Build an AI agent that:

* Takes a topic (e.g., “renewable energy advances 2025”)
* Searches the web or a provided text source
* Extracts key points
* Summarizes them neatly into a formatted report (with sources)

You’ll learn how to:

* Build an agent loop (task → action → result → refinement)
* Use Gemini Flash for reasoning and summarization
* Integrate basic web or file tools
* Manage prompt context and outputs

**Stretch Goals (if you finish early)**

* Add **user memory** (so it remembers prior research topics).
* Add a **GUI** (using Streamlit or Flask).
* Add **multiple modes**: quick summary vs deep dive.
* Export reports as **PDFs**.

| **Day** | **Focus** | **What You’ll Do** |
| --- | --- | --- |
| 1 | **Setup** | Get environment ready and test Gemini Flash API. |
| 2 | **Core Agent Logic** | Build a simple prompt loop for: “Think → Act → Reflect.” |
| 3 | **Search Tool** | Add a simple search function (Google or DuckDuckGo API). |
| 4 | **Summarization** | Format and refine summaries using Gemini. |
| 5 | **Memory / Storage** | Let agent keep short-term context or notes (JSON or text file). |
| 6 | **Output Formatting** | Make clean Markdown or HTML report. |
| 7 | **Polish & Test** | Run a few topics, adjust prompts, finalize. |

A white background with black text

AI-generated content may be incorrect.

research\_agent/

│

├── main.py

├── agent.py

├── search\_tool.py

├── summarizer.py

├── config.py

└── requirements.txt

pip install google-generativeai duckduckgo-search rich

from duckduckgo\_search import DDGS

import google.generativeai as genai

genai.configure(api\_key="YOUR\_API\_KEY")

def search\_topic(topic):

results = DDGS().text(topic, max\_results=5)

return [r['body'] for r in results]

def summarize\_text(text, model="gemini-1.5-flash"):

prompt = f"Summarize the following research clearly and concisely:\n\n{text}"

response = genai.GenerativeModel(model).generate\_content(prompt)

return response.text

def research\_agent(topic):

snippets = search\_topic(topic)

summaries = [summarize\_text(s) for s in snippets]

final\_summary = summarize\_text("\n".join(summaries))

with open("report.md", "w") as f:

f.write(f"# {topic}\n\n{final\_summary}")

print("✅ Report generated!")

research\_agent("AI in healthcare 2025")

setx GEMINI\_API\_KEY "AIzaSyDG52srZzO665OSazAEyQHWwR\_Xc21UVpY" # Windows

echo $env:GOOGLE\_API\_KEY

pip install -r requirements.txt

python main.py

You can confirm your version:

pip show google-generativeai

If it’s **1.0.0 or newer**, use "models/gemini-1.5-flash".  
If it’s **below 1.0.0**, use "gemini-1.5-flash-latest".

1. **models/gemini-flash-latest**: This is likely the most current and recommended "flash" model for general use if you want something fast and cost-effective.
2. **models/gemini-pro-latest**: This is the latest "pro" model. Since gemini-pro specifically didn't work, using gemini-pro-latest might provide access to a newer, compatible version.
3. **models/gemini-2.5-flash**: A specific version of the Flash model, likely stable.
4. **models/gemini-2.5-pro**: A specific version of the Pro model, also likely stable.