

# GenAI Workshop Series: Analysis & Interpretation

Yassine Baghoussi, Inês Sousa, Lia Patricio and Gabriel David

Auditorium A - INESC TEC

CREATING A FULFILLING  
AND SUSTAINABLE FUTURE  
THROUGH IMPACTFUL  
**SCIENCE, TECHNOLOGY  
AND INNOVATION.**

## GenAI Workshop Series - Session 3

### Turning Data into Insight

From organized research to insightful results

# Using AI as Your Research Assistant

## Session 1 – Ideation & Planning: From Ideas to Ethics

- November 6, 2025 | 9:30 AM – 11 AM | Auditorium A

## Session 2 – Research Design & Execution: Designing with Intelligence

- January 16, 2026 | 9:30 AM – 11 AM | Auditorium A

## **Session 3 – Analysis & Interpretation: Turning Data into Insight**

- **February 26, 2026 | 11:00 AM – 12:30 AM | Auditorium A**

## Session 4 – Reviewing & Dissemination: Sharing with Impact

- March 2026

# Today's Journey

## 90-Minute Workshop Structure:

Time	Section	Focus
10 min	<b>Part 1:</b> Foundation	Understanding Agentic AI
25 min	<b>Part 2:</b> AI Tools for Data Analysis	Claude, Claude Code, Jupyter AI, Cursor
15 min	<b>Part 3:</b> AI Tools for Interpretation	Research synthesis, visualization, reporting
25 min	<b>Part 4:</b> Hands-On Exercise	End-to-end analysis workflow
15 min	<b>Part 5:</b> Discussion & Wrap-up	Key takeaways, next steps

# Workshop Series Overview

## Session 1

### Ideation and Planning

From Ideas to Ethics

## Session 2

### Research Design and Execution

Designing with Intelligence

## Session 3

### Analysis & Interpretation

Turning Data into Insight

## Session 4

### Reviewing & Dissemination

Sharing with Impact

# Previous Sessions Recap

## Session 1 – Ideation & Planning: From Ideas to Ethics

November 6, 2025 | 9:30 AM – 11 AM

### What We Covered:

- **The AI Revolution in Research Discovery**
  - From keyword search to semantic understanding
  - Tools that find papers by meaning, not just matching terms
- **Key Principles Established:**
  - AI as brainstorming partner, not oracle
  - Always verify AI-suggested citations
  - Use specialized tools for specialized tasks (Elicit, Scite.ai, Connected Papers)
  - Combine tools for comprehensive literature coverage
- **The Gap We Left:** *"You have ideas. You have literature. Now what?"*



Foundation model

Scopus® AI

Retrieval  
Augmented  
Generation

# Previous Sessions Recap

## Session 2 – Research Design & Execution: Designing with Intelligence

January 16, 2026 | 9:30 AM – 11 AM

### What We Covered:

#### In-Context Learning & Custom GPT

- Persistent system instructions and knowledge bases
- Custom GPT vs Projects (behavior vs context)
- Building domain-specific AI assistants

#### AI Scientist – Start Ready from Scratch

- Seed ideas to complete papers

#### AI Task Management

- Claude Code for local file operations
- Notion AI for research databases
- Model Context Protocol (MCP) integrations

**The Gap We Left:** *You can design and plan research with AI. But how do you automate data analysis?*

### Projects vs Custom GPTs

When to use which?



#### Projects

Organize chats and files.  
Per-project memory.



#### Custom GPTs

Fixed instructions, reusable.  
Actions and knowledge.

# Overview

PART 1: clear separation between foundation context and other parts

# Agentic AI vs. Generative AI

Agentic AI acts autonomously to complete goals through a perceive-reason-act-learn-coordinate cycle. Generative AI creates outputs from prompts. The shift: from generating content to achieving objectives

AI Agents

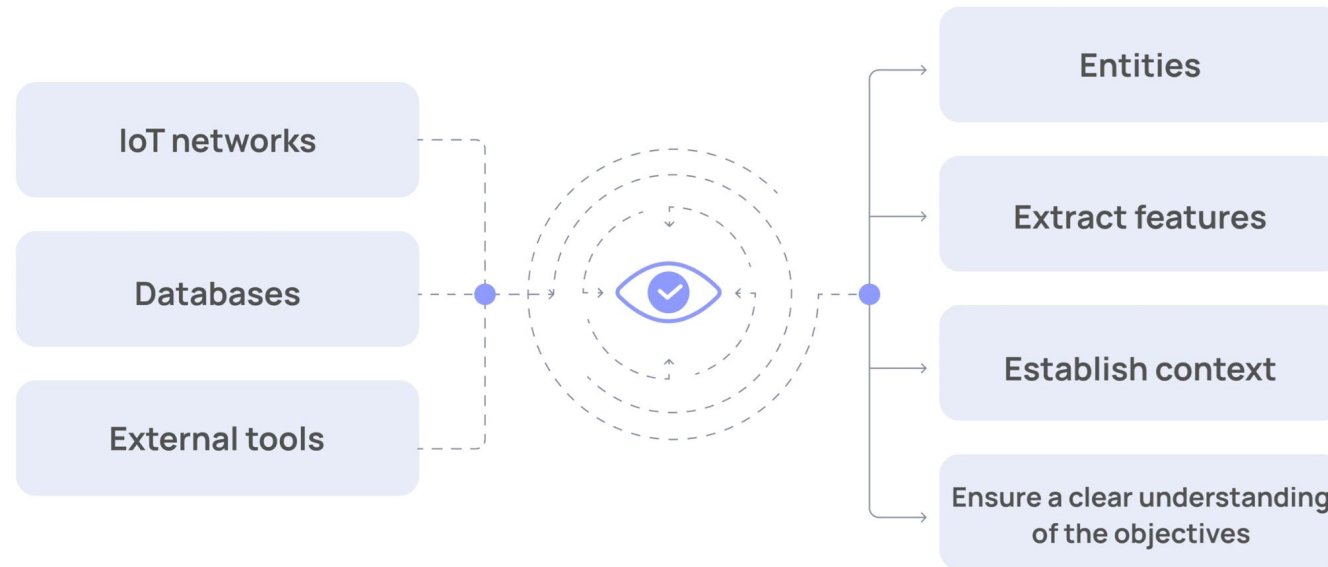


Agentic AI



# Perceive Phase of Agentic AI

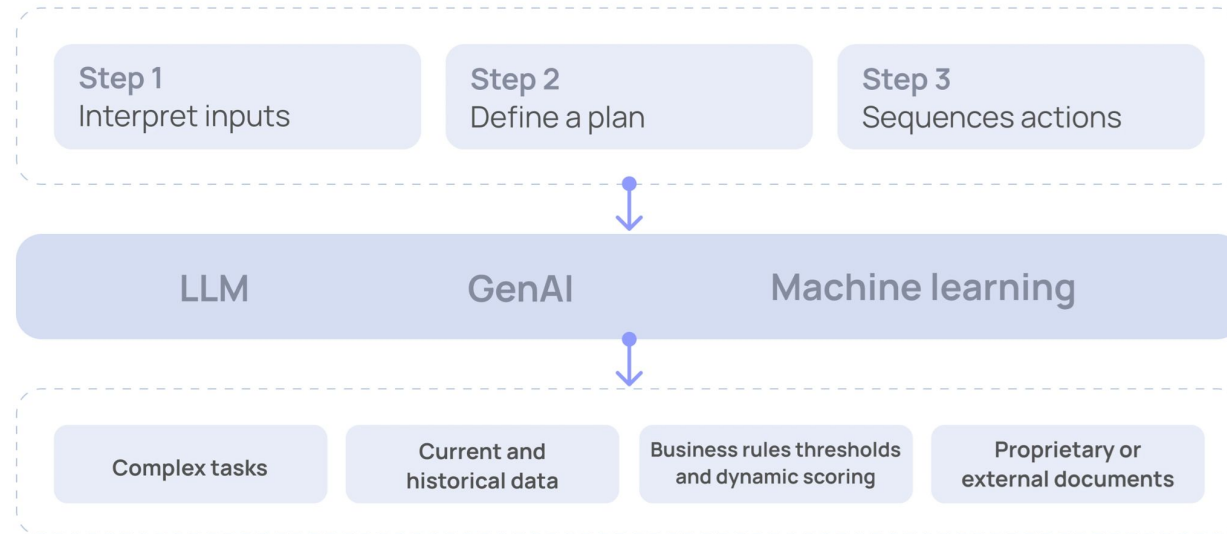
Agentic AI gathers information from multiple sources simultaneously your data files, databases, external APIs, and real-time tools, to build a complete picture of your research context



**For researchers:** Claude Code can perceive your project structure, dataset characteristics, code dependencies, and coordinate with tools like Perplexity or GitHub, all before taking action

# Reason Phase of Agentic AI

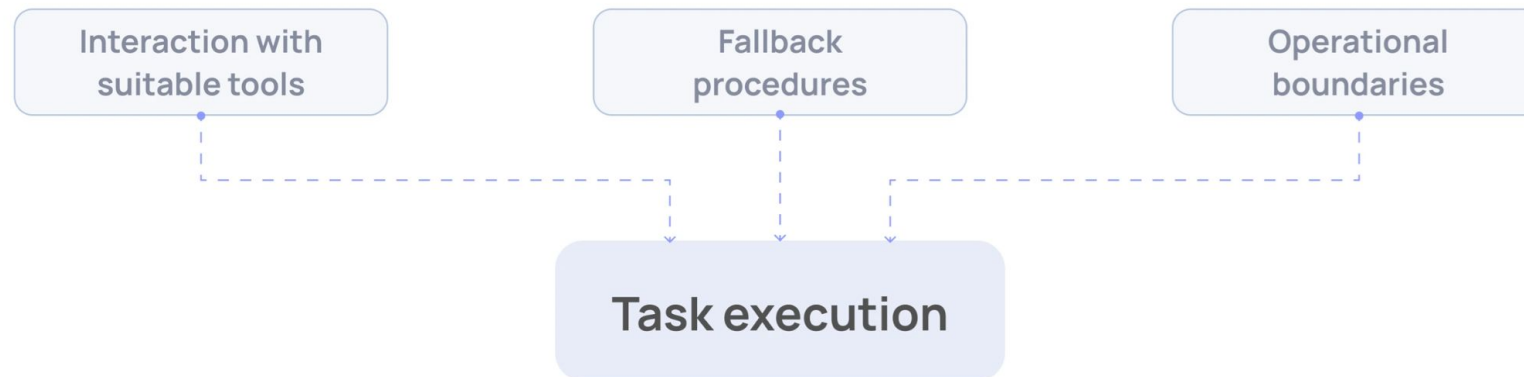
After perceiving the context, Agentic AI reasons about the best approach, breaking complex tasks into steps, considering constraints, and planning optimal strategies using multiple AI techniques.



This reasoning ability, handling ambiguity, adapting to constraints, and planning multi-step solutions, is what separates Agentic AI from traditional automation

# Act Phase of Agentic AI

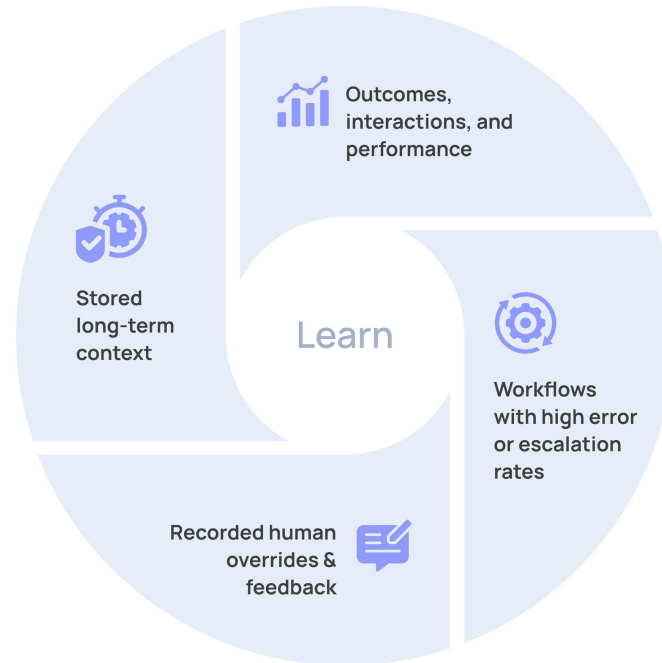
Agentic AI executes planned tasks by selecting and using appropriate tools, APIs, code execution, file operations, while respecting operational boundaries and handling errors autonomously.



Predefined constraints and fallback procedures ensure reliable, safe execution while maintaining autonomy.

# Learn Phase of Agentic AI

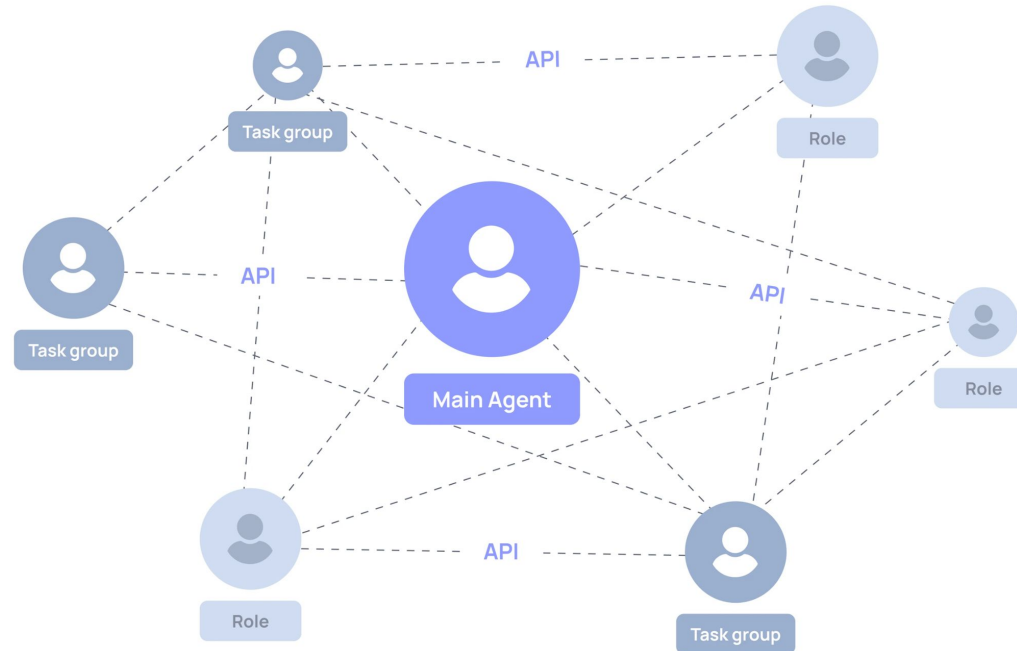
Agentic AI doesn't just execute tasks once. It learns from outcomes, captures performance metrics, incorporates your feedback, and stores context to improve future interactions.



This learning loop enables the system to adapt, reduce errors, and handle increasingly complex research tasks over time.

# Coordinate Phase of Agentic AI

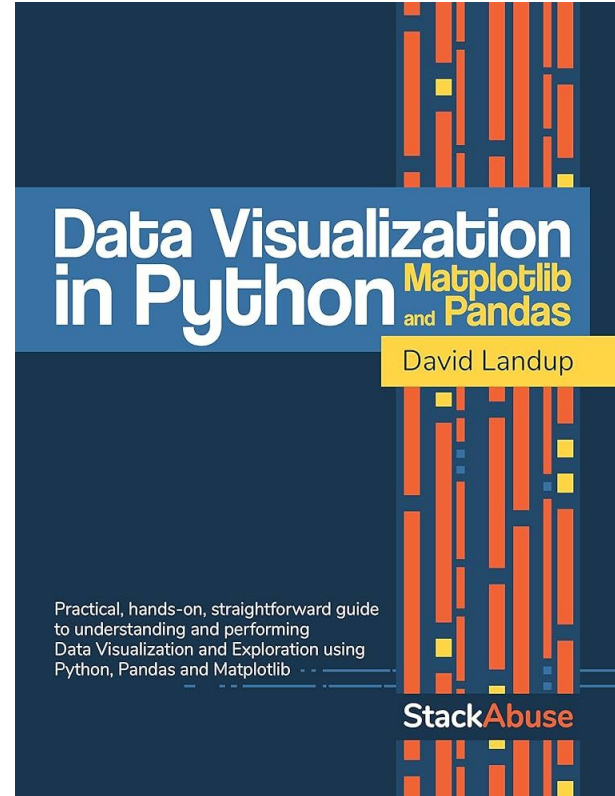
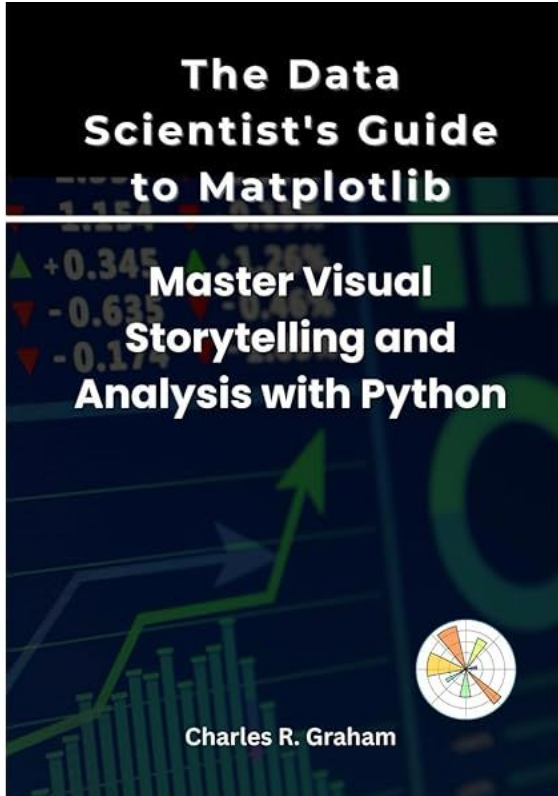
Agentic AI coordinates multiple specialized agents through APIs and protocols, with a main agent orchestrating tasks, managing priorities, and resolving conflicts to achieve complex research goals.



This coordination through Model Context Protocol (MCP) enables researchers to leverage multiple tools and data sources seamlessly.

## PART 2: AI Tools for Data Analysis

# The Research Analysis Challenge



Northwestern  
Department of Statistics  
and Data Science

Introduction to Data  
Science with Python

Preface

Quick overview: Python  
programming

- 1 Introduction to Python and  
Jupyter Notebooks
- 2 Data structures

Exploratory data analysis

- 3 Reading data
- 4 NumPy
- 5 Pandas
- 6 Data visualization
- 7 Data cleaning and preparation
- 8 Data wrangling
- 9 Data aggregation
- 10 Alternative solutions by  
students

Appendices  
References

Exploratory data analysis > 3 Reading data

## 3 Reading data

### 3.1 Types of data - structured and unstructured

Reading data is the first step to extract information from it. Data can exist broadly in two formats:

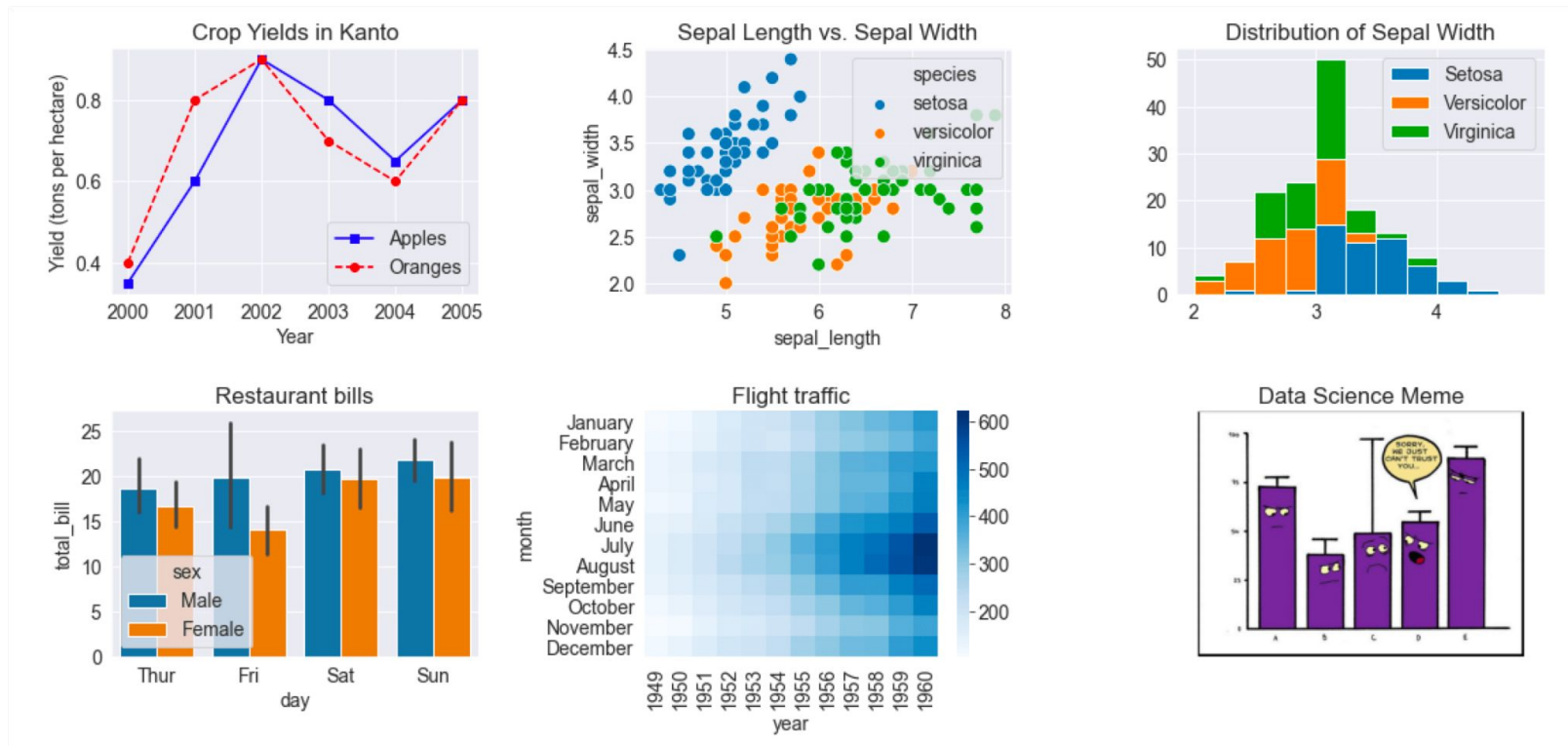
1. Structured data, and
2. Unstructured data.

Structured data is typically stored in a tabular form, where rows in the data correspond to “observations” and columns correspond to “variables”. For example, the following dataset contains 5 observations, where each observation (or row) consists of information about a movie. The variables (or columns) contain different pieces of information about a given movie. As all variables for a given row are related to the same movie, the data below is also called relational data.

Title	US Gross	Production Budget	Release Date	Major Genre	Creative Type	Rotten Tomatoes Rating	IMDB Rating
0 The Shawshank Redemption	28241469	25000000	Sep 23 1994	Drama	Fiction	88	9.2
1 Inception	285630280	160000000	Jul 16 2010	Horror/ Thriller	Fiction	87	9.1
2 One Flew Over the Cuckoo's Nest	108981275	4400000	Nov 19 1975	Comedy	Fiction	96	8.9
3 The Dark Knight	533345358	185000000	Jul 18	Action/ Thriller	Fiction	93	8.9

# Where Does Research Time Actually Go?

Studies show researchers spend 60–80% of analysis time on: (Backend)



This isn't about replacing your Python skills, it's about spending that skill where it matters most.

# Hours of Debugging... Solved in Seconds

Researchers traditionally spent hours in a cycle of trial, error, and endless searching. AI changes that completely.

## How to fix error in installing TensorFlow?

Asked 1 year, 5 months ago Modified 1 year, 5 months ago Viewed 443 times

I have the following problem with my TensorFlow installation. I am using Python 3.12.

```

Traceback (most recent call last):
  File "C:\Users\user\AppData\Roaming\Python\Python312\site-packages\tensorflow\python\
    from tensorflow.python._pywrap_tensorflow_internal import *
  ImportError: DLL load failed while importing _pywrap_tensorflow_internal: A dynamic

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "D:\midterm\1st.py", line 5, in <module>
    from deepface import DeepFace
    
```

The screenshot shows search results for the error. The top result is from YouTube by user OSPY, titled 'Tensorflow ImportError DLL load failed The specified module ...'. Below it is a GitHub issue titled 'ImportError: DLL load failed while importing ...' with a description: '17/10/2020 — When i import tensorflow like normal it throw an error, I have done the same thing on my laptop and there it all works fine, ik have the exact ...'. At the bottom, the Google search bar is visible with the text 'Gooooooogle' and navigation buttons.

Same error. 3 hours vs 60 seconds. That's the difference.

# Two Paths to Get Started Today

Choose based on your comfort level:

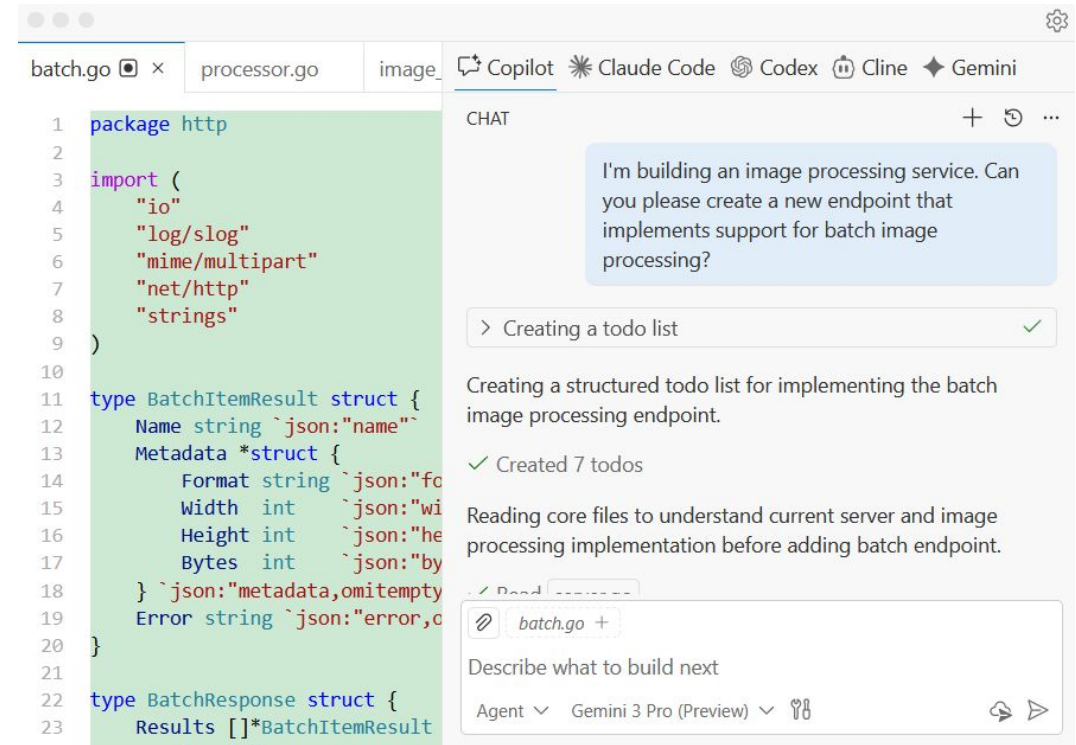
## PATH A: No Setup Needed

- Claude.ai or ChatGPT
- Upload your data, start analyzing
- Best for: quick exploration, interpretation, debugging

## PATH B: Full Power (Setup needed, 15 min)

- Claude Code in VS Code + MCP
- Autonomous pipelines, multi-tool workflows
- Best for: automation, reproducible research

Note that in both paths, data is shared with the agent.



The screenshot shows a VS Code editor with a Go code file named 'processor.go' and the Copilot chat interface. The code defines a batch image processing endpoint. The chat interface shows a user query and the Copilot response.

```
1 package http
2
3 import (
4     "io"
5     "log/slog"
6     "mime/multipart"
7     "net/http"
8     "strings"
9 )
10
11 type BatchItemResult struct {
12     Name string `json:"name"`
13     Metadata *struct {
14         Format string `json:"format"`
15         Width int `json:"width"`
16         Height int `json:"height"`
17         Bytes int `json:"bytes"`
18     } `json:"metadata,omitempty"`
19     Error string `json:"error,omitempty"`
20 }
21
22 type BatchResponse struct {
23     Results []*BatchItemResult
```

CHAT

I'm building an image processing service. Can you please create a new endpoint that implements support for batch image processing?

> Creating a todo list ✓

Creating a structured todo list for implementing the batch image processing endpoint.

✓ Created 7 todos

Reading core files to understand current server and image processing implementation before adding batch endpoint.

batch.go +

Describe what to build next

Agent Gemini 3 Pro (Preview)

# PATH B: Claude Code

## What you need installed:

- VS Code or any other IDE ([code.visualstudio.com](https://code.visualstudio.com))
- Node.js ([nodejs.org](https://nodejs.org))
- Claude Code: `npm install -g @anthropic-ai/claude-code`

## Outputs:

- Building complete data processing pipelines
- Automated data cleaning and transformation
- Batch processing multiple datasets
- Creating reusable analysis scripts
- Complex multi-step analyses
- **MCP** connection

- Everything ran successfully. Here's what was created:

### Folder & Data:

- data/ folder with two benchmark CSVs:
  - iris.csv – 150 samples, 4 features, 3 classes (classic flower classification)
  - wine.csv – 178 samples, 13 features, 3 classes (wine cultivar recognition)

### Models trained (train\_model.py):

Three classifiers were compared on each dataset – Logistic Regression, Random Forest, and Gradient Boosting – with 5-fold cross-validation and an 80/20 train/test split.

### Results:

Dataset	Best Model	Test Accuracy
Iris	Gradient Boosting	96.67%
Wine	Random Forest	100.0%

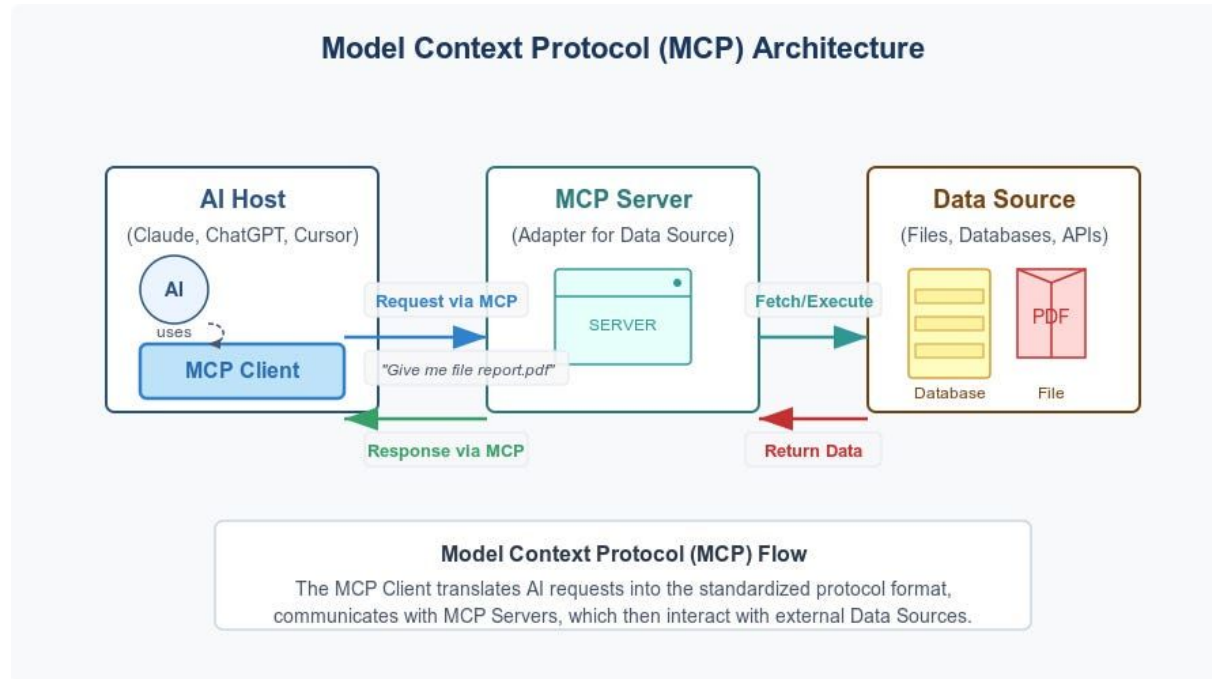
You can re-run anytime with `python train_model.py` from the AI Sessions directory.

\*Cogitated for 58s

> now visualize the results

# PATH B: Connecting Tools via MCP

The bridge that lets Claude Code talk to external tools (your databases, Airtable, the web, GitHub...)



# PATH B: Connecting Tools via MCP

In IDE terminal, e.g. in VS code terminal

Windows/PowerShell users:

- Single quotes may cause issues with `claude mcp add-json`

Workaround: use `$config`

- variable: `$config = '{"type":"stdio","command":"npx",...}'`
- `claude mcp add-json perplexity $config`
- Verify it works: `claude mcp list perplexity` · ✓  
Connected

```
(base) PS C:\Users\LSTM\Desktop\AI Sessions> claude mcp list
Checking MCP server health...

perplexity: cmd /c npx -y @perplexity-ai/mcp-server - ✓ Connected
```

```
(base) PS C:\Users\LSTM\Desktop\AI Sessions> claude

Claude Code v2.1.47
Welcome back Averroès!

Tips for getting started
Run /init to create a CLAUDE.md file with instructions for...

Recent activity
No recent activity

Opus 4.6 · Claude Max
~\Desktop\AI Sessions

> /mcp

Manage MCP servers
1 server

Local MCPs (C:\Users\LSTM\.claude.json [project: C:\Users\LSTM\Desktop\AI Sessions])
> perplexity · ✓ connected

https://code.claude.com/docs/en/mcp for help
↑↓ to navigate · Enter to confirm · Esc to cancel
```

# Agentic AI in Action

Remember Part 1? Here it is working in real time on your research



```

> search perplexity about read and write machine learning paradigm, models that not
only read from data but also write on it

• perplexity - perplexity_research (MCP)(messages: [{"role": "user", "content": "what is the
'read and write' machine learning paradigm? I'm
looking for research on ML models that not only read
from data but also write on it - models that can
modify, augment, or write back to their training
data or external memory. This includes concepts like
neural networks with external memory (Neural Turing
Machines, Differentiable Neural Computers),
memory-augmented neural networks, models that update
knowledge bases, retrieval-augmented generation
with write-back capabilities, and any recent
research on ML systems that actively modify their
data stores rather than just passively consuming
data."}], reasoning_effort: "high")

└ Running...
  
```

• Done. Here's a summary of what I found and fixed:

Issues Identified & Fixed

#	Issue	Details	Fix Applied
1	Column name whitespace	8 columns had leading/trailing spaces (e.g., " BMI ", "Life expectancy ")	Stripped all whitespace
2	Encoding corruption	"Côte d'Ivoire" was garbled	Fixed to proper name
3	False zeros	611 zeros in percentage expenditure (not real zeros - missing data)	Replaced with NaN, then imputed
4	Suspicious Population data	482 values across 87 countries were wildly off (e.g., Afghanistan jumping from 33M to 327K between years)	Flagged outliers (>5x or <0.2x country median), replaced with interpolated values
5	Missing values	2,563 total NaN across 14 columns (Population 22%, Hepatitis B 19%, GDP 15%)	Per-country linear interpolation, then global median for remaining gaps
6	File format mismatch	.xls extension but actual CSV content	Saved clean file as proper .csv

Output Files

- data/Life\_Expectancy\_Clean.csv - cleaned dataset (2,938 rows, 22 cols, 0 missing values)
- data/fix\_life\_expectancy\_data.py - reproducible fix script
- data/fix\_report.txt - full log of all changes

\*Worked for 2m 40s

# Jupyter AI

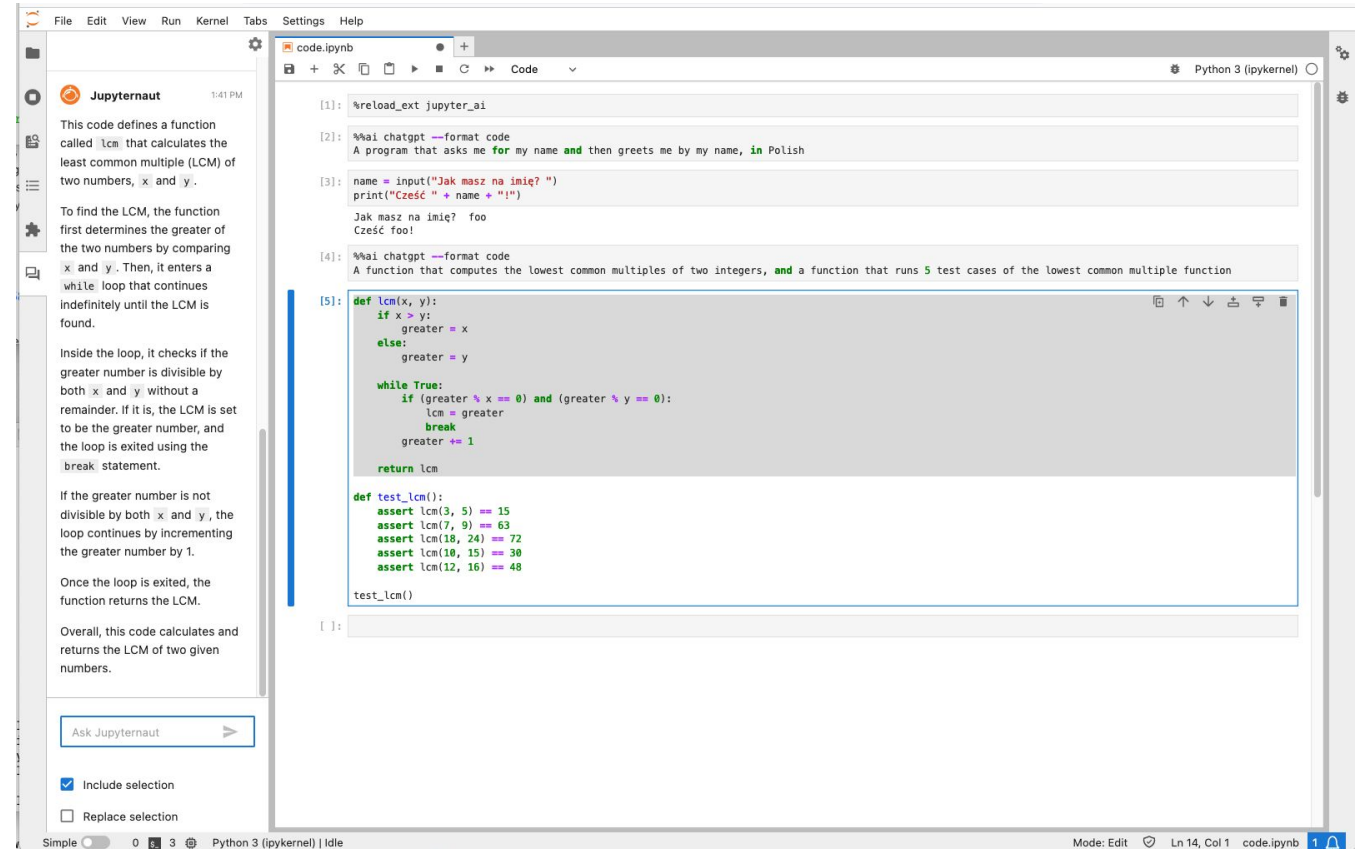
## Jupyter is an In-notebook AI assistant

- Chat with AI inside Jupyter notebooks
- Generate code cells on demand
- Explain data patterns
- Fix errors without leaving notebook

## Install in 2 minutes:

`pip install jupyter-ai`

**Configure your model:** `%ai setup anthropic`



# Cursor

Cursor — when VS Code extensions aren't enough

Best for:

- Very large multi-file research projects
- When you want AI deeply integrated, not added on

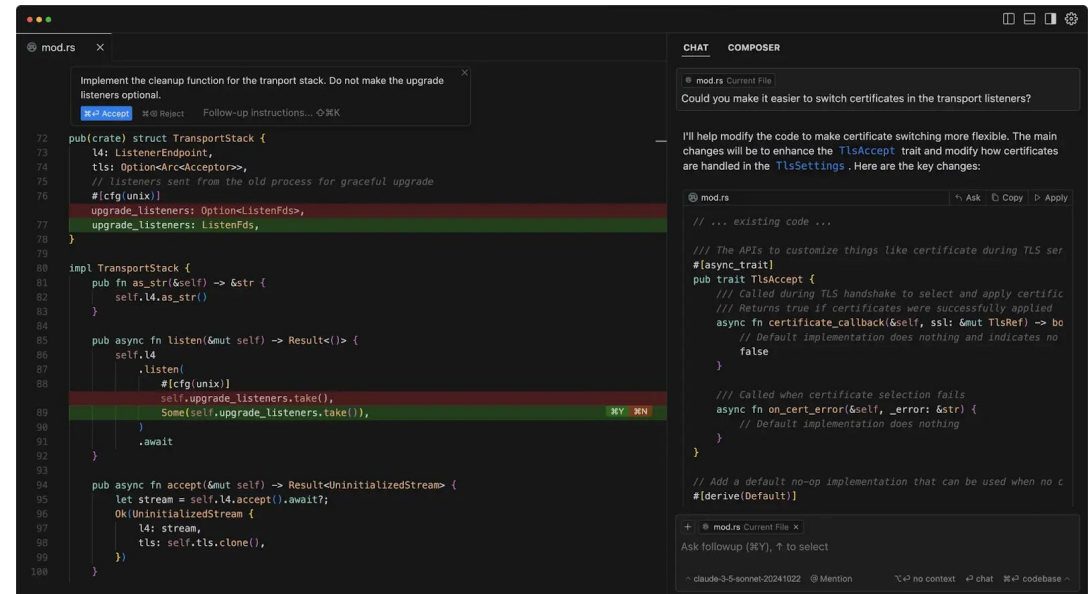
Free tier: [cursor.com](https://cursor.com)

- 2,000 code completions per month
- 50 AI requests (slow speed)
- 14-day Pro trial when you first sign up
- Full editor — no AI limits on basic features

But ...

Pro \$20/month — unlimited completions + Composer

Ultra \$200/month — power users



# LangGraph: Build Your Own Research Agent

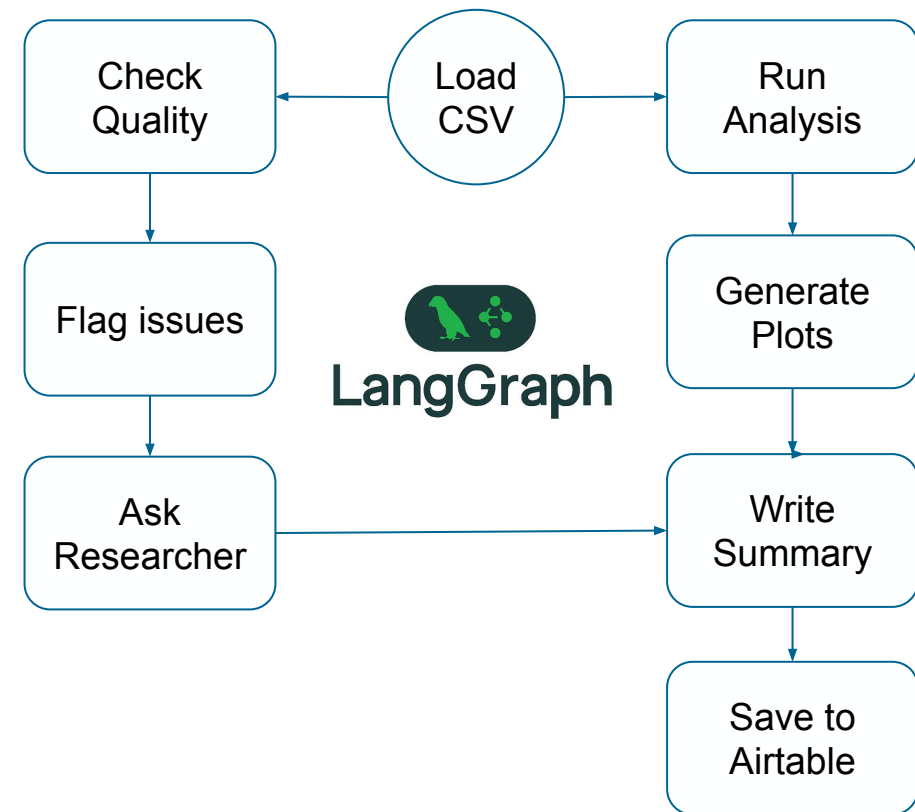
Claude Code and Cursor are tools someone else built. **LangGraph** lets you build your OWN agentic AI workflow tailored to your exact research process.

## Think of it as:

- Nodes = tasks your agent performs
- Edges = decisions between tasks
- Memory = context carried across steps
- Human-in-loop = you approve before it acts

## Install:

```
pip install langgraph
```



# n8n — Agentic Workflows Without Code

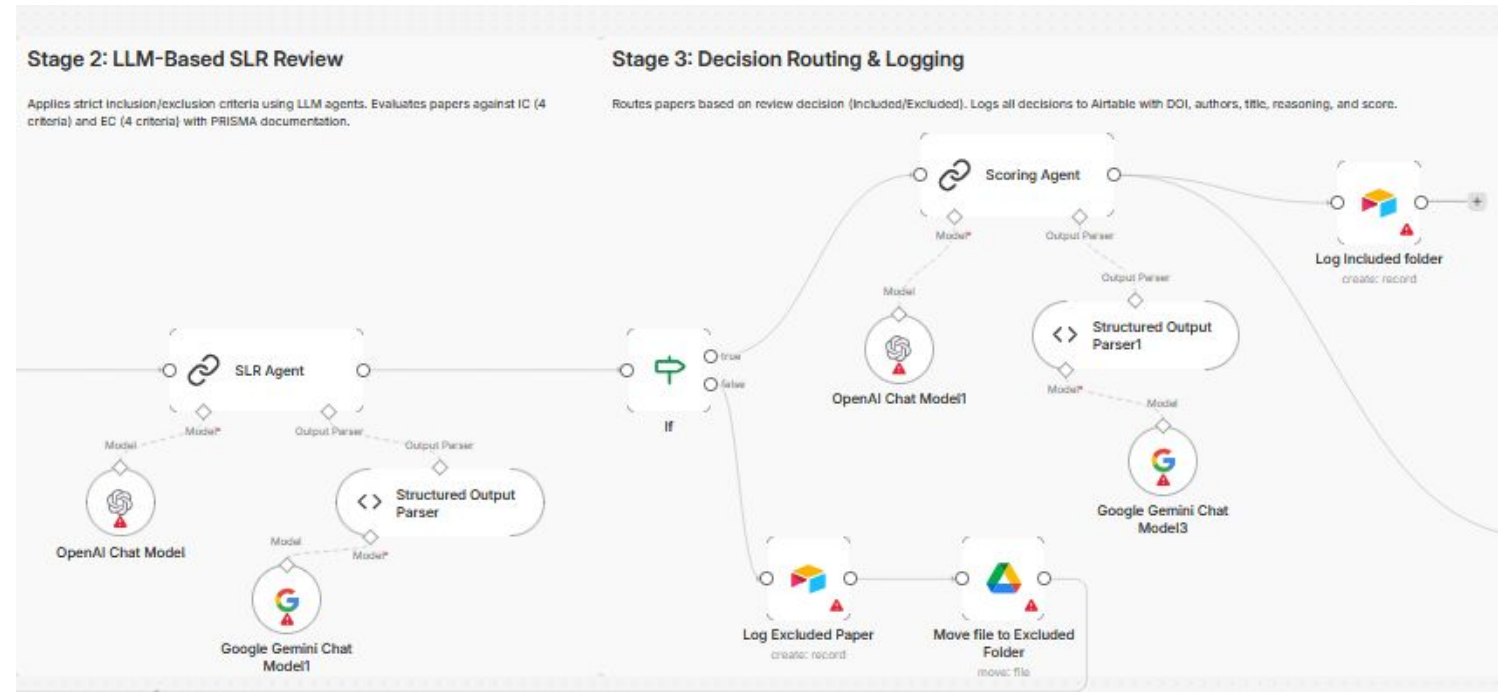
LangGraph requires Python. n8n doesn't. Build the same powerful research automation with drag & drop, no coding required.

n8n is a visual workflow automation tool

- No code required
- 400+ integrations built in
- Self-hosted (your data stays private)  
OR cloud version available
- Free & open source

Install locally:  
npx n8n

Or use cloud: n8n.io



# Make.com - Visual Agent Builder

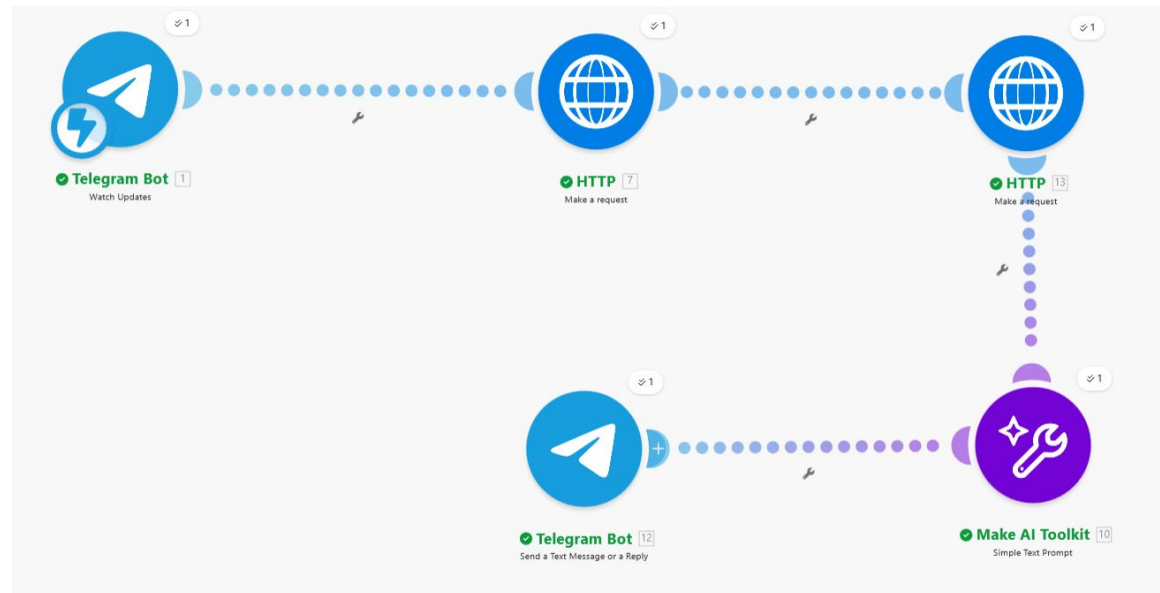
Like n8n, but with more integrations and a stronger focus on AI agents built directly into the workflow canvas. No coding required.

What it is:

- Visual drag & drop workflow + AI agent builder
- 3,000+ app integrations (vs n8n's 400+)
- AI agent decisions visible step by step
- Free tier available: make.com

Research use cases:

- Add AI Agent module
- Select Anthropic as provider
- Requires Anthropic API key



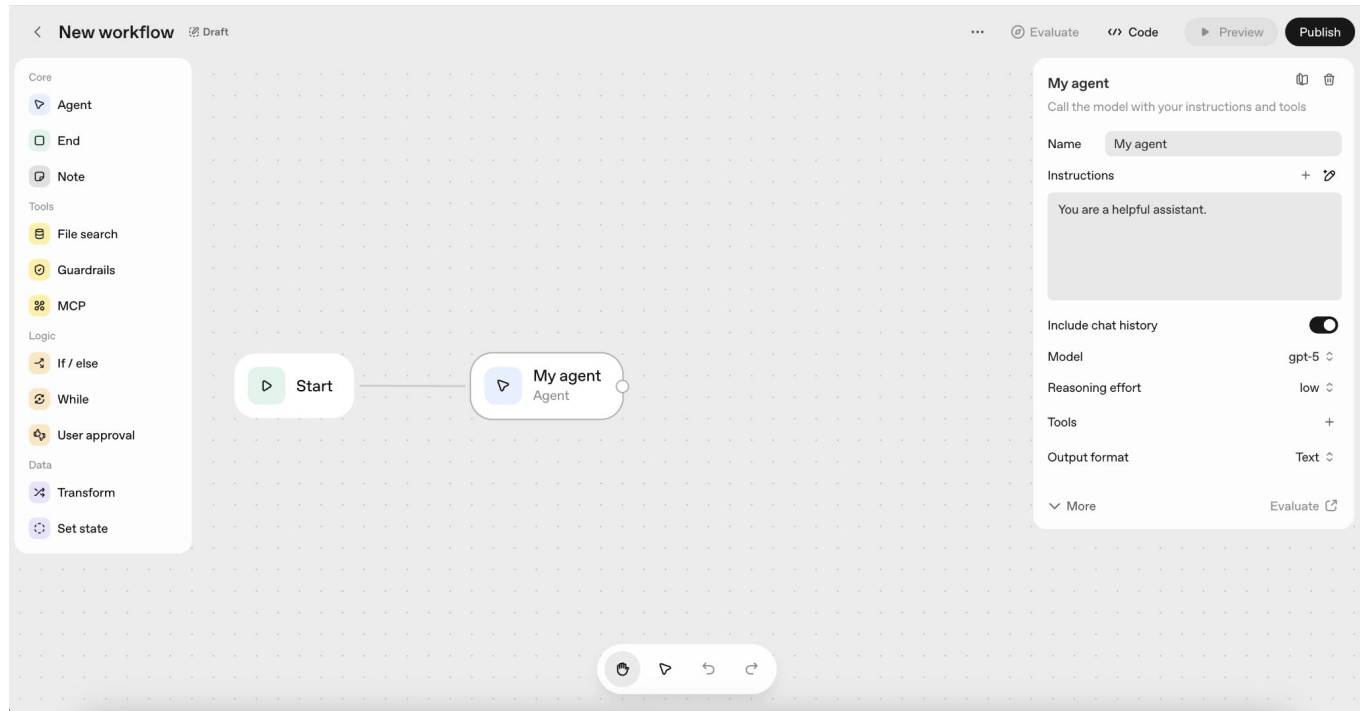
# OpenAI Agent Builder: Visual Agent Canvas

Like Make.com but built by OpenAI. Visual drag & drop agent builder with MCP support and instant code export. No coding required to start.

- Visual node-based agent builder
- Built-in MCP support
- Export to Python/TypeScript code
- Deploy instantly with ChatKit
- Free with OpenAI account

## Nodes available:

- Agent (the AI brain)
- File Search (search your documents)
- Guardrails (safety filters)
- MCP (connect external tools)
- If/else (decision logic)
- While (loops)
- User approval (human in the loop)



# Airtable MCP: Your Research Database Connected to Claude

Stop copying results manually. Claude Code writes directly to your Airtable database while analyzing your data.

What Airtable MCP does:

Without MCP:

- Run analysis in Claude Code
- Copy results manually
- Paste into spreadsheet
- Format and save
- (repeat every time)

With Airtable MCP:

- Run analysis in Claude Code
- Claude saves results automatically
- Database updated instantly
- Team sees results in real time

The screenshot shows the 'Install MCP Server' interface for Airtable. It features a search bar at the top with the Airtable logo and a list of tools: Cursor, Claude, Windsurf, and HTTP. Below the search bar, there is a terminal command box with the instruction: 'Paste and run this command in your terminal to set up Cursor with MCP'. A blue overlay with white text provides a 6-step installation guide:

1. Go to [mcp.composio.dev](https://mcp.composio.dev)
2. Search "Airtable"
3. Click "Connect"
4. Copy the MCP command
5. Run in Claude Code terminal
6. Done!

The background interface also displays the Airtable logo, a description of Airtable as a productivity and project management tool, and a table with columns for 'VIEWS' (823) and 'USERS' (274). Below the table, it shows the category 'Productivity & Project Management' and the version '0.1.0 (Updated 12h ago)'. At the bottom, there is a 'List Bases' section with a description: 'Retrieves all airtable bases accessible to the authenticated user, which may include an 'offset' for pagination.' and a note '1 action available'.

## PART 3: AI Tools for Interpretation

# PandasAI - Natural Language Data Queries

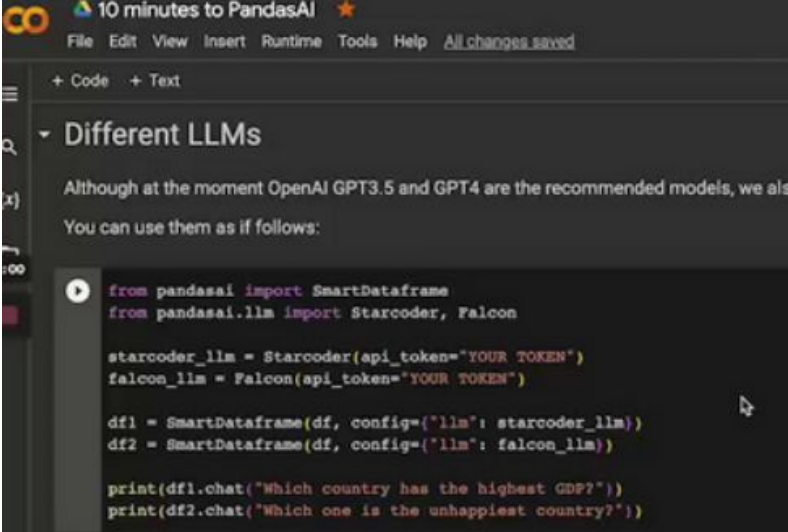
pip install **pandasai**

```
from pandasai import SmartDataframe
```

```
df = SmartDataframe("data.csv", config={"llm": "openai"})
```

## Natural language queries:

- `df.chat("What are the top 5 correlations in this dataset?")`
- `df.chat("Show me outliers in the dependent variable")`
- `df.chat("Create a correlation heatmap excluding variables with >50% missing data")`
- `df.chat("Which features best predict outcome? Run feature importance")`



The screenshot shows a code editor window titled "10 minutes to PandasAI". The code in the editor is as follows:

```
from pandasai import SmartDataframe
from pandasai.llm import Starcoder, Falcon

starcoder_llm = Starcoder(api_token="YOUR TOKEN")
falcon_llm = Falcon(api_token="YOUR TOKEN")

df1 = SmartDataframe(df, config={"llm": starcoder_llm})
df2 = SmartDataframe(df, config={"llm": falcon_llm})

print(df1.chat("Which country has the highest GDP?"))
print(df2.chat("Which one is the unhappiest country?"))
```

Below the code, there is a section titled "Different LLMs" with the following text:

Although at the moment OpenAI GPT3.5 and GPT4 are the recommended models, we also support other LLMs like Starcoder and Falcon. You can use them as if follows:

Supports: OpenAI, Claude, HuggingFace, local models

# Sweetviz & D-Tale - Instant EDA Reports

```
pip install sweetviz
```

```
import sweetviz as sv
```

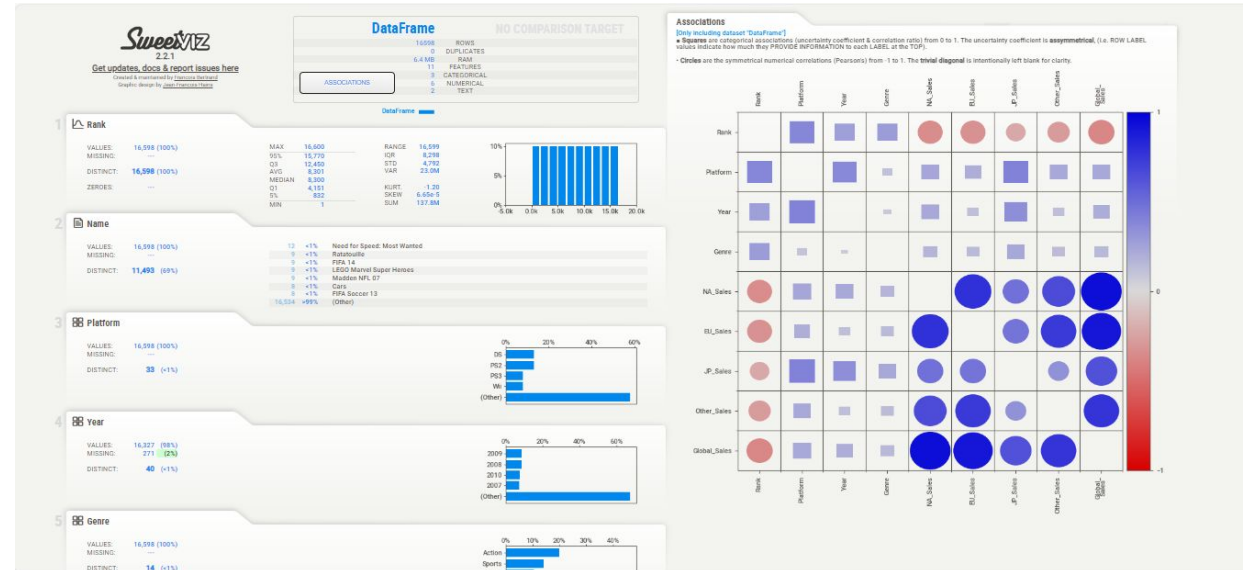
```
report = sv.analyze(df)
```

```
report.show_html("report.html")
```

Generates:

- Univariate analysis for all variables
- Correlations with target variable
- Missing data patterns
- Distribution comparisons
- Association matrices

Perfect for: Quick data understanding, sharing with collaborators



# Beautiful Tables Without Pain

pip install **great-tables** (or **itables**)

```
from great_tables import GT
```

```
(GT(results_df)
 .tab_header(title="Regression Results")
 .fmt_number(columns=["coef", "se"], decimals=3)
 .tab_style(
   style=style.fill(color="lightblue"),
   locations=loc.body(rows=lambda x: x["p"] < 0.05)
 )
 .tab_source_note("* p < 0.05")
 )
```

Export to: HTML, LaTeX, Word, PNG

S&P 500					
2010-06-07 to 2010-06-14					
date	open	high	low	close	volume
Mon, Jun 14, 2010	\$1,095.00	\$1,105.91	\$1,089.03	\$1,089.63	4.43B
Fri, Jun 11, 2010	\$1,082.65	\$1,092.25	\$1,077.12	\$1,091.60	4.06B
Thu, Jun 10, 2010	\$1,058.77	\$1,087.85	\$1,058.77	\$1,086.84	5.14B
Wed, Jun 9, 2010	\$1,062.75	\$1,077.74	\$1,052.25	\$1,055.69	5.98B
Tue, Jun 8, 2010	\$1,050.81	\$1,063.15	\$1,042.17	\$1,062.00	6.19B
Mon, Jun 7, 2010	\$1,065.84	\$1,071.36	\$1,049.86	\$1,050.47	5.47B

# Jupyter Lab Extensions Power Combo

## **jupyterlab-lsp (Language Server Protocol)**

- Real-time code suggestions, linting, formatting
- `pip install jupyterlab-lsp python-lsp-server`

## **jupyter-ai**

- Chat interface inside notebooks
- Generate cells, explain code, debug
- `pip install jupyter-ai`

## **jupyterlab-execute-time**

- Track cell execution duration
- Find bottlenecks in analysis

## **nbdime**

- Git diff for notebooks
- Track changes in analyses over time

## **jupyterlab-code-formatter**

- Auto-format code (black, autopep8)
- One click to clean code

## **RISE**

- Present notebooks as slides
- Live code demos in presentations

## PART 4: Live Demo Exercise

# Vibe Coding

Agentic AI Data Analysis

WHO Life Expectancy Dataset

<b>20 min</b> Live demo	<b>7 Steps</b> Follow along	<b>Claude Code</b> VS Code terminal
----------------------------	--------------------------------	--

**WE ARE SCIENCE.  
WE ARE TECHNOLOGY.  
WE ARE INNOVATION.  
WE ARE INESC TEC.**

FUNDING



PROJECT FUNDING

