

Unit 2 - Model choice and advanced settings



- ☰ 2.1 Unit Introduction
- ☰ 2.2 Choosing the LLM
- ☰ 2.3 Advanced settings in Make AI agent
- ☰ 2.4 Wrap up



Unit 2

Model choice and advanced settings

2.1 Unit Introduction

Welcome to the second unit of Tools and AI agent's settings optimization course!

In unit 1, you learned best practices for configuring and optimizing your AI agent's tools.

Now, you will cover how to properly configure the rest of your AI agent settings in Make.

In this unit, you'll learn:

how to choose the LLM to improve your AI agent's performance

what are the advanced settings of Make's AI agent

how to adjust advanced settings

Let's begin!

[Continue to 2.2: Choosing the LLM](#)



2.2 Choosing the LLM

When you configure an AI agent, one of the most critical decisions you make is which **large language model (LLM)** powers it.

The model you select determines three critical performance factors:



Capability

How well your AI agent handles the tasks you need.



Cost

How much each interaction costs depending on the model.



Speed

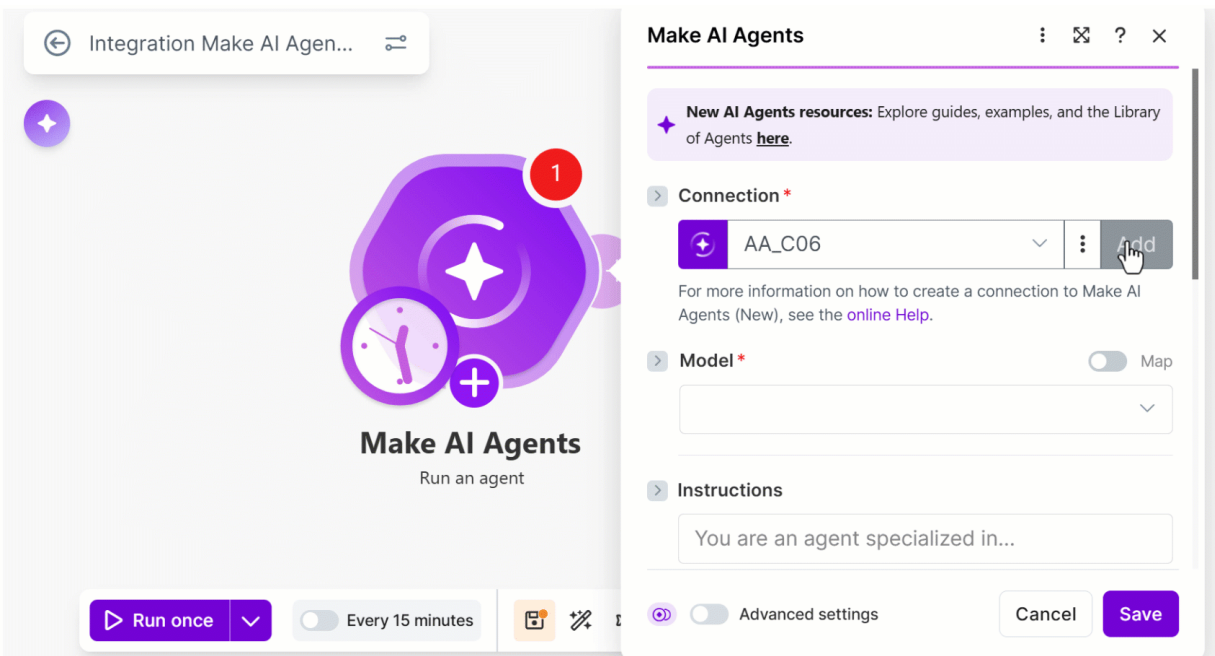
How quickly your AI agent responds to requests.



You learned how to connect the LLM in the [AI agents in Make](#) course. You can review the course if you need to.

Not every task requires the most advanced (and expensive) model available.

Your goal is to match the model to your AI agent's task requirements, balancing capability, cost, and speed. It's about picking the best model for the task.



In Make, you configure your model choice through the **Run an agent** module. You'll first connect to an AI provider (Make's AI provider, OpenAI, Anthropic Claude, Gemini AI, etc.), then select the specific model from that provider's available options.

Each provider offers models with distinct capability levels, pricing structures, and performance characteristics.



Pro Tip

Make shows a **description** under each model option. Use these descriptions to narrow down which model suits your task before you

start testing.



Let's walk through a practical approach you can follow to figure out which model fits your specific needs.

[Continue to 2.2.1: How to select the right model](#)

When choosing the model, your goal is to use the least powerful model that still completes the tasks accurately.

Steps to follow to decide which model is right for your AI agent:



Step 1

Evaluate the tasks' requirements



Step 2

Start big, test the most capable model



Step 3

Optimize down, test smaller models

Step 1: Evaluate the tasks' requirements

1

Step 1 - Evaluate the tasks' requirements

Before you can start testing your AI agent with any models, you need to analyze what tasks your AI agent actually needs to accomplish.

Here's how you can do it:

Start by identifying the task basic needs

Assess what your task requires:



Interpreting and following instructions



Reasoning through problems step-by-step



Maintaining context across interactions



Using tools effectively

Consider if you need more advanced or specialized capabilities

Determine if you need:



Code writing and debugging



Mathematical or quantitative analysis



Multi-language processing



Domain expertise (legal, medical, technical fields)

Now that you've evaluated your agent's tasks requirements, you're ready to start testing.

Step 2: Start big, test the most capable model

It's time to establish what **good performance** actually looks like for your specific use case.

Begin testing with the most advanced model available. This approach establishes the best possible outcome for your tasks, which becomes your reference point when evaluating other models. Then you can test progressively smaller models, reducing cost and improving speed, until you find the minimum capability that still delivers acceptable results.

- 1 Select the most advanced model:** set up your AI agent using the most capable model available.
- 2 Run comprehensive tests:** execute tasks your agent will actually perform (e.g. if it categorizes emails, test with real email examples). This establishes your baseline, the best possible model performance.

3

Evaluate and iterate: review accuracy, speed, and cost in execution logs. Then test progressively smaller models until you find the optimal balance.

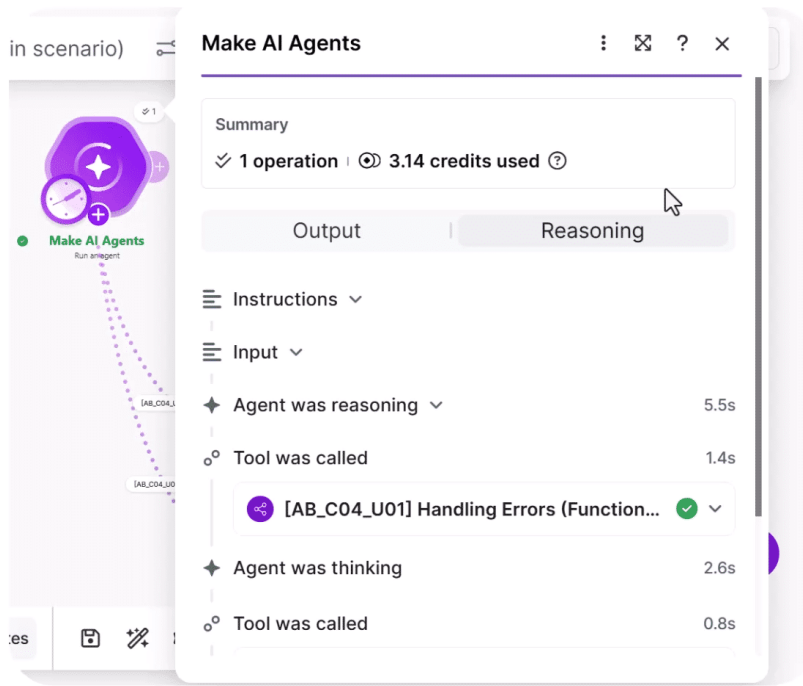
Here's what to evaluate and where to find it in Make after you run your tests.

Click each one to learn more.

Accuracy —

Did the AI agent complete the tasks correctly? Did it call the right tools? Was the output accurate?

After you run **Make AI agent**, you can check the result and tool call in the **Reasoning** and **Output** tabs.

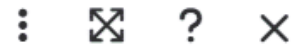


Cost

How many tokens did each interaction use?

You can check the **output** to review the **credit usage** and see the **AI token usage**.

Make AI Agents



Summary

✓ 1 operation | 3.14 credits used ⓘ

Output

Reasoning

✓ Initialization

✓ Operation 1 ▲

3.14 4.9 KB

INPUT



+ Bundle 1: (Collection)

OUTPUT

+ Bundle 1: (Collection)

METADATA

– Empty

CREDIT USAGE

- Credits used for: (Collection)

Operation cost: 1 credit

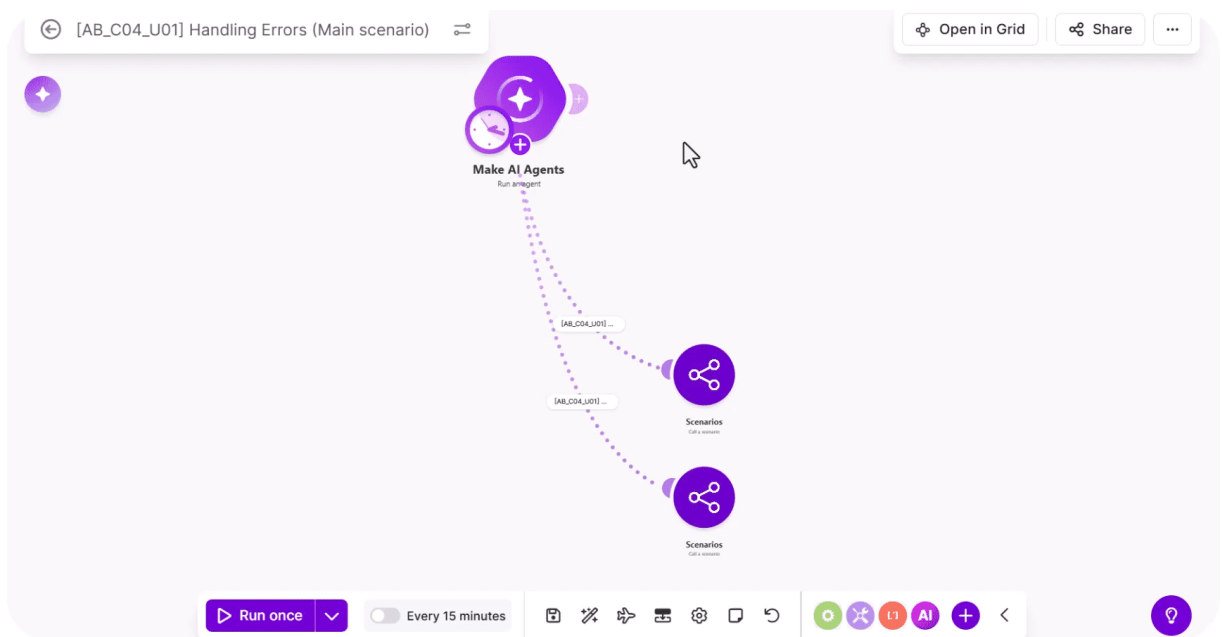
AI tokens: 2.14 credits

✓ Commit

Speed —

How long did each run take?

After you run your tests, check the **execution history**, this will show you how much time each run took.



You now know what the most capable model can achieve for your agent's tasks. You have data about accuracy, speed, and cost.

But here's the key question: **Do you actually need that level of capability, or can a less expensive model deliver acceptable results?**

Step 3 helps you find out.

Step 3: Optimize down, test smaller models

3

Step 3: Optimize down, test smaller models

You tested the most capable model. **Now you need to test smaller models to find the right balance between performance and cost.** Here is where you will use the assessment you did in Step 1. It will set the boundaries for how far down you can optimize:

- **If your task only needs basic capabilities**, you have significant room to optimize. You can test much smaller models and likely still maintain quality.
- **If your task requires advanced or specialized capabilities** (like code generation), you must stay within models that have those capabilities.

Select a less powerful model from the dropdown



Change only the model selection, keep all other agent configuration the same



Run your identical test



Evaluate the same three metrics: accuracy, cost, and speed



What's the advantage of using a smaller model?

Smaller models cost significantly less and typically run faster.

Testing smaller models helps you find the sweet spot, the least powerful model that still meets your quality requirements. You avoid paying for capability you don't need. You also ensure your model can handle the tasks reliably.

This three-step approach helps remove guesswork from model selection. By starting at the top and working down, you establish what's possible before making compromises. Most importantly, you optimize the model for your specific use cases.

After you choose the model, you can configure the advanced settings that control how your AI agent operates within Make.

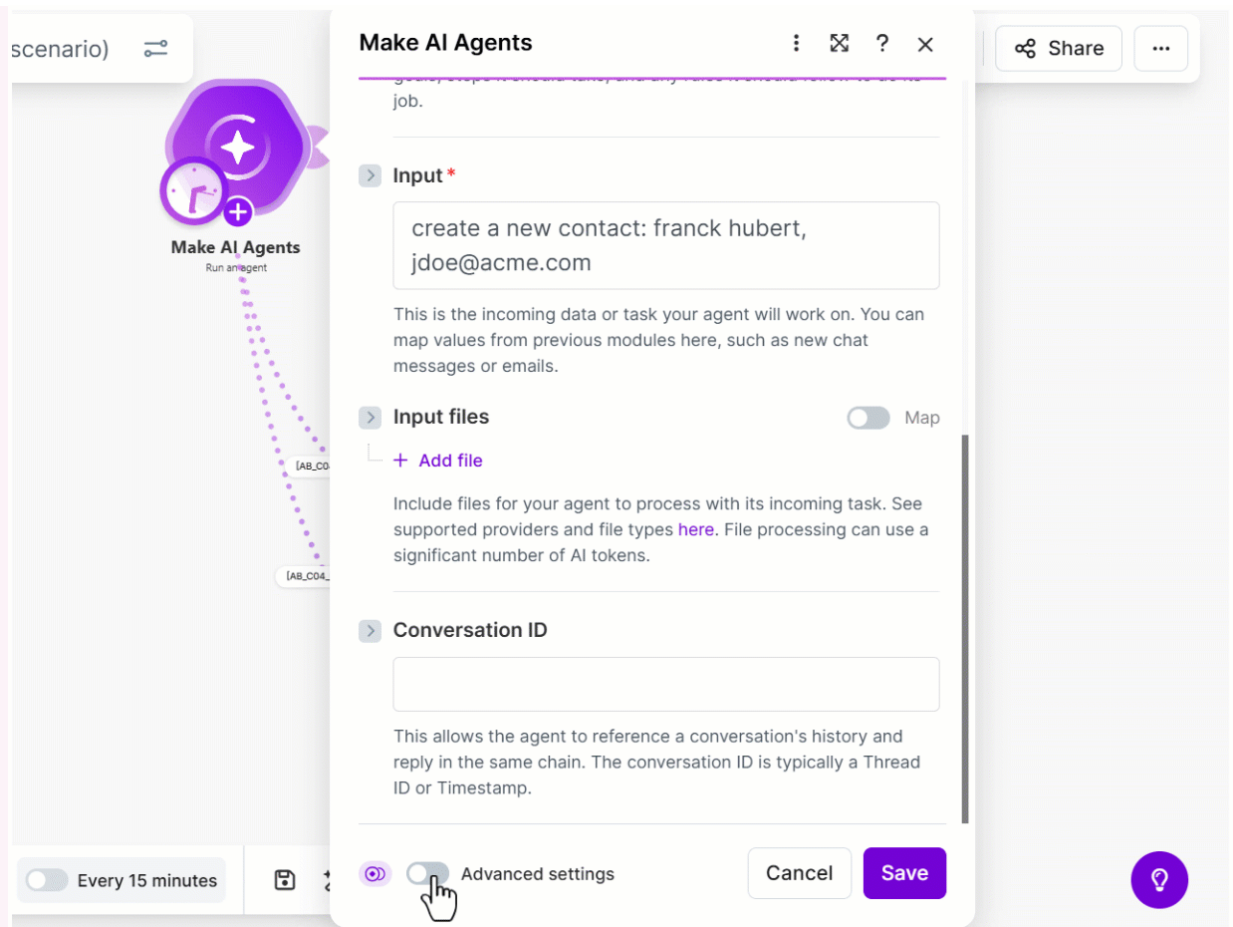
[Continue to 2.3: Advanced settings in Make AI agent](#)



2.3 Advanced settings in Make AI agent

Advanced settings let you optimize how your AI agent functions by fine-tuning its behavior and customizing its performance.

These settings are:



1. **Maximum conversation history:** defines the maximum number of replies the agent can use as context in a conversation.
2. **Step timeout:** limits the maximum time the agent should take to reason.
3. **Response format:** determines how your AI agent structures its responses.

Let's explore each one in more detail.

Continue to 2.3.1: Maximum conversation history

2.3.1 Maximum conversation history



Maximum conversation history controls how many past interactions your AI agent can remember. This setting limits the number of previous interactions your AI agent can access from the conversation thread.

When you use **conversation ID**, Make stores conversation history. Each time your AI agent runs, Make adds all previous interactions within your set limit into the context window. This is where the **maximum**

conversation history setting comes in, it determines how far back your agent can look.

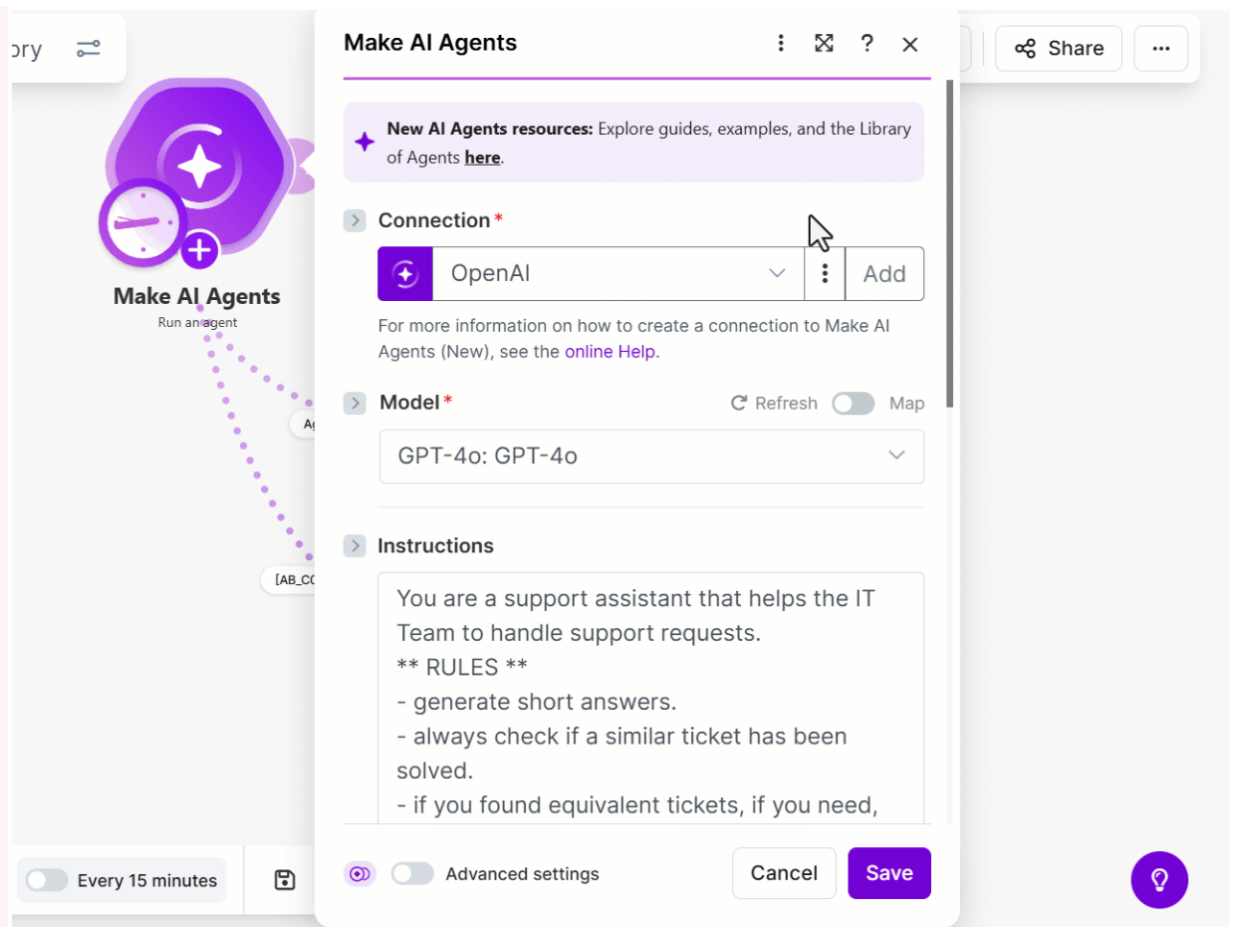


Remember, you covered all about conversation ID in the [Information management and security](#) course.



Pro Tip

Unlimited conversation history consumes excessive tokens and increases costs with every call. Setting the right limit balances context retention with cost efficiency. Your AI agent maintains the memory it needs without processing irrelevant old messages.



Make sets the default at 10 replies.

This works well for most use cases, but you can adjust based on your conversation patterns.

Here are the options to consider when you need to adjust this setting:



Increase (12-20)

When conversations span many interactions and users reference information from early exchanges.

Example: Technical support agent troubleshooting complex issues.



Keep default (10)

When you need recent context but older messages become irrelevant.

Example: E-commerce support agent helping customers track orders, needs to remember the order number and recent discussion.



Decrease (1-3)

When each interaction is independent and cost efficiency is critical.

Example: Invoice data management agent processes an invoice, asks for approval if the amount is high, then takes action based on the response. It only needs to remember 1-2 previous exchanges, not the full conversation history.

Let's see how this works in practice.

[Continue to see a practical example](#)

Practical example



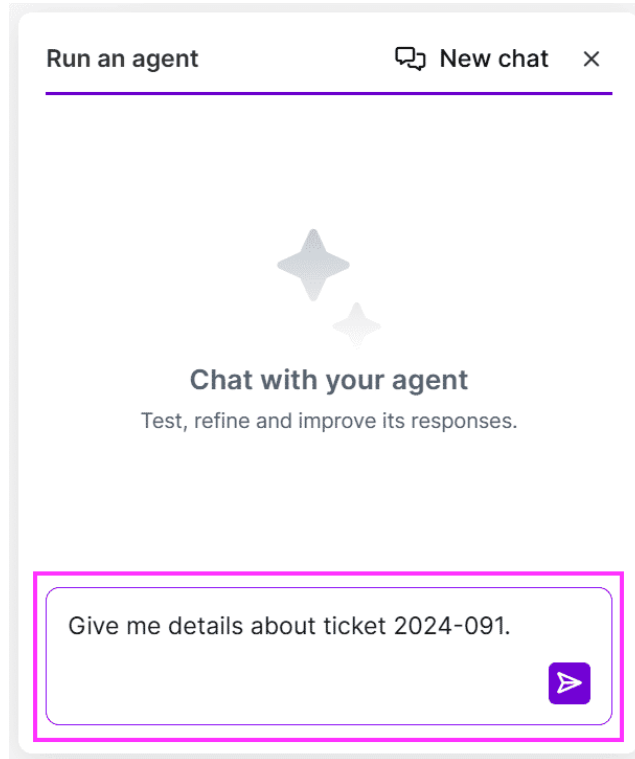
A support assistant AI agent helps IT engineers get information about support tickets through chat. It needs to remember context across multiple questions in the same conversation so engineers don't have to repeat information.

Let's see what happens when you configure the agent to remember a **maximum of one reply in conversation history**.

Work through each stage before you continue.

Step 1

(1/4) Maximum conversation history: 1



A user asks their first question: **Give me details about ticket 2024-091.**

Step 2

(2/4) Maximum conversation history: 1

Run an agent New chat ×

Give me details about ticket 2024-091.

✦ Agent was thinking 1.4s

Tool
Agent Knowledge ✓ ∨

✦ Agent was thinking 1.9s

Ticket 2024-091 addressed an issue with a laptop fan running at maximum speed constantly, making the computer very loud even when idle. The resolution involved detecting and removing cryptocurrency miner malware using an antivirus scan, cleaning startup programs, which normalized CPU usage, and resulted in the fan running quietly again.

The agent searches the system, finds the ticket, and returns the details.

Step 3

(3/4) Maximum conversation history: 1

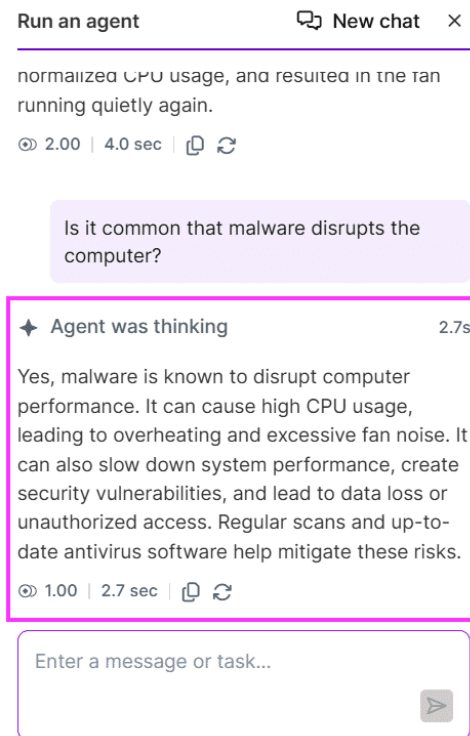
The screenshot shows a chat window titled "Run an agent" with a "New chat" button. The chat history includes:

- An "Agent was thinking" status for 1.4s.
- A tool call for "Agent Knowledge" (marked as successful with a green checkmark).
- Another "Agent was thinking" status for 1.9s.
- A text response: "Ticket 2024-091 addressed an issue with a laptop fan running at maximum speed constantly, making the computer very loud even when idle. The resolution involved detecting and removing cryptocurrency miner malware using an antivirus scan, cleaning startup programs, which normalized CPU usage, and resulted in the fan running quietly again." Below this text are icons for a speech bubble, a timer (2.00 | 4.0 sec), a copy icon, and a refresh icon.
- A new user message in a pink-bordered box: "Is it common that malware disrupts the computer?" with a send button.

The user then asks a follow-up question: **Is it common that malware disrupts the computer?**

Step 4

(4/4) Maximum conversation history: 1



The screenshot shows a chat interface with a header "Run an agent" and a "New chat" button. The previous message is "normalized CPU usage, and resulted in the fan running quietly again." with a timestamp of 2.00 and a duration of 4.0 sec. The current message is "Is it common that malware disrupts the computer?" with a timestamp of 1.00 and a duration of 2.7 sec. The response is "Agent was thinking" followed by a detailed answer about malware. The input field at the bottom contains the text "Enter a message or task..." and a send button.

Run an agent New chat ×

normalized CPU usage, and resulted in the fan running quietly again.

2.00 | 4.0 sec |

Is it common that malware disrupts the computer?

✦ Agent was thinking 2.7s

Yes, malware is known to disrupt computer performance. It can cause high CPU usage, leading to overheating and excessive fan noise. It can also slow down system performance, create security vulnerabilities, and lead to data loss or unauthorized access. Regular scans and up-to-date antivirus software help mitigate these risks.

1.00 | 2.7 sec |

Enter a message or task...

The agent uses context from the previous interaction to provide a relevant answer. So far, the agent is working as expected.



Now let's see what happens when the conversation history limit is too low.

Step 1

(1/2) Maximum conversation history: 1

Run an agent New chat ×



normalized CPU usage, and resulted in the fan running quietly again.


2.00 | 4.0 sec |  

Is it common that malware disrupts the computer?

✦ Agent was thinking 2.7s

Yes, malware is known to disrupt computer performance. It can cause high CPU usage, leading to overheating and excessive fan noise. It can also slow down system performance, create security vulnerabilities, and lead to data loss or unauthorized access. Regular scans and up-to-date antivirus software help mitigate these risks.

1.00 | 2.7 sec |  

Remind me of the ticket number. 

The user asks a third question: **Remind me of the ticket number.**

Step 2

(2/2) Maximum conversation history: 1

Run an agent New chat ×

res, malware is known to disrupt computer performance. It can cause high CPU usage, leading to overheating and excessive fan noise. It can also slow down system performance, create security vulnerabilities, and lead to data loss or unauthorized access. Regular scans and up-to-date antivirus software help mitigate these risks.

1.00 | 2.7 sec |

Remind me of the ticket number.

✦ Agent was thinking 1s

Could you please provide more details about the issue or the specific ticket you're referring to? That way, I can assist you better.

1.00 | 1.0 sec |

Enter a message or task...

But this time, the agent responds: **Could you please provide more details or context about the ticket you're referring to?**

What happened? The agent has forgotten the earlier context about ticket 2024-091 and malware because those interactions have been pushed out of its accessible history.

Now it's time to test again.

Step 1

(1/3) Maximum conversation history: 10

> Model configuration

> Maximum conversation history

Define the maximum number of replies the agent can use as context in a conversation.

You **increase the limit** so the agent uses **10 replies as context in conversation history**.

Step 2

(2/3) Maximum conversation history: 10

Run an agent New chat ×

Give me details about ticket 2024-091.

✦ Agent was thinking 2.3s

Tool ✓ ▾
Agent Knowledge

✦ Agent was thinking 2.5s

Ticket 2024-091 details:

- **Issue:** Laptop fan running at maximum speed constantly, making the computer very loud even when idle with no applications running.
- **Resolution:** Checked Task Manager and discovered cryptocurrency miner malware. Removed the malware using an antivirus scan. Cleaned startup programs. CPU usage

Enter a message or task... ▶

User asks the **same questions** as in the previous conversation.

Step 3

(3/3) Maximum conversation history: 10

Run an agent New chat ×

discovered cryptocurrency miner malware. Removed the malware using an antivirus scan. Cleaned startup programs. CPU usage normalized, and the fan is now running quietly.

2.00 | 5.0 sec | 📄 ↻

Remind me of the ticket number.

✦ Agent was thinking 0.9s

The ticket number is IT-2024-091.

1.00 | 0.9 sec | 📄 ↻

Enter a message or task... ➤

This time, when the user asks **Remind me of the ticket number**, the agent responds: **The ticket number is 2024-091.**

The agent maintains context throughout the entire conversation because all interactions fit within the 10-reply limit.

Maximum conversation history controls what your agent remembers.

The step timeout setting controls how long your agent is allowed to reason.

Let's discover more about it.

Continue to 2.3.2: Step timeout

2.3.2 Step timeout

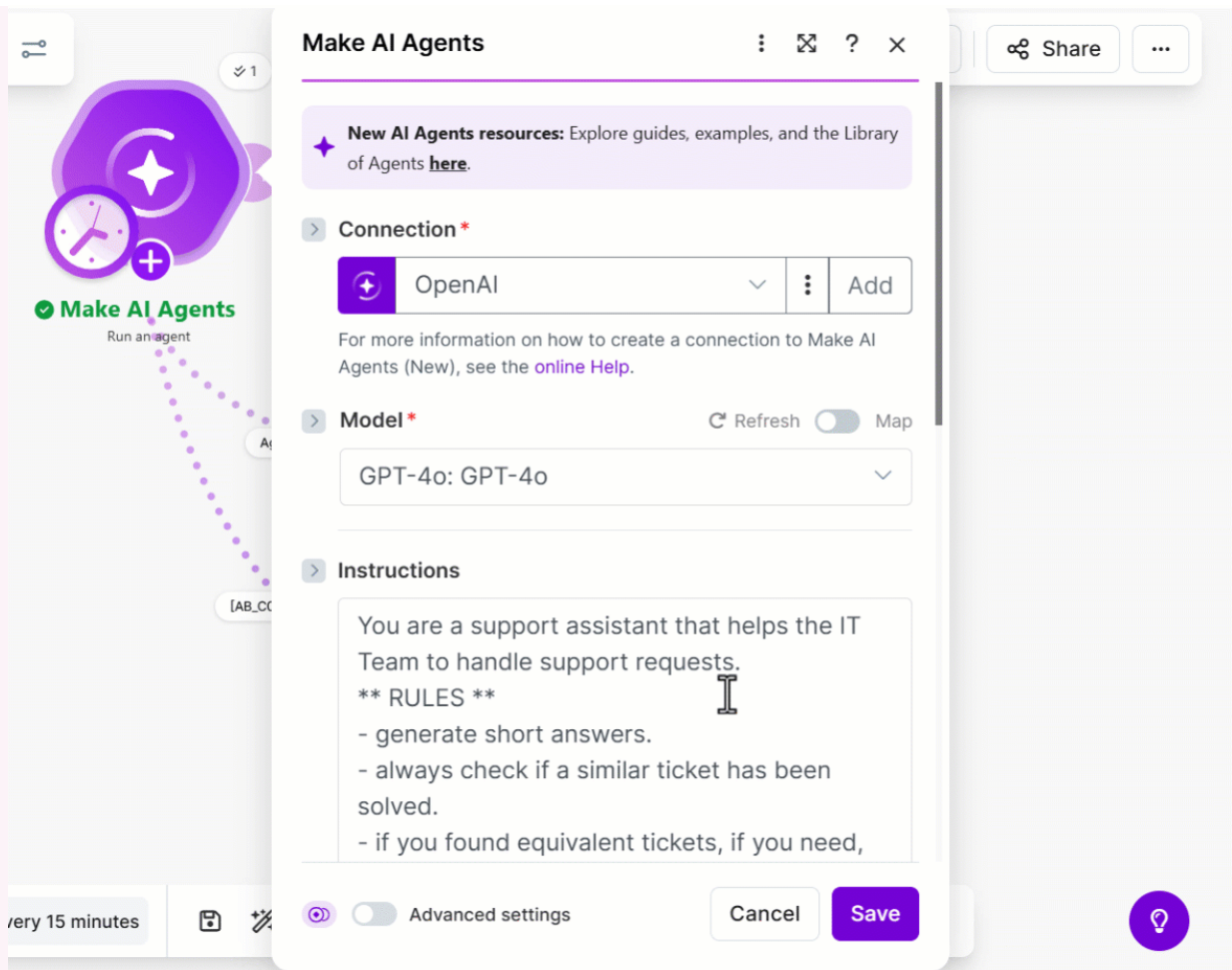


Step timeout sets the **maximum duration (in seconds)** that the LLM can spend **reasoning and processing**.

This is specifically the time the model takes to think through its current task, make decisions, and generate responses. **If the model exceeds this time limit**, Make terminates the agent execution and returns a timeout error.



Note that step timeout controls when Make stops waiting, not how fast your agent processes. It measures only the model's reasoning time, tool execution time is not included.



If you leave the **Step timeout** field empty, the **default timeout** is 300 seconds (5 minutes).

The maximum timeout you can give the agent is 600 seconds (10 minutes).

Consider adjusting this setting in the following cases:

Increase

- Your agent performs complex reasoning with multiple decision

Decrease

- Your agent handles quick, straightforward tasks

reasoning with multiple decision points or extensive analysis

- You see frequent timeout errors before your agent completes its tasks

straightforward tasks

- You need to give users a fast response

Practical examples on when to adjust this setting



Daily finance report agent

- **AI agent:** Analyzes trade data, reviews market news, and generates comprehensive reports with recommendations
- **Problem:** Frequent 300-second timeouts before the agent completes its reasoning
- **Solution:** Increase timeout to 500-600 seconds to give the model adequate time for complex analysis



Lead qualification agent

- **AI agent:** Quickly scores incoming leads based on simple criteria (company size, industry, budget)
- **Problem:** Simple classification task shouldn't take long, but agent occasionally gets stuck processing
- **Solution:** Set timeout to 120-150 seconds to fail fast if something goes wrong, preventing long waits



Content summarization agent

- **AI agent:** Reads long research papers and generates executive summaries with key findings
- **Problem:** Model needs time to process 10,000+ word documents and extract main points
- **Solution:** Increase timeout to 400-500 seconds to allow thorough analysis without rushing

Continue to 2.3.3: Response format

2.3.3 Response format

When your AI agent completes a task, it returns a response containing the results.

By default, agents return responses as **free-form text**, natural language that's easy for humans to read. This works well when the output goes directly to users, but when you need to use that data in automation, extracting specific values, routing based on categories, or mapping to database fields, you need a **structured format**.

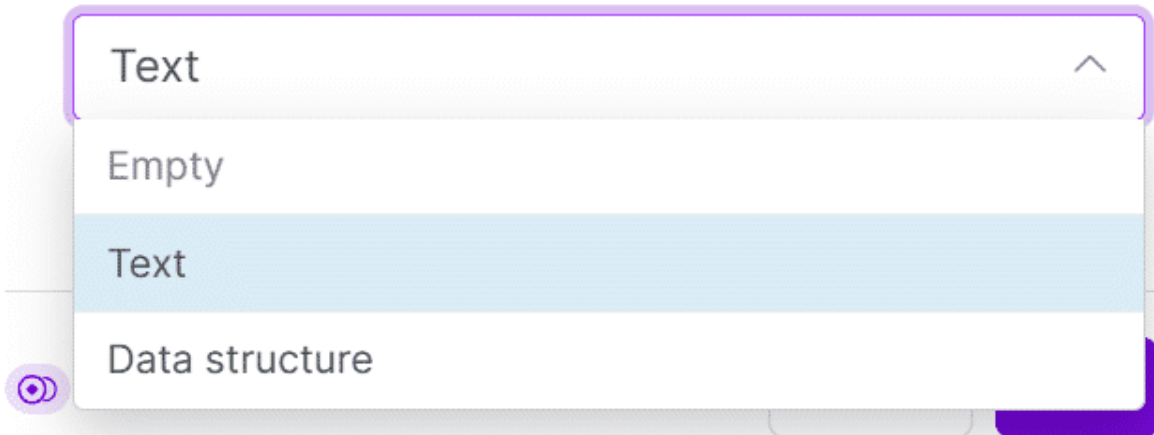


Response format defines the required structure for your agent's response. When you configure this setting, the agent must return data matching your specified structure. This ensures your agent will structure its output in a predictable format you can reliably use in subsequent Make modules.

You can choose the agent to return its answer in one of these formats:


- **Text:** The response is text.
- **Data Structure:** The response is based on the specific fields you define.


> Response format




Here's a few examples of when you can use each type:


Text


 Your agent writes content for people to read


 The output goes directly to users

 You don't need to extract or process specific data

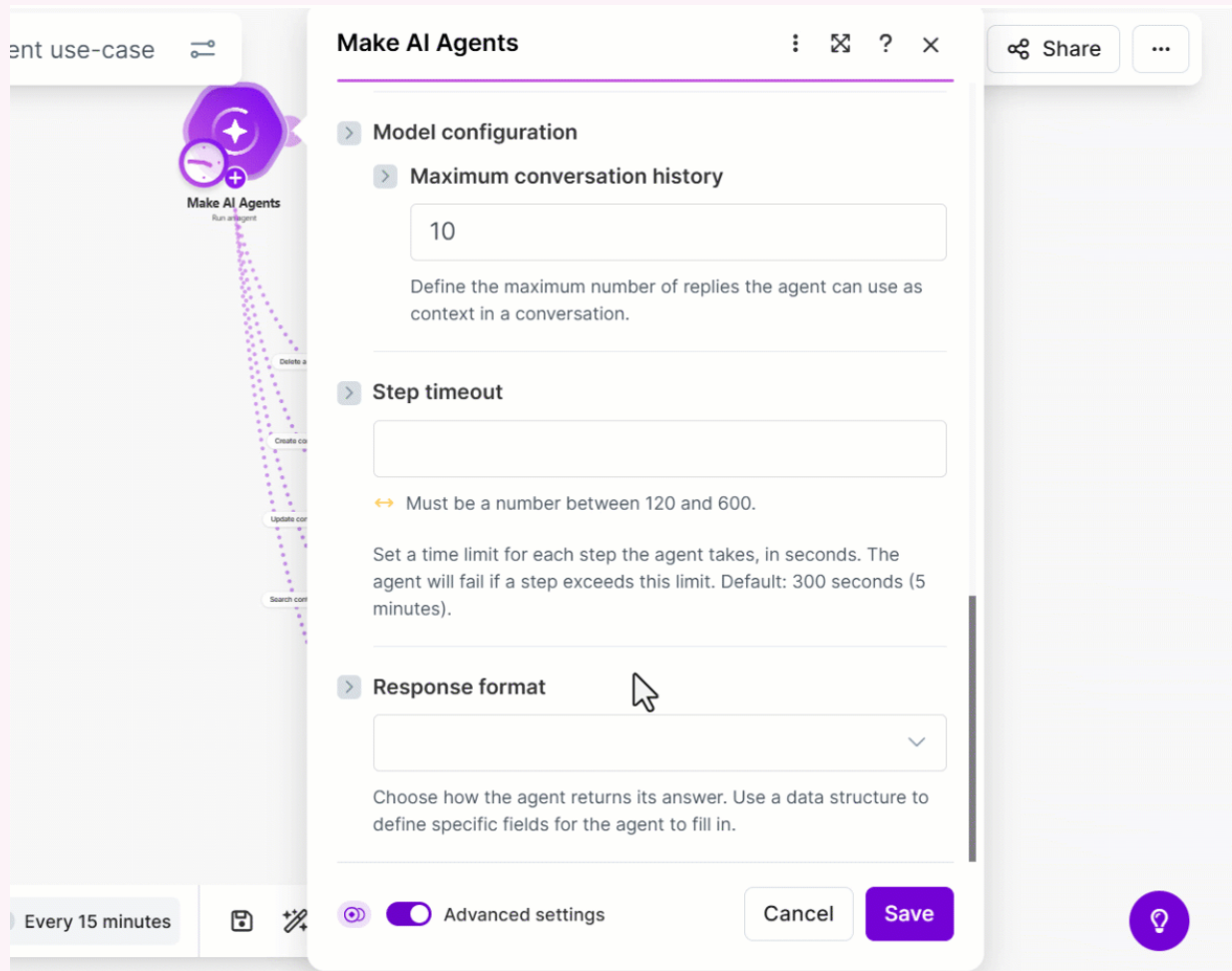
Data structure

 Your agent extracts specific data points

 You need to map and process data in subsequent Make modules

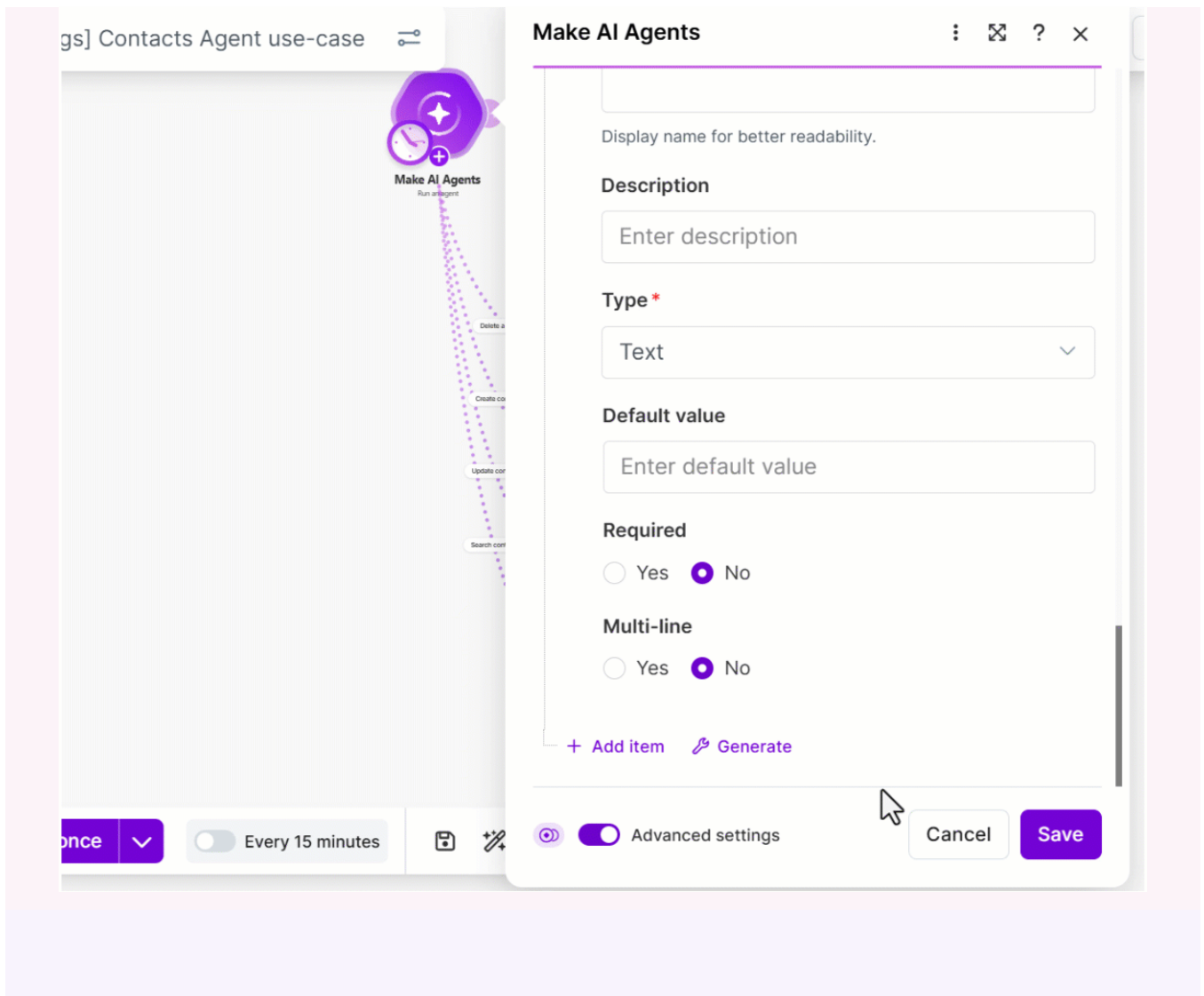
 You integrate with databases or external systems

Let's take a look at how to configure the Data Structure format in your AI agent.



If you select the option **Data Structure**, you can add your **Response structure** manually by clicking **+ add item**, or select **Generate**.

If you click **Generate**, Make generates a structure based on the **content type** and **sample data** you define, this will show the agent the structure you want. Available content types are **JSON**, **XML**, **Form Data**, and **Query String**.



Pro Tip

Including descriptions in each item of the **response format** helps the AI agent identify what data belongs in that field, ensuring your agent returns the right information in the right place when generating a response.

Let's see this in practice.

[Continue to see a practical example](#)

Practical example



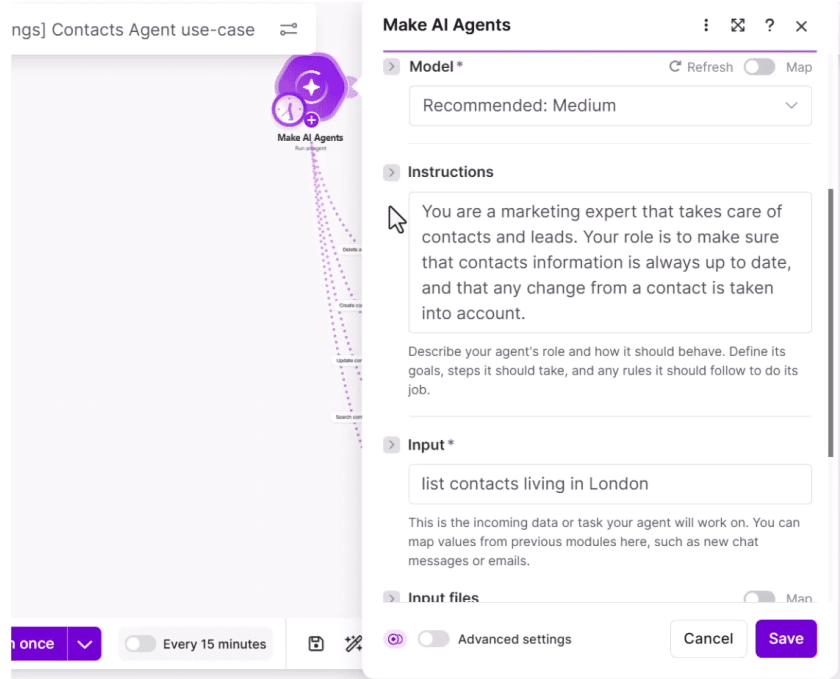
Your AI agent's task is to search contacts and return data in a format you can map and process in subsequent Make modules.

Here's how you can configure the **Response format** in your AI agent:

Work through each stage before you continue.

Step 1

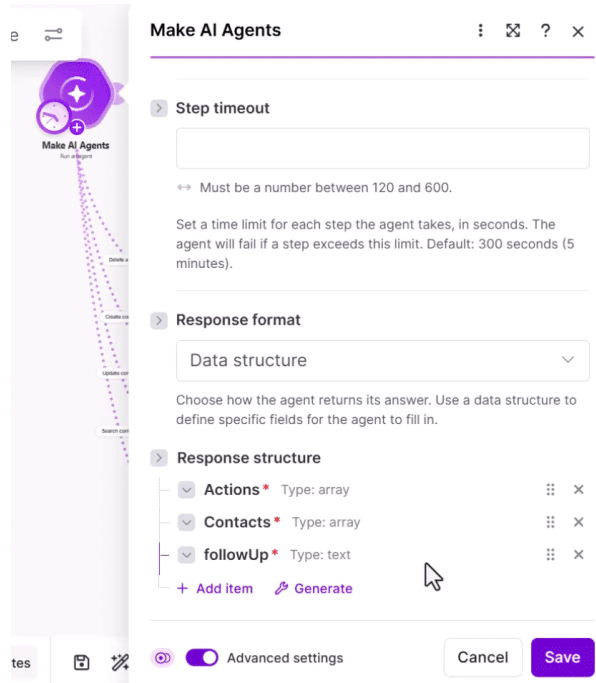
Configure your AI agent settings



After you click the **Advanced settings** tab in your Make AI agent, scroll down to **Response format** and select **Data Structure**. Click **Generate** and select **JSON** as your content type. This tells your agent to return structured **JSON** instead of plain text, making the data easy to map and process.

Step 1

Define your structure using sample data



Make AI Agents

Step timeout

↔ Must be a number between 120 and 600.

Set a time limit for each step the agent takes, in seconds. The agent will fail if a step exceeds this limit. Default: 300 seconds (5 minutes).

Response format


Data structure

Choose how the agent returns its answer. Use a data structure to define specific fields for the agent to fill in.




Response structure

- Actions* Type: array
- Contacts* Type: array
- followUp* Type: text

+ Add item Generate

tes  Advanced settings Cancel Save

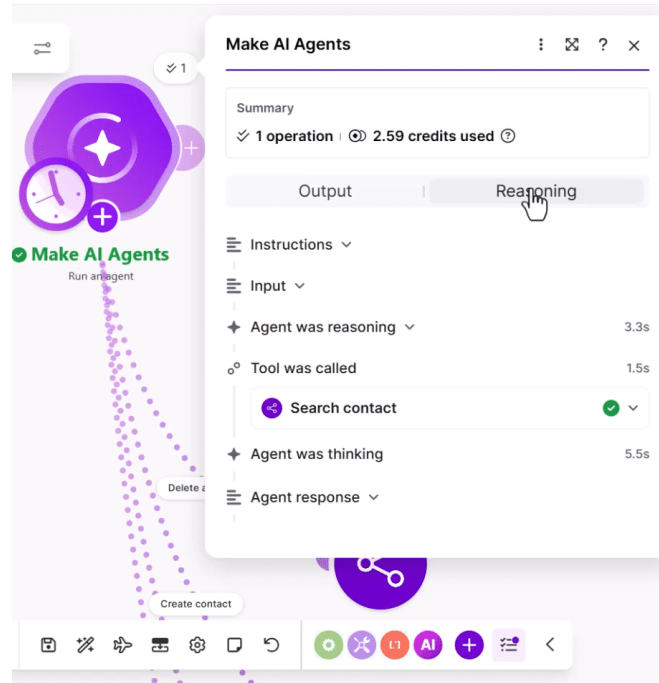
Provide **sample data** that shows the structure you want. This lets Make automatically develop a **Response structure** with the fields you determined:

-  **actions** (array): List of actions the agent performed during this task
-  **contacts** (array): Contact information with fields:
 - firstName
 - lastName
 - email
-  **followUpQuestion** (text): Any follow-up question the agent needs answered

The agent knows that it needs to return the output in the specific format you determined.

Step 1

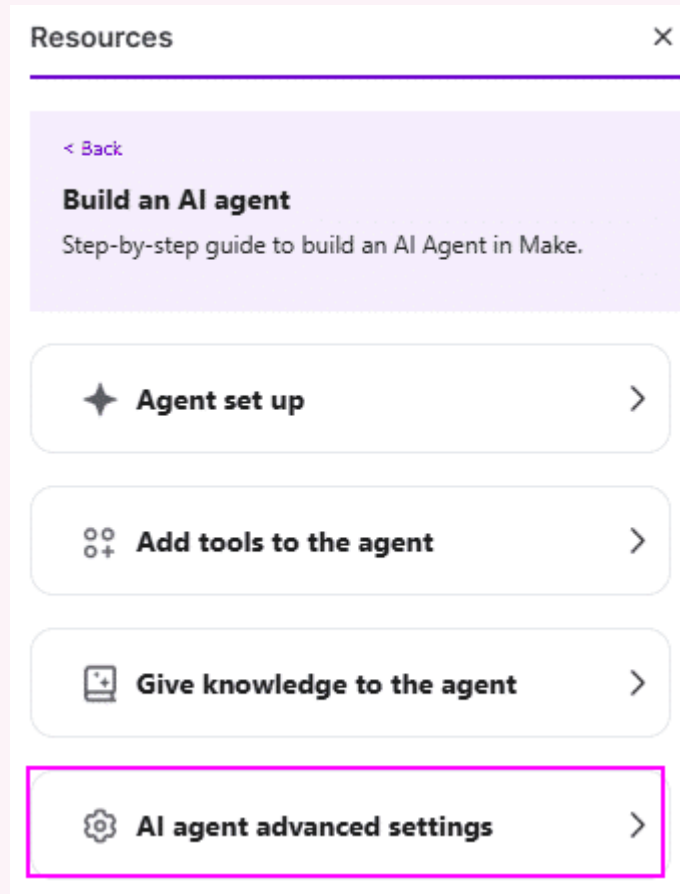
✓ What you get



After you run the agent, check the output bubble.

Your agent returns responses in the exact **JSON** format you defined. Instead of free-form text, you receive structured data you can immediately use in your Make scenario.

The **response format** determines whether your agent returns free-form text or structured data, it also allows you to specify the structure you want. You get predictable data you can immediately route, process, or send to other systems without additional parsing steps.



Want to set up these settings in your AI agent?

Go to the [Make website](#) and create a new scenario.

Click on the lightbulb in the bottom right to open the **Resources**. Click **AI agent advanced settings** and follow the instructions to learn how to apply these settings!

Configuring advanced settings in your AI agent ensures your agent performs reliably, without wasting resources, and according to your defined structure.

[Continue to the wrap up for this unit](#)



2.4 Wrap up

1

Model selection determines your AI agent's capability, cost, and speed. Start by testing the most capable model to see what's possible, then test smaller models to find the right balance. **The goal is to use the least powerful model that still completes the tasks accurately,** avoiding both overspending on unnecessary capability and choosing a model that can't handle the work.

2

Advanced settings allow you to fine-tune your AI agent's performance. **Maximum conversation history** limits how many past interactions your agent can use as context in a conversation. **Step timeout** sets how long

the model can spend reasoning and processing before Make stops the execution. **Response format** determines whether your agent returns free-form text or structured data, it also allows you to specify the structure you want.

3

These settings work together to optimize your agent's performance. Model selection gives your agent the right level of capability for the task. **Configure these settings based on your specific use case to balance quality, cost, and speed.**

Unit complete!

Congratulations! You've learned how to select the right model and configure your AI agent's advanced settings in Make.

By now you should have an understanding of:

- how to choose the LLM to improve your AI agent's performance
- what are the advanced settings of Make's AI agent
- how to adjust advanced settings



In the next course, you will plan and build a production ready agent in Make.

 make | academy



Mark this task complete to continue to the next unit.

