How to Build an AI Agent with n8n for Free

AI Automation & Agents · about 22 minutes · 2026-11-11

Install n8n with Docker, connect a free Gemini or Groq model, and build an AI agent that remembers, uses tools, and runs on a schedule. A complete 30-step build that costs nothing.

Things you will need

- A computer with 8 GB of RAM and 2 GB of free disk space
- Docker Desktop on Windows or macOS, or Docker Engine on Linux
- n8n community edition, self-hosted and free
- A Google account and a free Gemini API key from AI Studio
- Optional: a free Groq or OpenRouter account as a backup provider
- A Gmail or Telegram account for delivering the agent's output
- A password manager for the owner account and the API keys
- Sixty to ninety minutes for the first complete build

Install n8n on your own computer

1. **Check your computer before you install**
   Confirm your machine has at least 8 GB of RAM and 2 GB of free disk before downloading anything, because Docker plus the n8n image needs that headroom to run comfortably.
2. **Install Docker Desktop**
   Download Docker Desktop from the official site and install it with the default options; on Windows it will ask to enable WSL 2, which you should accept.
3. **Run n8n with one Docker command**
   Create a persistent volume with docker volume create n8n_data, then start n8n with docker run -it --rm --name n8n -p 5678:5678 -v n8n_data:/home/node/.n8n docker.n8n.io/n8nio/n8n.
4. **Create the owner account at localhost:5678**
   Open http://localhost:5678 in your browser and create the owner account when n8n asks for an email and password on first launch.
5. **Learn the three parts of the canvas**
   Spend two minutes on the empty workflow before building: nodes are the boxes that do things, connections are the lines that pass data between them, and the Executions tab records every run.
6. **Set the timezone before any schedule**
   Stop the container and restart it with -e GENERIC_TIMEZONE=Europe/Berlin -e TZ=Europe/Berlin added to the docker run command, using your own zone.

Connect a free AI model

7. **Get a free Gemini API key**
   Sign in at Google AI Studio with any Google account and create an API key; the Gemini API free tier needs no credit card and is enough for a personal agent.
8. **Line up a backup free provider**
   Create a free account at Groq and note OpenRouter's free models as a second fallback, because

free tiers have daily caps and a single provider can throttle you mid-task.

9. **Store the key as an n8n credential**

   In n8n, open Credentials, choose Add credential, search for Google Gemini(PaLM) Api, and paste the key there rather than into any node field.

10. **Add the AI Agent node**

    Click the plus button, search for AI Agent, and add it; it lives in the Advanced AI section and is the brain of this build.

11. **Attach a chat model sub-node**

    Click the Chat Model slot under the agent and choose Google Gemini Chat Model, then select the credential you saved and a Flash-family model such as gemini-flash-latest.

12. **Test the model with a one-line prompt**

    Open the agent node, type Say hello in exactly five words into the chat panel, and run it; a coherent reply means the key, credential, and model are wired correctly.

Give the agent a job, memory, and tools

13. **Write a system prompt that defines the job**

    Open the agent's Options, add a System Message, and write the job in plain English: the role, the task, the output format, and the limits.

14. **Add Simple Memory so it remembers**

    Click the Memory slot under the agent and add Simple Memory, then set the context window to around ten messages.

15. **Give it an HTTP Request Tool**

    Click the Tool slot, add the HTTP Request Tool, and write a tool description that tells the model when to use it, for example Fetch the full text of a URL the user provides.

16. **Add Wikipedia and Calculator tools**

    Add the Wikipedia tool for factual lookups and the Calculator tool for arithmetic, because language models guess numbers and dates unless a tool computes them.

17. **Cap iterations and treat web content as untrusted**

    Set the agent's maximum iterations to five or ten so a confused model cannot loop forever and burn your free quota in minutes.

18. **Chat-test the finished agent**

    Add a Chat Trigger node, connect it to the agent's input, and click Open Chat to talk to your build in a proper chat window.

Put the agent on autopilot

19. **Swap chat for a Schedule Trigger**

    Delete the Chat Trigger and add a Schedule Trigger set to every day at 08:00, so the agent works while you sleep.

20. **Fetch fresh input with RSS Feed Read**

    Add an RSS Feed Read node between the schedule and the agent, and give it one or two feed URLs you actually read, such as a favourite blog or a news section.

21. Or trigger it from anywhere with a Webhook

Add a Webhook node as an alternative trigger when you want the agent to run on demand, for example from a phone shortcut or another app.

22. Feed the data into the prompt with expressions

In the agent's prompt field, switch to expression mode and reference the incoming items, for example Summarise these articles: {{ $json.title }} with links {{ $json.link }}.

23. Deliver the answer to Gmail or Telegram

Add a Gmail node after the agent to email the summary to yourself, or a Telegram node to push it to your phone; Telegram is simpler because a bot token from BotFather is all it needs.

24. Catch failures before they vanish

Open each node's Settings tab and enable Retry On Fail with three attempts five seconds apart, so a transient network blip does not kill the morning run.

Test, activate, and keep it free

25. Read the execution log like a debugger

Open the Executions tab, click the latest run, and inspect each node's input and output JSON by clicking it on the canvas.

26. Pin test data and iterate the prompt

Pin the RSS node's output so every test run uses the same articles instead of fetching new ones, then edit the system prompt and rerun until the summary format is exactly right.

27. Activate and verify one unattended run

Flip the Active toggle in the top-right corner, then wait for the next scheduled time and confirm a successful execution appears without you touching anything.

28. Stay inside the free-tier limits

Check your provider's current requests-per-minute and requests-per-day caps on the Gemini pricing page, and keep your schedule well under them.

29. Export and back up the workflow

Open the workflow's three-dot menu and download it as JSON, then store that file with your other backups; importing it into any n8n instance restores the whole build in seconds.

30. Keep the instance private and updated

Never port-forward 5678 to the internet; an exposed n8n is an open door to every credential inside it.

Sources

1. Install n8n with Docker
   https://docs.n8n.io/hosting/installation/docker/
2. AI Agent node documentation
   https://docs.n8n.io/integrations/builtin/cluster-nodes/root-nodes/n8n-nodes-langchain.agent/
3. Advanced AI in n8n
   https://docs.n8n.io/advanced-ai/
4. Gemini API pricing and free tier
   https://ai.google.dev/gemini-api/docs/pricing
5. Docker Desktop overview
   https://docs.docker.com/desktop/
6. MediaWiki API main page
   https://www.mediawiki.org/wiki/API:Main_page

7. n8n community forum
   https://community.n8n.io/

Written as a checklist from https://www.howcompass.com/how-to/Build-an-AI-Agent-with-n8n-for-Free. Corrections: https://www.howcompass.co