Gmail

Rohit Aggarwal <rohit313agg@gmail.com>

OpenAI Releases New Deep Research API

Unwind AI <unwindai@mail.beehiiv.com>
Reply-To: Unwind AI <unwindainewsletter@gmail.com>
To: "rohit313agg@gmail.com" <rohit313agg@gmail.com>

Mon, Jul 7, 2025 at 8:31 AM

July 07, 2025 | Read Online

OpenAI Releases New Deep Research API

PLUS: China's ByteDance opensources AI SWE Agent, Scrape web content behind auth walls


Shubham Saboo & Gargi Gupta

July 07, 2025

Today’s top AI Highlights:

1. **ByteDance just opensourced their AI SWE agent**
2. **Build Deep Research agents with OpenAI API in ~20 lines of code**
3. **Your MoE models are wasting compute - Huawei's model fixes that**
4. **Customize Claude Code workflows with shell commands**
5. **Easily scrape web content behind auth walls**

& so much more!

Read time: 3 mins

AI Tutorial

Building tools that truly understand your documents is hard. Most RAG implementations just retrieve similar text chunks without actually reasoning about them, leading to shallow responses. The real solution lies in creating a system that can process documents, search the web when needed, and deliver thoughtful analysis. Moreover, running the pipeline locally would reduce latency and ensure privacy and control over sensitive data.

In this tutorial, we'll build a powerful **Local RAG Reasoning Agent** that runs entirely on your own machine. You'll be able to choose between multiple state-of-the-art opensource models like **Qwen 3**, **Gemma 3**, and **DeepSeek R1** to power your system.

This hybrid setup combines document processing, vector search, and web search capabilities to deliver thoughtful, context-aware responses without cloud dependencies.

We share hands-on tutorials like this every week, designed to help you stay ahead in the world of AI. **If you're serious about leveling up your AI skills and staying ahead of the curve, subscribe now and be the first to access our latest tutorials.**

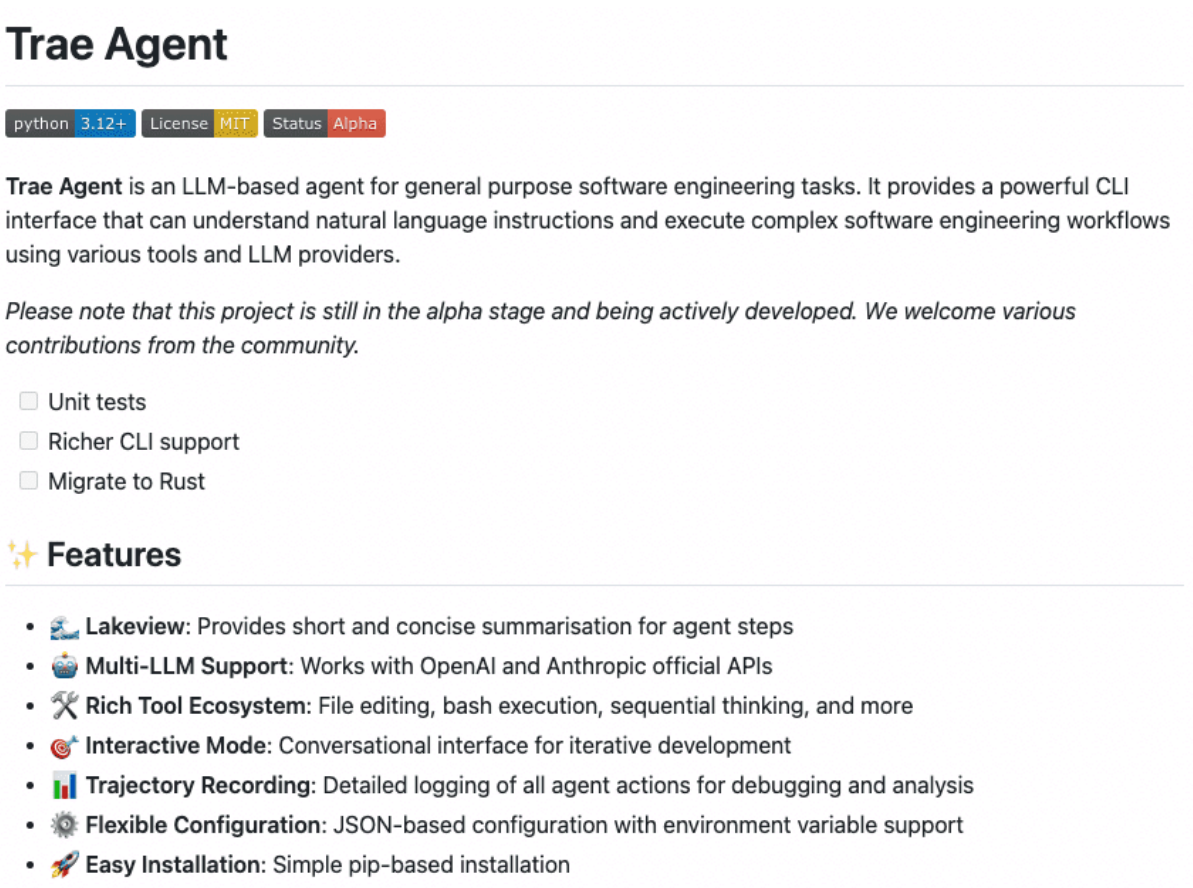
Build a Qwen 3 Local RAG Reasoning Agent

Fully functional local agentic RAG app with step-by-step instructions (100% opensource)

Don’t forget to share this newsletter on your social channels and tag **Unwind AI** (**X**, **LinkedIn**, **Threads**) to support us!

Latest Developments

ByteDance’s Opensource AI Software Engineering Agent 👨‍🔧🧰🔓



Trae Agent

python 3.12+ License MIT Status Alpha

Trae Agent is an LLM-based agent for general purpose software engineering tasks. It provides a powerful CLI interface that can understand natural language instructions and execute complex software engineering workflows using various tools and LLM providers.

Please note that this project is still in the alpha stage and being actively developed. We welcome various contributions from the community.

- [ ] Unit tests
- [ ] Richer CLI support
- [ ] Migrate to Rust

✨ Features

- 🌊 **Lakeview**: Provides short and concise summarisation for agent steps
- 🤖 **Multi-LLM Support**: Works with OpenAI and Anthropic official APIs
- 🛠️ **Rich Tool Ecosystem**: File editing, bash execution, sequential thinking, and more
- 🎯 **Interactive Mode**: Conversational interface for iterative development
- 📊 **Trajectory Recording**: Detailed logging of all agent actions for debugging and analysis
- ⚙️ **Flexible Configuration**: JSON-based configuration with environment variable support
- 🚀 **Easy Installation**: Simple pip-based installation

ByteDance just decided to give away the secret sauce.

They have opensourced **Trae Agent**, the AI coding agent in their agentic terminal Trae.

While everyone's debating subscription prices for Cursor and Claude, ByteDance dropped Trae Agent as MIT-licensed code that any developer can clone and modify.

It’s not just a chat assistant; it's an autonomous agent that powers their IDE's Builder mode, which helps you develop projects from scratch, utilize tools and MCP servers, analyze and edit code files, run

commands, and more. It gives developers direct access to enterprise-grade agentic architecture without the monthly fees.

Key Highlights:

1. **Autonomous task execution** - Converts plain English instructions into multi-step software engineering workflows with automatic task breakdown.
2. **Execution logging** - Records every agent decision, tool call, and workflow step in detailed JSON files for debugging and analysis.
3. **Rich built-in tool ecosystem** - Includes file editing, bash execution, sequential thinking, and task completion tools designed for real development workflows.
4. **Interaction modes** - Offers both interactive chat sessions and direct CLI task execution with JSON-based configuration.

Build Deep Research Agents With OpenAI API 🧐 🌐 🔎

```python
# Define the research agent
research_agent = Agent(
    name="Research Agent",
    model="o4-mini-deep-research-2025-06-26",
    tools=[WebSearchTool()],
    instructions="You perform deep empirical research based on the user's question."
)

# Async function to run the research and print streaming progress
async def basic_research(query):
    print(f"Researching: {query}")
    result_stream = Runner.run_streamed(
        research_agent,
        query
    )

    async for ev in result_stream.stream_events():
        if ev.type == "agent_updated_stream_event":
            print(f"\n--- switched to agent: {ev.new_agent.name} ---")
            print(f"\n--- RESEARCHING ---")
        elif (
            ev.type == "raw_response_event"
            and hasattr(ev.data, "item")
            and hasattr(ev.data.item, "action")
        ):
            action = ev.data.item.action or {}
            if action.get("type") == "search":
                print(f"[Web search] query={action.get('query')!r}")

    # streaming is complete → final_output is now populated
    return result_stream.final_output

# Run the research and print the result
result = await basic_research("Research the economic impact of semaglutide on global healthcare sy
print(result)
```

OpenAI just dropped their Deep Research into the API, and they're packing some serious firepower.

These are the same post-trained o3 and o4-mini models that power Deep Research in ChatGPT, now available with full programmatic access and native support for web search, code execution, and MCP

servers that let you plug into your own data sources.

The Deep Research API handles complex, multi-step research workflows autonomously - you throw it a high-level query and it plans sub-questions, searches the web, executes code, and synthesizes everything into structured, citation-rich reports. OpenAI has released some amazing examples on building your own deep research agents using their API and the Agents SDK with MCP in about 20 lines of code.

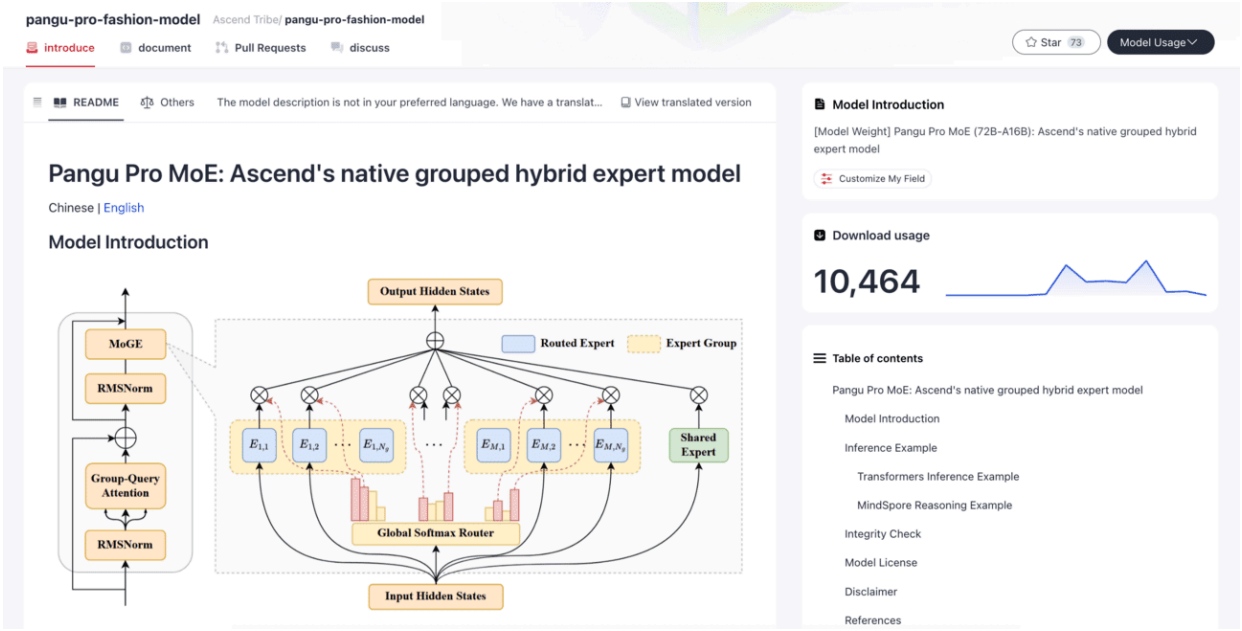
Key Highlights:

1. **Autonomous Research Workflow** - The models decompose complex queries into sub-questions, perform web searches, and synthesize results into structured reports with inline citations and source metadata, all without manual intervention.
2. **MCP Server support** - Native MCP support lets you connect the research models to your internal databases, documents, and APIs, enabling research that combines public web data with your proprietary knowledge.
3. **Multi-Agent Architecture** - The API works seamlessly with OpenAI's Agents SDK, allowing you to build sophisticated multi-agent pipelines with triage, clarification, instruction, and research agents working together.
4. **Implementation** - Don't forget to check out these cookbooks by OpenAI:

- **Introduction to Deep Research API** to build your own Deep Research in around 30 lines of code.
- **Deep Research API with the Agents SDK** and a multi-agent system using four agents.

Artificial Analysis found these endpoints come with a hefty price tag - they spent $100 on o3-deep-research and $9.18 on o4-mini-deep-research across just 10 test queries. The deep research versions are substantially more expensive than standard o3 and o4-mini endpoints due to their specialized training and high token usage.

Huawei’s Opensource Model Fixes MoE Problem ⚡ 🚀



Mixture of Experts architecture powers some of the frontier models today. But they might be wasting your compute resources because some experts work overtime while others sit idle.

MoE models can't control which experts get activated for each token. This seemingly small issue creates a massive bottleneck: some experts become overworked while others sit idle. Since experts are distributed across different devices, your busiest device becomes the slowest link that throttles your entire inference pipeline.

China’s **Huawei released Pangu Pro MoE 72B** model that tackles this chaos with a Mixture of Grouped Experts (MoGE) that enforces balanced expert activation across predefined groups.

Unlike traditional MoE, MoGE guarantees equal workload distribution, making the 72B model (with 16B active parameters) run significantly smoother across distributed setups.

Key Highlights:

1. **Expert Grouping** - MoGE architecture groups 64 routing experts into 8 equal groups, forcing tokens to activate exactly one expert per group, eliminating the load imbalance problem.
2. **High Throughput** - Achieves 1,148 tokens/second per card, with an even better performance with speculative acceleration on Ascend NPU hardware. The model is specifically optimized for runtime performance on these NPUs.
3. **Model Comparisons** - Outperforms Qwen3-32B and Gemma3-27B across benchmarks while using only 16B active parameters.
4. **Open Source on Ascend NPUs** - Full model weights and inference code available now for Huawei's Ascend 300I Duo and 800I A2 NPU chips, with support for Transformers and MindSpore frameworks.

Quick Bites

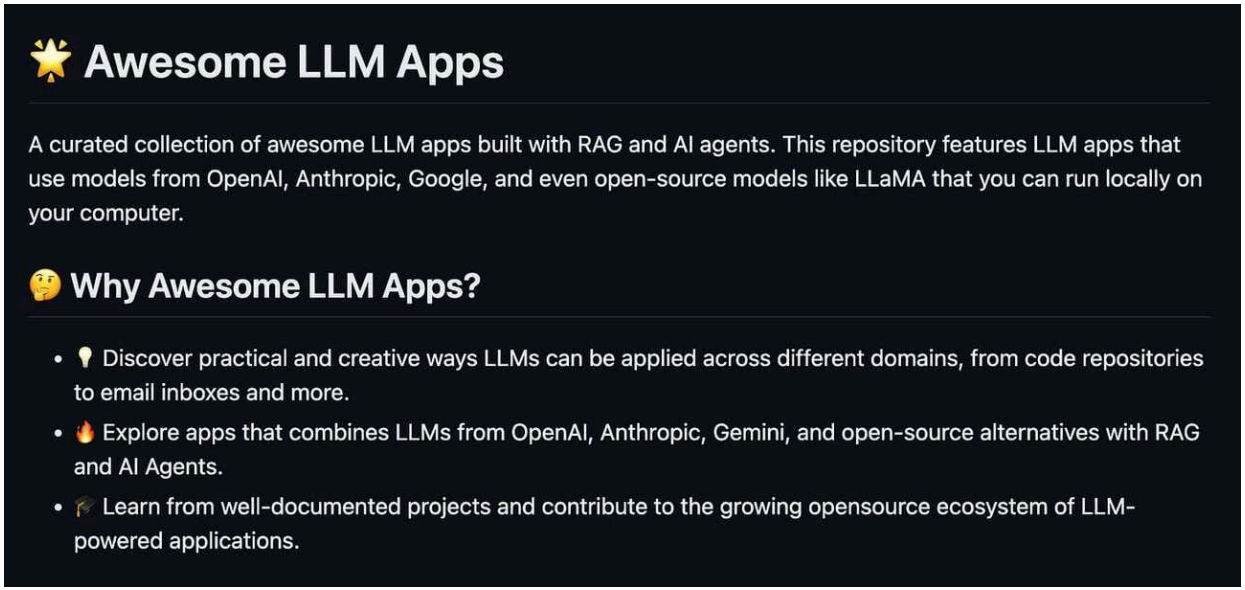
You can now **intercept and control every step of Claude Code's agent loop using shell commands with this new feature called Hooks**. These user-defined commands execute at specific points in the workflow, giving you deterministic control over when certain actions happen—whether that's sending Slack notifications after task completion, running linters after every file write, or implementing custom permission gates. Configure hooks through the `/hooks` command and choose from events like PreToolUse, PostToolUse, or Stop to trigger your custom workflows.

This Chrome extension just claimed the top spot on the Halluminate Web Bench with an **81.39% success rate, beating OpenAI's Operator and every other major web agent. rtrvr.ai** achieved this by taking a fundamentally different approach - running locally and using a DOM-based approach. These allow the AI agents to bypass bot detection and access your logged-in accounts seamlessly, as well as understand the

web pages more robustly than visual parsing. At 0.9 minutes average task time, it's also 7x faster than the next best competitor.

Tools of the Trade

1. **Firecrawl's Authenticated Scraping**: New /scrape endpoint that lets you extract content from pages behind login walls by maintaining session state across requests. You authenticate once through their browser interface or API, get a token, then use it for subsequent scrapes of protected content.
2. **MCP CLI**: Command-line interface for using MCP servers with LLMs, with features like streaming responses, concurrent tool execution, performance monitoring, etc. Offers three modes- chat, interactive, and command mode - with support for multiple providers (OpenAI, Anthropic, Ollama).
3. **BrowserOS**: Opensource agentic browser that integrates AI agents directly into the browsing experience for automating tasks like research, exploration, form filling, etc., completely locally. Supports multiple AI providers, including OpenAI, Anthropic, and local models via Ollama.
4. **Prompt Coder**: Turn your ideas into perfect prompts. It converts UI screenshots into prompts optimized for AI coding agents like Cursor, Bolt, Windsurf, and Trae. It analyzes design elements and generates framework-specific code prompts for Next.js, React, Vue, Flutter, and vanilla JavaScript.
5. **Awesome LLM Apps**: Build awesome LLM apps with RAG, AI agents, MCP, and more to interact with data sources like GitHub, Gmail, PDFs, and YouTube videos, and automate complex work.


🌟 Awesome LLM Apps
A curated collection of awesome LLM apps built with RAG and AI agents. This repository features LLM apps that use models from OpenAI, Anthropic, Google, and even open-source models like LLaMA that you can run locally on your computer.
🤔 Why Awesome LLM Apps?
• 💡 Discover practical and creative ways LLMs can be applied across different domains, from code repositories to email inboxes and more.
• 🔥 Explore apps that combines LLMs from OpenAI, Anthropic, Gemini, and open-source alternatives with RAG and AI Agents.
• 🎓 Learn from well-documented projects and contribute to the growing opensource ecosystem of LLM-powered applications.

Hot Takes

1. It's crazy that it's 2025 and Siri is just as bad as it was in 2015 ~
   greg

2. gemini 2.5 pro is a weird model extremely good but also extremely weird ~
   adi

3. Sometimes I step into a meeting with AI researchers and they'll be casually demoing technology that will put us in the literal Matrix when it launches.

   And then they'll be like "what's for lunch?" ~
   Nikita Bier

That's all for today! See you tomorrow with more such AI-filled content.

Don't forget to share this newsletter on your social channels and tag **Unwind AI** to support us!

Unwind AI - **X** | **LinkedIn** | **Threads**

Awesome LLM Apps | **Sponsor Us**

PS: We curate this AI newsletter every day for FREE, your support is what keeps us going. If you find value in what you read, share it with at least one, two (or 20) of your friends 😊

Subscribe now for FREE!

If this email was forwarded to you, **subscribe here** to get email from Unwind AI daily for free!

Update your email preferences or unsubscribe here

228 Park Ave S, #29976, New York, New York 10003, United States