

THE BROKER TIMES

GUIDE 2 OF 5

Make setup

AI WhatsApp Agent Engine · thebrokertimes.com.au

Time: 30 minutes, assuming the data stores are already created.

Before you start: create the two data stores from `data-structures/data-stores.pdf`. Do it first — importing the blueprint before the stores exist means re-mapping every field by hand afterwards.

Step 1 — Import the blueprint

1. In Make, go to **Scenarios** → **Create a new scenario**
2. Click the `...` menu (bottom of the editor) → **Import Blueprint**
3. Choose `blueprints/01-whatsapp-ai-agent.blueprint.json`
4. Save

You'll see the scenario laid out with a router splitting into **four** branches: opt-out, first contact, name given, and the AI route. Several modules will show a warning triangle. That's expected — connections and data stores never travel inside a blueprint. The next steps clear every one of them.

Step 2 — The WhatsApp connection and webhook

Module 1 — WhatsApp Business Cloud → Watch Events

1. Open it. Under **Webhook**, click **Add**.
2. Name it "WhatsApp inbound".
3. Under **Connection**, click **Add** and fill in: - **App ID** — from your Meta app dashboard - **App Secret** — from Settings → Basic - **Access token** — the *permanent* System User token from Guide 1
4. Save. Make now shows you a **webhook URL** and a **verify token**.

Now register that URL with Meta:

5. Meta app dashboard → **WhatsApp** → **Configuration** → Webhook → **Edit**
6. Paste Make's URL into **Callback URL** and Make's token into **Verify token**
7. Click **Verify and save**. If it fails, the token has a stray space in it — that's almost always the cause.
8. Still on that page, under **Webhook fields**, click **Manage** and subscribe to **messages**. Nothing else is needed.

Modules 21, 34, 50 and 62 — Send a Message

There are four of these, one on each router branch. Open each one and pick the connection you just made from the dropdown. Nothing else to change on this pass.

MODULE	BRANCH	WHAT IT SENDS
21	Opt-out	The unsubscribe confirmation
50	First contact	The AI disclosure and the name question
62	Name given	The four-row loan-type list
34	AI route	Every reply the model writes

Miss one and that branch fails silently the first time a real client hits it, so check all four before you move on.

Step 3 — Point the data store modules at your stores

Ten modules need a data store selected. Six point at `wa_contacts`, four at `wa_messages`. Getting one the wrong way round is the most common setup mistake in this build, so work down the table and tick them off:

MODULE	STORE	WHAT IT DOES
3	<code>wa_contacts</code>	Looks up the contact, checks stage and opt-out status
20	<code>wa_contacts</code>	Marks them opted out
51	<code>wa_contacts</code>	Creates the contact on first message, stage <code>AWAITING_NAME</code>
60	<code>wa_contacts</code>	Saves their name, stage <code>AWAITING_TYPE</code>
37	<code>wa_contacts</code>	Updates stage and captured notes after each AI reply
28	<code>wa_messages</code>	Pulls the last 15 messages for context
52	<code>wa_messages</code>	Logs their opening message
61	<code>wa_messages</code>	Logs the message that contained their name
35	<code>wa_messages</code>	Logs what the client said
36	<code>wa_messages</code>	Logs what the agent replied

A quick way to check: **every module numbered 3, 20, 37, 51 or 60 is contacts. Every module numbered 28, 35, 36, 52 or 61 is messages.**

Open each, pick the store from the dropdown. The field mappings underneath should populate automatically because the field names match. If a field shows as blank, your data structure field name doesn't match exactly — fix the name in the data structure rather than re-mapping here.

Check module 3 and 28 both have "Continue the route when there are no results" switched on. It should come across from the blueprint, but verify it. Without it, the very first message from a new contact goes nowhere and you'll waste an hour wondering why.

Step 4 — The system prompt

Module 30 — Transform to JSON

This module exists purely so your prompt gets escaped correctly before it goes into the API request. Without it, a single apostrophe or line break in your prompt breaks the JSON and the scenario fails.

Do not remove the quote marks around `{{30.json}}` in module 32. Transform to JSON returns the text *escaped but without surrounding quotes*, so the request body supplies them. We tested this against a string containing a double quote, an apostrophe, an ampersand, a backslash and a line break — it parses correctly as written, and breaks if you take the quotes off.

1. Open it
2. Delete the placeholder text in the **Value** field
3. Paste your completed system prompt from `prompts/system-prompt.md` — with your business details filled in and your knowledge base pasted into the bottom section
4. Do not add quote marks around it

Module 31 is the same mechanism for the customer's message and history. Leave it alone.

Step 4b — Your onboarding messages

Two messages are sent by the scenario itself rather than by the model. That makes them instant, completely predictable, and free of AI cost — which matters, because one of them carries your AI disclosure.

Module 50 — the greeting and the name question

Plain text. It introduces the business, discloses that this is an automated assistant, points at your privacy policy, and asks their name.

Replace **two** placeholders here: `[YOUR BUSINESS NAME]` and `[PRIVACY POLICY LINK]`. Keep the disclosure sentence and keep the privacy line — between them they are what makes an AI on a licensed broker's number defensible. If you do not have a privacy policy page, get one before you go live; the compliance pack explains what it needs to cover.

Module 62 — the loan-type menu

Sent the moment they answer with their name. Reads "Thanks . What are you after?"

This is an **interactive list message**, not reply buttons. That is deliberate: WhatsApp caps reply buttons at **three**, and there are four loan areas. A list handles up to ten rows and gives each one

a description line, at the cost of one extra tap — the client taps *Choose one*, then picks from the sheet that opens.

Edit these:

- **Body text** — keep the `{{2.msg_text}}` token, that is their name.
- **Button** — the label that opens the list. Ships as *Choose one*. Max 20 characters.
- **Section title** — *How can we help?* Max 24 characters.
- **Rows** — the four loan areas, each with a title and a description.

Meta's hard limits: **row title 24 characters, description 72, up to 10 rows**. Count before you change them; an over-length value is rejected at send time, not at save time, so it fails silently in front of a client.

Delete any row you don't write. If you don't do personal loans, remove that row rather than letting the agent offer it and then hand over. Three rows or fewer is fine — the list still works.

If you end up with exactly three areas and want to save the client a tap, the blueprint can be switched back to reply buttons. Ask us and we'll send that version.

Whatever the buttons say, make sure your knowledge base covers those areas. A button the agent can't then talk about is worse than no button.

The tap comes back as an ordinary inbound message carrying the row title, which is why module 2 reads whichever of three things is present: typed text, a reply-button title, or a list-row title. You don't need to change that formula.

Step 4c — Photos, voice notes and stickers

WhatsApp clients send photos of trucks, PDFs of contracts and voice notes. The agent cannot see any of them.

The scenario handles this rather than ignoring it. Module 2's filter passes those messages through, and the message text the model receives becomes `[the customer sent an attachment you cannot see]`. The system prompt then has the agent say plainly that it can't open attachments in chat, ask for the key details as text, and offer a call.

That is deliberate. Silence is the one response guaranteed to make a client think your business ignored them.

Step 5 — OpenRouter key

Module 32 — HTTP → Make a request. Replace `YOUR_OPENROUTER_API_KEY` in the Authorization header. See Guide 3.

Step 6 — The hot lead notification

Module 38 fires when the model marks a conversation HOT, requests a handover, or records an opt-out. It's set to POST to a webhook URL you provide.

Opt-outs are in there deliberately. If somebody says STOP — and especially if they ask you to delete their information — a person needs to know, and the agent has already gone quiet on them. That notification is your only signal.

Easiest options:

- **Another Make scenario** with a Custom Webhook trigger, which then emails or SMSs you. Two modules.
- **Your CRM's inbound webhook**, if it has one.
- **Slack or Teams** incoming webhook.

Paste the URL into the **URL** field. If you don't want notifications yet, right click module 38 and **Disable** it — don't leave a placeholder URL in place, it will throw errors on every hot lead.

What the receiving end gets. Module 38 posts as a normal HTML form (`application/x-www-form-urlencoded`), not JSON, because form fields can't be broken by an apostrophe in a client's message. The fields are `stage`, `handover`, `wa_id`, `name`, `capture`, `last_message` and `agent_reply`. If you're catching it with a Make Custom Webhook, they arrive as separate mappable items with no parsing needed.

Step 7 — Scenario settings

Click the ⚙ at the bottom of the editor:

- **Sequential processing: ON.** This matters. Two messages from the same person arriving seconds apart will otherwise be processed in parallel, and both will miss each other's context. Sequential makes them queue.
 - **Allow storing of Incomplete Executions: ON.** When something fails you want the bundle kept so you can look at it.
 - **Number of consecutive errors: 3**
-

Step 8 — Turn it on

Toggle the scenario **ON** (bottom left). Because the trigger is instant, it now sits waiting for Meta to call the webhook.

Go to Guide 4 and test it before it meets a real client.

What running costs look like

Roughly eight to ten Make operations per inbound message, depending on which route runs. A hundred-message month is around a thousand operations. Watch your own counter for the first week rather than trusting that estimate — your conversation lengths will differ.

If ops become a problem, the cheapest saving is dropping module 28's limit from 15 to 10.