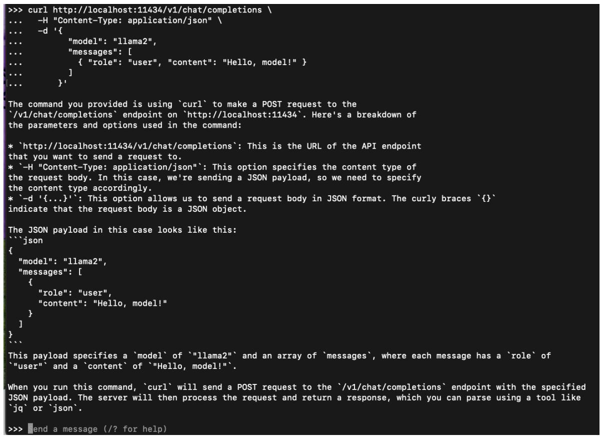
---

Date: November 13, 2025

Status: All MCP plugins tested successfully!

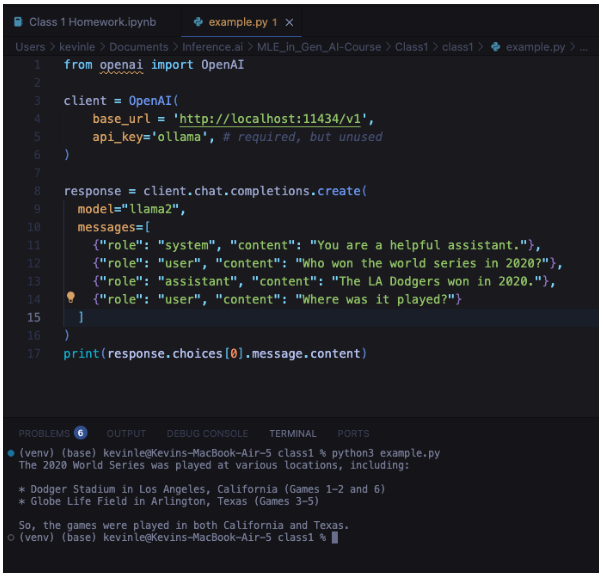
Part 2 Play with Ollama

Curl Method



~~~
>>> curl http://localhost:11434/v1/chat/completions \
...   -H "Content-Type: application/json" \
...   -d '{
...         "model": "llama2",
...         "messages": [
...           { "role": "user", "content": "Hello, model!" }
...         ]
...       }'

The command you provided is using `curl` to make a POST request to the
`/v1/chat/completions` endpoint on `http://localhost:11434`. Here's a breakdown of
the parameters and options used in the command:

* `http://localhost:11434/v1/chat/completions`: This is the URL of the API endpoint
that you want to send a request to.
* `-H "Content-Type: application/json"`: This option specifies the content type of
the request body. In this case, we're sending a JSON payload, so we need to specify
the content type accordingly.
* `-d '{...}'`: This option allows us to send a request body in JSON format. The curly braces `{}`
indicate that the request body is a JSON object.

The JSON payload in this case looks like this:
```json
{
  "model": "llama2",
  "messages": [
    {
      "role": "user",
      "content": "Hello, model!"
    }
  ]
}
```
This payload specifies a `model` of `"llama2"` and an array of `messages`, where each message has a `role` of
`"user"` and a `content` of `"Hello, model!"`.

When you run this command, `curl` will send a POST request to the `/v1/chat/completions` endpoint with the specified
JSON payload. The server will then process the request and return a response, which you can parse using a tool like
`jq` or `json`.

>>> Send a message (/? for help)
~~~

OpenAI Python Library


Class 1 Homework.ipynb
example.py 1
Users > kevinle > Documents > Inference.ai > MLE_in_Gen_AI-Course > Class1 > class1 > example.py > ...
PROBLEMS 6
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS

```python
from openai import OpenAI

client = OpenAI(
    base_url = 'http://localhost:11434/v1',
    api_key='ollama', # required, but unused
)

response = client.chat.completions.create(
  model="llama2",
  messages=[
    {"role": "system", "content": "You are a helpful assistant."},
    {"role": "user", "content": "Who won the world series in 2020?"},
    {"role": "assistant", "content": "The LA Dodgers won in 2020."},
    {"role": "user", "content": "Where was it played?"}
  ]
)
print(response.choices[0].message.content)
```

```
● (venv) (base) kevinle@Kevins-MacBook-Air-5 class1 % python3 example.py
  The 2020 World Series was played at various locations, including:

  * Dodger Stadium in Los Angeles, California (Games 1-2 and 6)
  * Globe Life Field in Arlington, Texas (Games 3-5)

  So, the games were played in both California and Texas.
○ (venv) (base) kevinle@Kevins-MacBook-Air-5 class1 %
```

Part 3

What is LCEL?	LCEL is the easiest way to build agents and applications with LLMs, using an architecture of fewer than 10 lines.
Why use LCEL?	LCEL is utilized if you want to build agents and autonomous applications quickly. Plus, the user doesn't need to know LangGraph to use LangChain.