

Guide for creating AI Lead Qualification Agent in Azure AI Studio

Josh Arrington

Chief Marketing Technology Officer, Kapturall

Adobe Marketo Engage Champion

Adobe

Table of Contents

Introduction	
Prerequisites	3
Create a Resource Group Create the Azure AI Studio workspace	3-4
Give your Agent Instructions	4-7
What to Review	8-9
Decision Process	
Load Knowledge into a Vector Store	10
Setting up Marketo Engage	11-13
Create Programs & Tokens	
Create Watch List	
Create Smart Campaign	
- Campaign Smart List	
- Campaign Flow	
- Campaign Schedule	
Giving our AI Agent Tools	14-20
enrichLeadData	
getLeadActivity	
requestMarketoCampaign	
Register the Logic App as a Tool in the Agent	21
Setting up a trigger for our AI Agent	22-23
Test the Agent (manual and evaluation runs)	24
Creating Additional Logic Apps	25-30
Ongoing improvement loop	31
Quick Checklist	32

Introduction

In today's competitive landscape, speed and accuracy in lead qualification are critical. Manually sifting through inbound leads is time-consuming and prone to inconsistency. This guide provides a comprehensive, step-by-step walkthrough for building an intelligent AI agent that automates this process, ensuring high-quality leads are identified and routed to sales faster than ever.

By leveraging the power of Microsoft Azure AI Studio and integrating it with Marketo Engage, you will create a sophisticated agent capable of analyzing lead data, enriching it with external information, evaluating it against your Ideal Customer Profile (ICP), and taking direct action within your marketing automation platform. This powerful combination allows you to build a scalable, consistent, and highly efficient lead qualification engine tailored to your business needs.

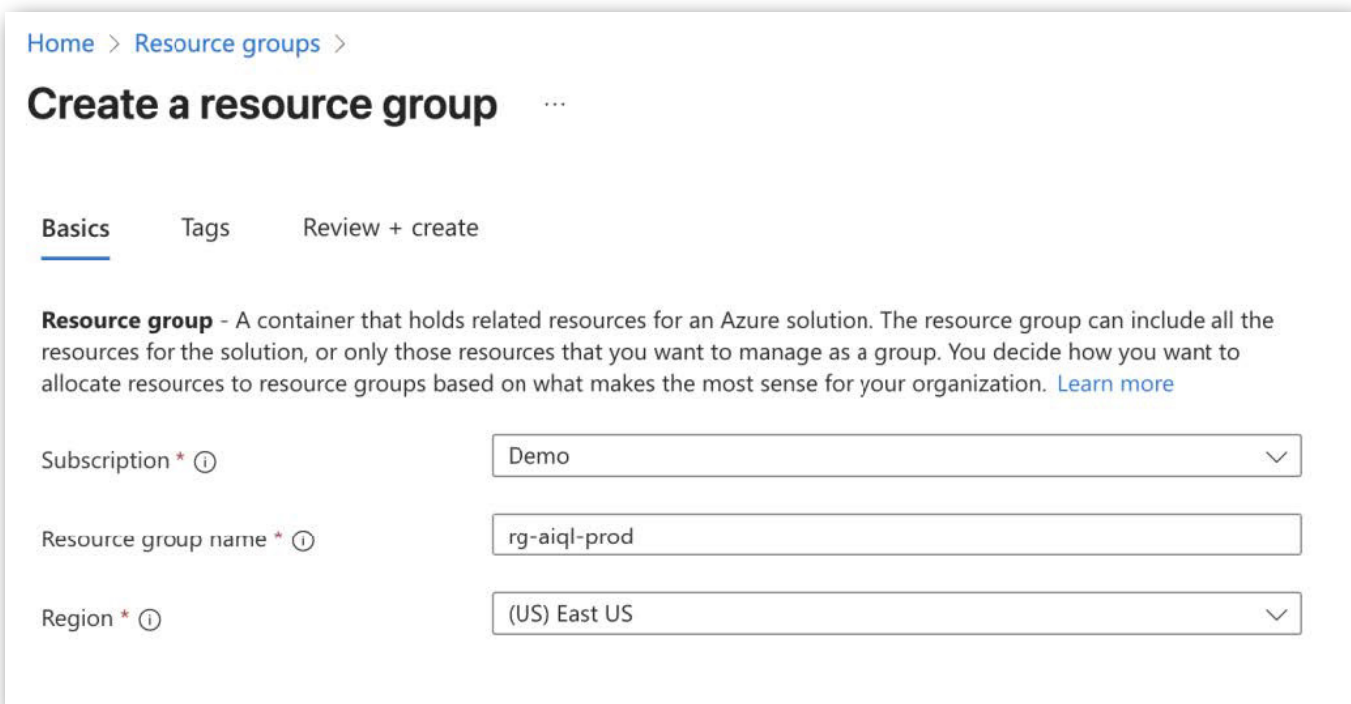
Prerequisites and Create a Resource Group

Prerequisites:

- Azure Account – If you don't have an Azure account, you can create one for free [here](#)
- Marketo API Credentials: Munchkin ID, REST URL, Client ID / Client Secret

Create a Resource Group:

1. [Azure Portal](#) → [Resource groups](#) → Create.
2. Name: rg-aiql-prod (or your naming standard), choose subscription + region → Review + Create.



Home > Resource groups >

Create a resource group

Basics Tags Review + create

Resource group - A container that holds related resources for an Azure solution. The resource group can include all the resources for the solution, or only those resources that you want to manage as a group. You decide how you want to allocate resources to resource groups based on what makes the most sense for your organization. [Learn more](#)

Subscription * ⓘ Demo

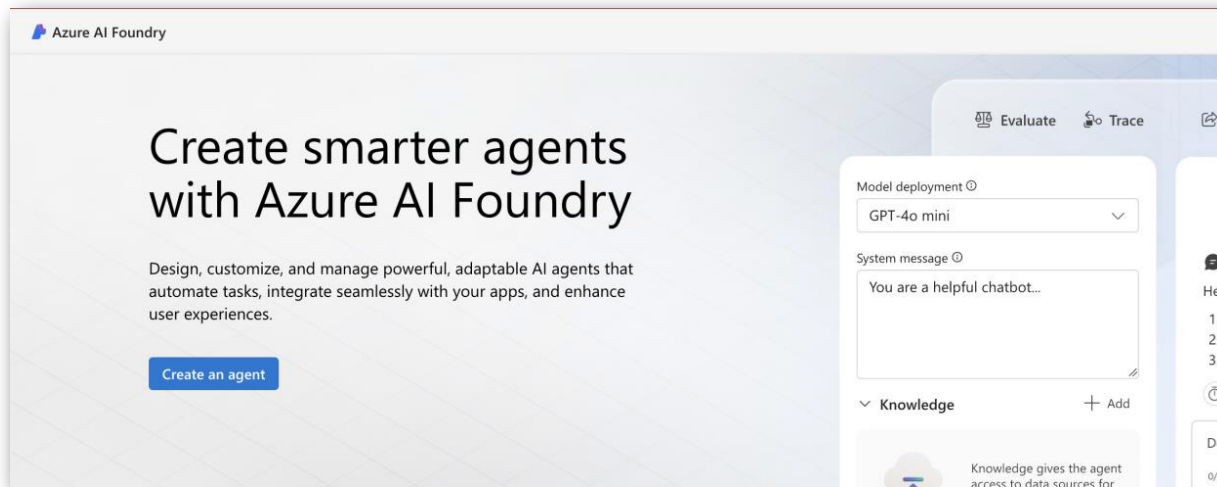
Resource group name * ⓘ rg-aiql-prod

Region * ⓘ (US) East US

Create the Azure AI Studio workspace

Azure AI Studio uses a **Hub** → **Project** structure and attaches dependent services.

1. Go to Azure AI Studio (formerly “Foundry”).



2.. Click the **Create an agent** button.

This will create an AI Foundry Hub, Project and Agent. Give both the project and AI Foundry Resource a name. Use the resource group you created before and select the same region.

Create a new project

To try out the generally available Agent experience and access the latest AI Foundry capabilities, you must create a project.

Project *

Advanced options
We'll set up a new project for you with defaults selected for optimal functionality
Your project will be located in the eastus2 region.

Project
(new) proj-aiql

Subscription *

Demo

Create a new subscription

Resource group *

rg-aiql-prod

Create new resource group

Azure AI Foundry resource *

ai-hub-marketo

Public network access
Enabled

Region *

East US 2

Got data, privacy or security policies to follow? [Configure in Azure Portal](#)

Create the Azure AI Studio workspace

3. Select a model

Choose a model to create a new deployment. For flows and other resources, create a deployment from their respective list. [Go to model catalog.](#)

Models 7 Inference tasks: Chat completion Show description

Search

gpt-4.1
Chat completion

gpt-4.1-mini
Chat completion

gpt-4.1-nano
Chat completion

gpt-4o
Chat completion

gpt-4o-mini
Chat completion

gpt-4
Chat completion

gpt-35-turbo
Chat completion

Prev

Next

gpt-4.1

Task: Chat completion

The gpt-4.1 series is the latest iteration of the `gpt-4o` model family. This iteration of models is specifically targeted for better coding and instruction following, making it better at handling complex technical and coding problems.

In addition, it increases the context token limit up to 1M input tokens and provides separate billing for small context (128k) and large (up to 1M) context inputs.

As with the previous gpt-4o model family, it supports a 16k output size and features such as:

- Text, image processing
- JSON Mode
- parallel function calling
- Enhanced accuracy and responsiveness
- Parity with English text and coding tasks compared to GPT-4 Turbo with Vision
- Superior performance in non-English languages and in vision tasks
- Support for enhancements
- Support for complex structured outputs.

4. Deploy your agent

Deployment name * 👁

gpt-4.1

Deployment type

Global Standard ▼

Global Standard: Pay per API call with the highest rate limits. [Learn more about Global deployment types](#).

Data might be processed globally, outside of the resource's Azure geography, but data storage remains in the AI resource's Azure geography. [Learn more about data residency](#).

Deployment details

Model version

2025-04-14

Capacity

50K tokens per minute (TPM)

Content safety

DefaultV2

Connected AI resource

ai-hub-marketo

Authentication type

Key

Resource location

East US 2

Version upgrade policy

Once a new default version is available

Customize

Deploy

Cancel

Tip: Keeping everything in the **same region & subscription** avoids cross-boundary headaches.

Create the Azure AI Studio workspace

5. Give the agent a name in the right panel

Setup Hide

Agent ID ⓘ

asst_CLbfTR8CM8SUVIC1iiDwyJ6c

Agent name

AI Lead Qualification Agent

Deployment * + Create new deployment ▼

gpt-4.1 (version:2025-04-14)

Take note of your Agent ID here. We will use it later to trigger the agent.

Give your Agent Instructions

Give your agent clear directions on what to do and how to do it. Include specific tasks, their order, and any special instructions like tone or engagement style.

Your job is to analyze inbound leads for our corporate real estate company and decide the most appropriate next step based on available data.

What To Review

- Lead data (behavioral, engagement, activity logs).
- Company data (firmographics such as size, industry, geography).
- Ideal Customer Profile (ICP) — check the vector store to evaluate how closely the lead and their company match our ICP.

After reviewing, you will call the Logic App by sending a JSON payload with the following fields:

- **leadID** – The Marketo ID of the lead.
- **aiAction** – The action you recommend for the lead. This must exactly match one of these values:
 - **qualify** - The lead is a good fit and ready for sales follow-up.
 - **disqualify** - The lead is not a fit and should be removed from active pursuit.
 - **nurture** - The lead is promising but not ready for direct sales contact; should enter a nurture program.
 - **hr** - The lead is an internal HR inquiry and should be routed to the HR team.
- **aiExplanation** – A concise summary of why you chose this action. Reference key factors like ICP match, company profile, lead behavior (e.g., form fills, email engagement), and intent signals.
- **aiCategory** – A short label for the lead's classification (e.g., "Strong ICP Fit," "Low Engagement," "Competitor," "HR Inquiry").
- **leadScore** – A numeric score (0–100) reflecting how likely the lead is to convert based on behavior and engagement.
- **icpScore** – A numeric score (0–100) reflecting how closely the lead matches our corporate real estate ICP (from the vector store).

Give your Agent Instructions

Decision process

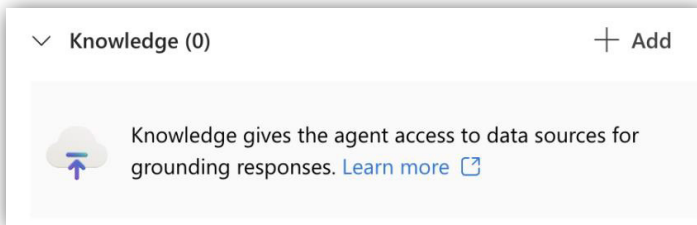
1. Always check the ICP fit in the vector store first to ensure alignment with the Ideal Customer Profile.
2. Weigh recent lead activity and engagement.
3. Use company firmographics to validate alignment with corporate real estate opportunities.
4. Choose the most appropriate aiAction.
5. Fill in all fields with clear, defensible reasoning.

If you are uncertain, you may call the **getApproval** tool before finalizing.

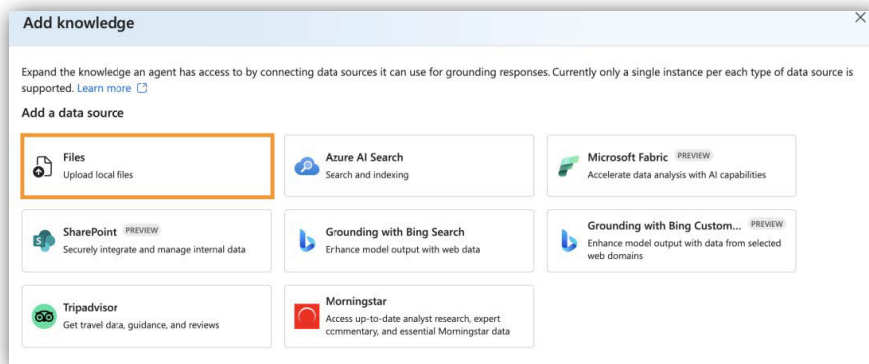
Once a decision has been made, call **requestMarketoCampaign** to trigger the appropriate action.

Load Knowledge into a Vector Store

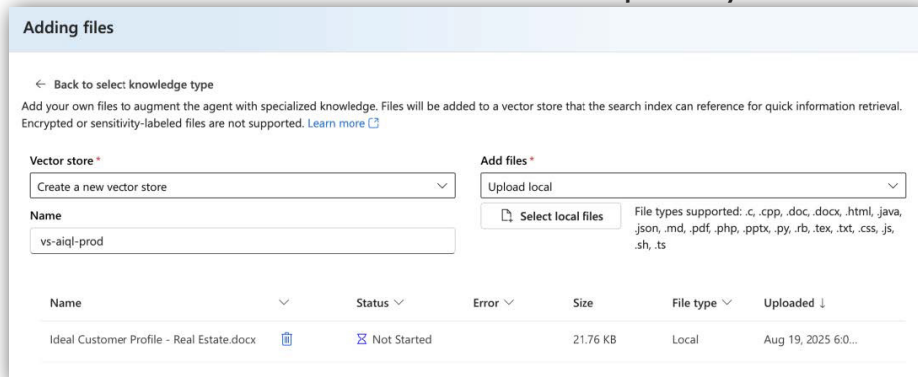
1. On the right panel find the **Knowledge** section and click **+ Add**



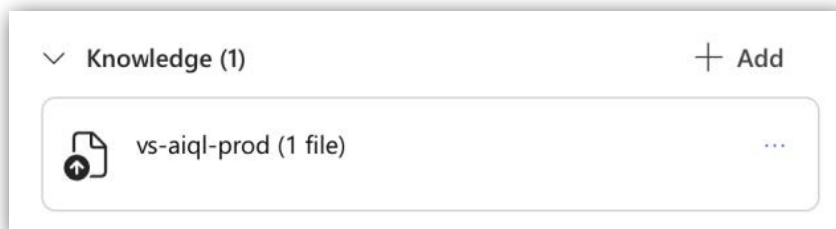
2. Select Files



3. Select or create a vector store and upload your ICP Document



4. Click the **Upload and save** button. You should now see the vector store under the knowledge section



Why vector store? It lets the agent find the **most relevant passages semantically**, not just exact keywords.

Setting up in Marketo Engage

Once our agent reviews and evaluates the leads we've asked it to pass back one of four actions (qualify, disqualify, nurture, hr) and some additional details that we will pass to Marketo Engage when triggering our smart campaign. We need to create a Smart Campaign for each of these actions

Create Programs and Tokens

Create Program Tokens for each value the agent is expected to send. The value we set at this moment isn't important, because when our agent calls the Smart Campaign it will pass in dynamic values for each lead.

Ty...	Token Name ▲	Value
Local (6 Tokens)		
	{{my.aiAction}}	none
	{{my.aiCategory}}	none
	{{my.aiExplanation}}	Double-Click for Details
	{{my.icpScore}}	0

Create Watch List

Create a static list. We will pull leads from this list for our agent to process

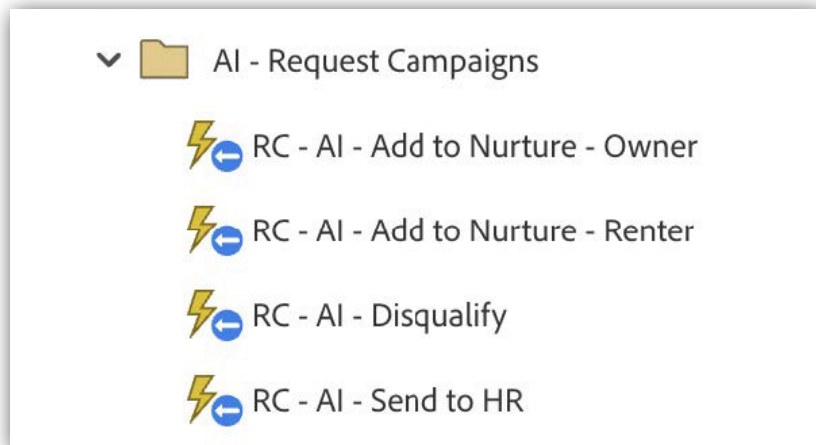
LIST - Leads to review

You can find the list ID in the URL. Navigate to your static list

<https://experience.adobe.com/#/@accountName/so:123-ABC-456/marketo-engage/classic/ST1020A1LA1>

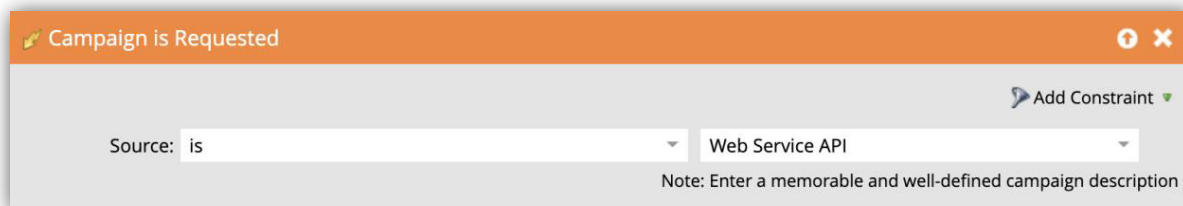
Setting up in Marketo Engage

Create Smart Campaign



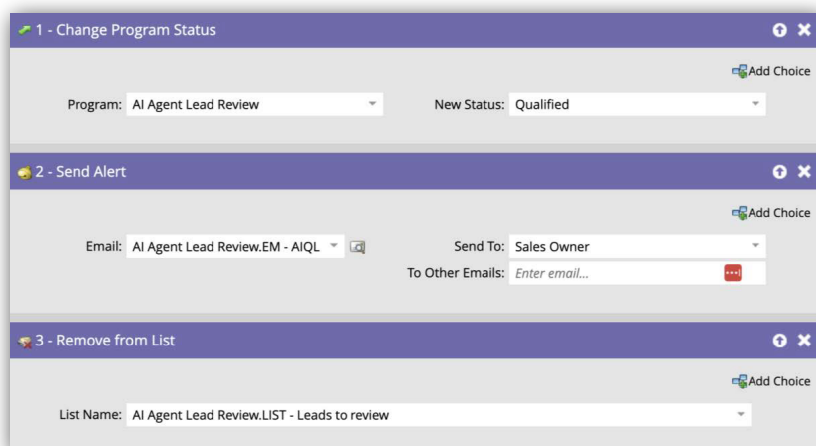
Campaign Smart List

The flow for our smart campaign will all use a single trigger which should be **Campaign is Requested** with source of **Web Service API** which will make our smart campaigns triggerable via the API.



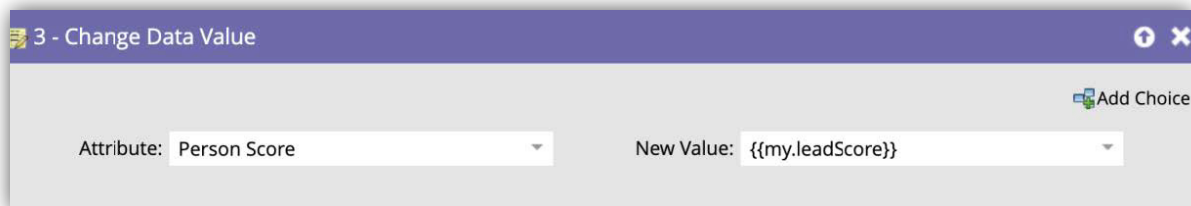
Campaign Flow

The flow for each campaign should match the action passed, but it can be whatever we want it to be. However, for our flow it is important to remove the user from the watchlist so it isn't processed again.



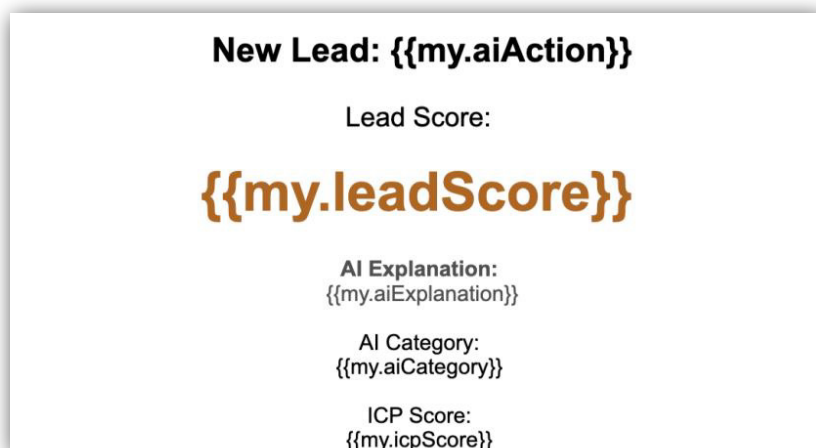
Setting up in Marketo Engage

We can use all of the program tokens to **change data value**



The screenshot shows a configuration window titled "3 - Change Data Value". It has a purple header bar with a close button. Below the header, there is a section with two dropdown menus. The first dropdown is labeled "Attribute:" and has "Person Score" selected. The second dropdown is labeled "New Value:" and has "{{my.leadScore}}" selected. There is an "Add Choice" button with a plus icon to the right of the "New Value:" dropdown.

We can also use them in the body of our emails. For a **sales alert**.

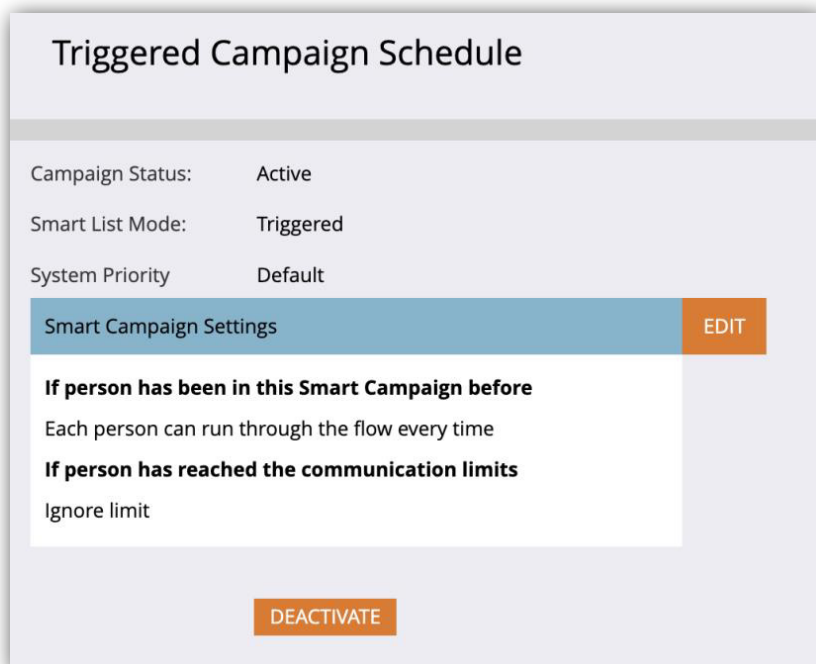


The screenshot shows an email template body with the following content:

- New Lead: {{my.aiAction}}**
- Lead Score:
- {{my.leadScore}}**
- AI Explanation:
{{my.aiExplanation}}
- AI Category:
{{my.aiCategory}}
- ICP Score:
{{my.icpScore}}

Campaign Schedule

Activate the campaigns so they are now available



The screenshot shows a configuration window titled "Triggered Campaign Schedule". It has a light purple header bar. Below the header, there is a section with three rows of settings:

- Campaign Status: Active
- Smart List Mode: Triggered
- System Priority: Default

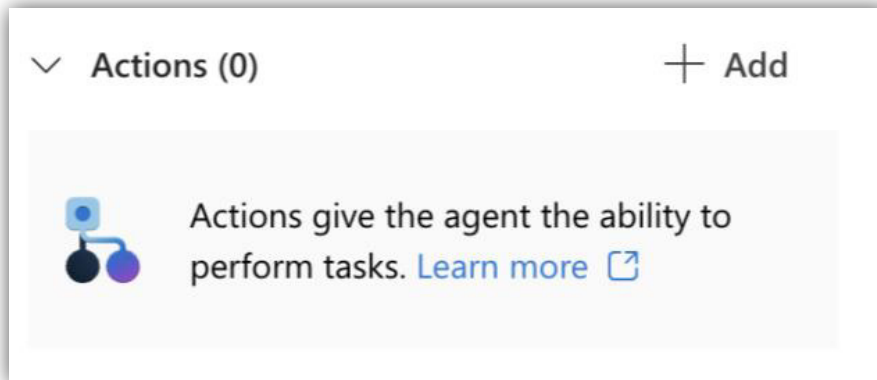
Below these settings, there is a blue bar labeled "Smart Campaign Settings" and an orange "EDIT" button. Below the blue bar, there is a white box with the following text:

- If person has been in this Smart Campaign before**
- Each person can run through the flow every time
- If person has reached the communication limits**
- Ignore limit

At the bottom of the window, there is an orange "DEACTIVATE" button.

Giving our AI Agent tools

In the right panel you will see an area for “Actions” where we can add tools for our agent to use, but first we need to create this functionality. In our case we will use Logic Apps to build our agents tools. These are no-code options for building data flows.



We will create three key actions:

- enrichLeadData
- getLeadActivity
- requestMarketoCampaign

Giving our AI Agent tools

enrichLeadData

Deploy to Azure

For lead enrichment, I'm using Reverse Contact via a simple HTTP API call to get a more complete picture of the lead and their company. You can use any third-party or internal resource for this purpose, but be sure to follow applicable regulations like GDPR and your company's policies. Some other great options are ZoomInfo, Lusha, and FullContact

Place the logic app in the same resource group and region that we created before. Here, you'll need to enter your reverse contact key. If you want to use a different service, you can still use this template. Just enter any value here and then remove this action from the Logic App flow. Later in this guide there is a guide to adding additional Logic Apps actions to our agent.

The screenshot shows the Azure Logic App configuration interface for a Logic App named 'enrichLeadData' in the 'rg-aiql-prod' resource group, (US) East US region. The workflow consists of three actions: 'Request from AI Agent', 'LinkedIn Lookup with Reverse Contact', and 'Response'. The 'Request from AI Agent' action is linked to the 'LinkedIn Lookup with Reverse Contact' action, which is linked to the 'Response' action. The 'Request from AI Agent' action has a 'Request Body JSON Schema' defined as:

```
{
  "type": "object",
  "properties": {
    "email": {
      "description": "Email address to use for lookup",
      "type": "string"
    }
  }
}
```

The 'LinkedIn Lookup with Reverse Contact' action is configured with the following details:

- URI: `https://api.reversecontact.com/enrichment`
- Method: GET
- Headers: (Empty)
- Queries: `apikey` (value: `your_API_key`), `email` (value: `email`)
- Body: (Empty)

The 'Response' action is configured with the following details:

- Body: (Empty)

Giving our AI Agent tools

getLeadActivity

This tool allows the agent to pull the lead's activity history to analyze as part of the holistic review. Unfortunately, there is currently no built in Marketo Engage connector for activities in Azure Logic Apps, and this one gets a little complicated. To help with this I've created a template you can deploy from this link:

Deploy to Azure

Place the logic app in the same resource group and region that we created before. Here you'll need to enter your Marketo REST API credentials and select the number of number of days of activity you want the agent to pull. Click **Review + create** and deploy.

Basics Review + create

Template
 Customized template [↗](#)
1 resource
[Edit template](#) [Edit parameters](#) [Visualize](#)

Project details
Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.
Subscription * ⓘ Demo
Resource group * ⓘ rg-aiql-prod
[Create new](#)

Instance details
Region * ⓘ (US) East US ✓
Logic App Name getLeadActivity ✓
Marketo Munchkin Id *
Marketo Client Id *
Marketo Client Secret *
Days_of_activity 30 ✓

Once created you will see the logic app in your Azure resource group.

Giving our AI Agent tools

requestMarketoCampaign

Deploy to Azure

Basics

Review + create

Template

Customized template 

2 resources

 Edit template

 Edit parameters

 Visualize

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * 

Demo 

Resource group * 

rg-aiql-prod 

[Create new](#)

Instance details

Region * 

(US) East US 

Logic App Name 

requestMarketoCampaign 

Marketo_connection_name 

marketoma 

Location 

[resourceGroup().location]

This template will create both the logic app and an API Connection: **marketoma**. First we need to set up the Marketo API connection. Click on it.

Resources

Recommendations

Filter for any field...

Type equals all 

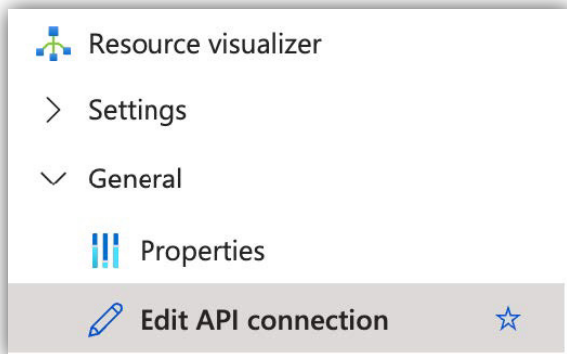
Location equals all 

 Add filter

Showing 1 to 6 of 6 records. ☐ Show hidden types 

Giving our AI Agent tools

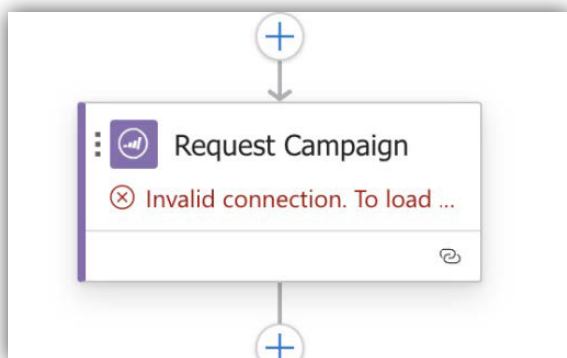
Next, in the left side menu select **Resource visualizer > General > Edit API connection**



Here fill in your Marketo REST API credentials and click **Save** To help generating the credentials, check out the [authentication documentation here](#).

A screenshot of a web form titled 'Edit API connection' with a pencil icon. Below the title is a descriptive sentence: 'Edit API connection lets you update the display name and refresh the authorization for this SaaS provider.' The form contains several input fields: 'API' (a dropdown menu showing 'Marketo MA'), 'Display Name' (a text box with 'marketoma'), 'MunchkinID *' (a text box with a help icon), 'Client ID *' (a text box with a help icon), and 'Client Secret *' (a text box with a help icon). Each field has a red asterisk indicating it is required.

If you don't set this up, you will see an error in the Request Campaign step in the Logic App:

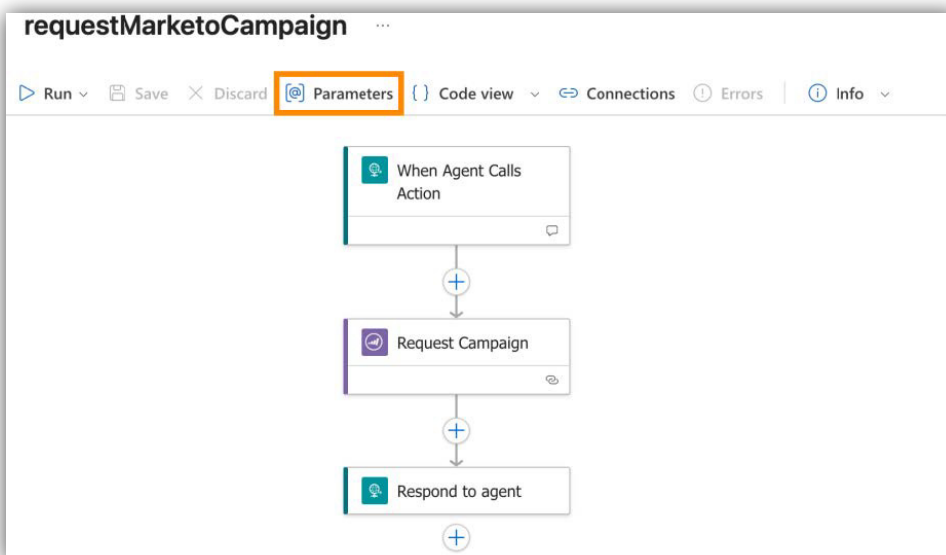


Giving our AI Agent tools

Next go back to the logic app and Edit



Edit the parameters



Giving our AI Agent tools

Here we can map the aiActions we've asked the agent to generate, to specific smart campaigns in Marketo. Update these to match your smart campaign IDs and add or change the AI actions you want. Also add a default smart campaign that will be called as a fallback when the AI Action isn't mapped to a smart campaign

You can find the Smart Campaign ID in the Marketo URL. Navigate to your Smart Campaign

<https://experience.adobe.com/#/@accountName/so:123-ABC-456/marketo-engage/classic/SC1236A1ZN38>

▼ campaignMapping

Object

Name *

campaignMapping

Type *

Object

Default value *

```
{"qualify":1239,"nurture":1237,"disqualify":1236,"hr":1240}
```

Actual value

▼ defaultCampaignId

Integer

Name *

defaultCampaignId

Type *

Integer

Default value *

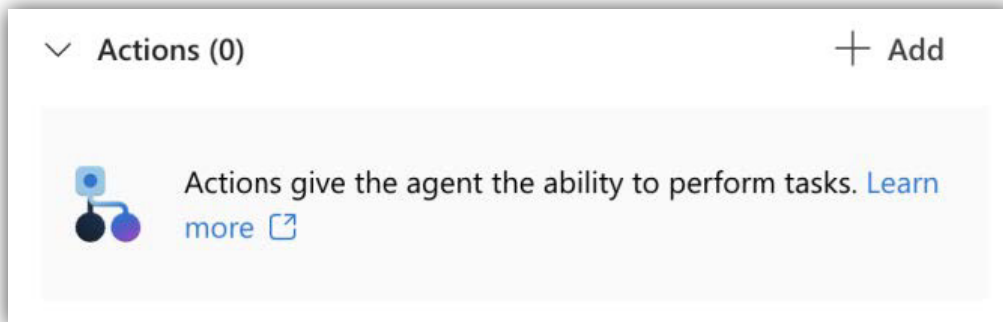
1234

Actual value

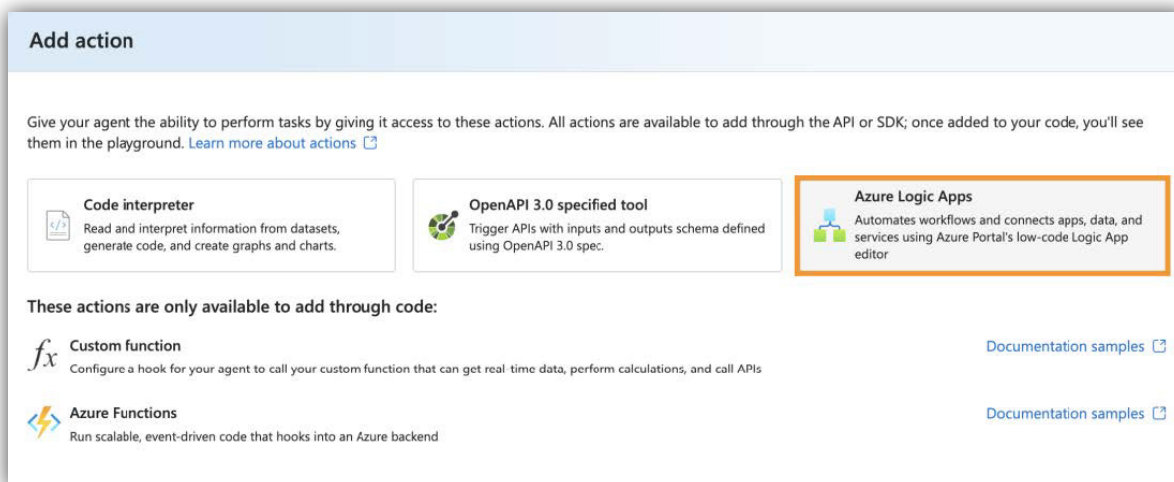
Register the Logic App as a Tool in the Agent

Now that we've created our actions we need to tell our AI Agent about them.

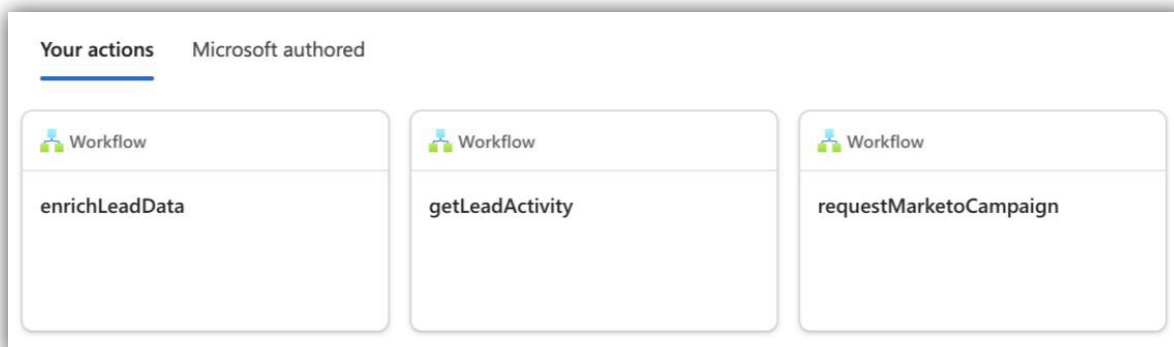
Back in AI Foundry, find the Actions section and click +Add



Select Azure Logic Apps



You should now see the Logic Apps created before. Select one and click **Create** then repeat this process for each action.



Setting up a trigger for our AI Agent

Now that our agent is setup with knowledge and tools, we need a way to trigger it. Specifically, we need to give it leads to review. Here again we will use a Logic App. I will set a schedule to check out Marketo Engage list every 10 minutes and process the leads inside. Here you have a template you can deploy to Azure.

Deploy to Azure

Basics Review + create

Template
 Customized template 
1 resource
[Edit template](#) [Edit parameters](#) [Visualize](#)

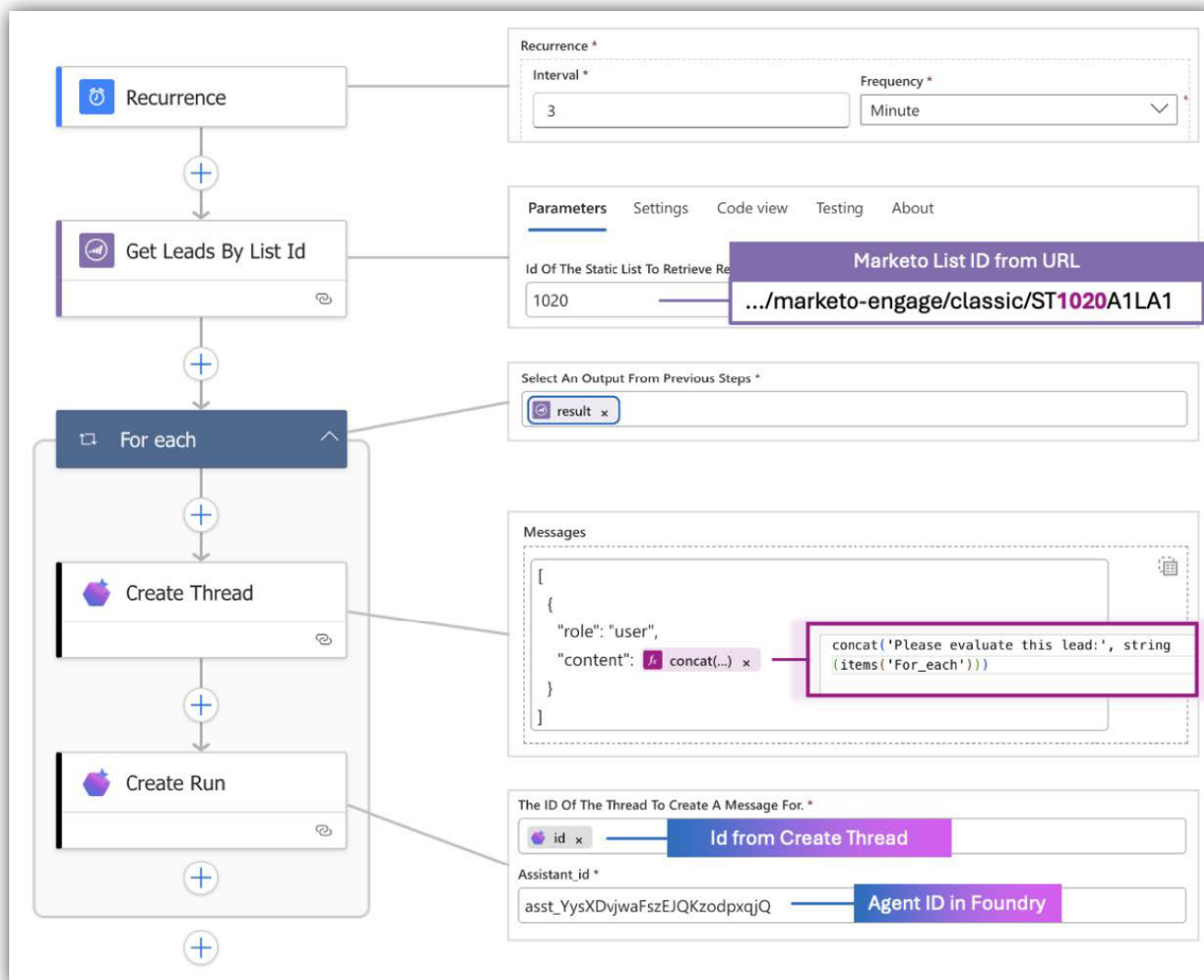
Project details
Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.
Subscription * ⓘ
Resource group * ⓘ
[Create new](#)

Instance details
Region * ⓘ ✓
Logic App Name ✓
Marketo_connection_name ⓘ ✓
Ai-agentservice_connection_name ⓘ ✓
Ai_foundry_uami_externalid ✓
Mkto List Id * ⓘ ✓
Ai-agent-id * ⓘ ✓

Setting up a trigger for our AI Agent

We will use the Marketo static list we created.

Timer → Agent: A Logic App **Recurrence** trigger (e.g., every 10 minutes) pulls leads from our static list and calls the Agent for each record.



Once configured the app will begin checking our static list for leads on the schedule we've set and send them along to our agent to process.

Test (manual and evaluation runs)

1. In **Agents** → **Test**, paste a sample lead JSON (company/contact + recent activity).
2. Watch tool calling:
3. If data is missing, it should call **Enrich**.
4. If strong fit/intent, it should call **Request Campaign** and include token values.
5. Validate the **Response** from the Logic App and confirm the **Smart Campaign** was triggered with the right **program tokens**.

For repeatable testing, use **Evaluations/Prompt Flow** to run batches of sample leads and compare outcomes.

Creating Additional Logic Apps

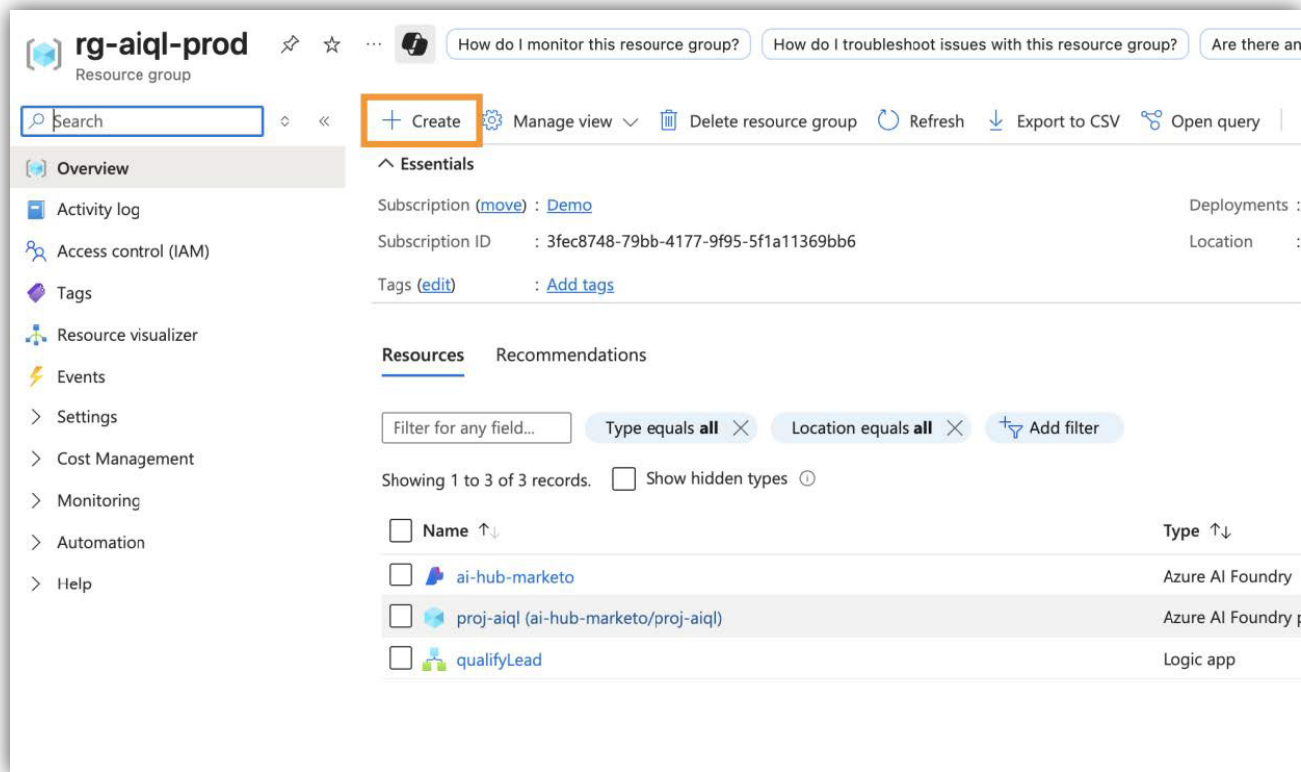
I've created templates for each of the logic apps I'm using, however you may want to create your own from scratch. This is how you can do this.

In order to use a Logic App with our AI Agent it must meet specific requirements

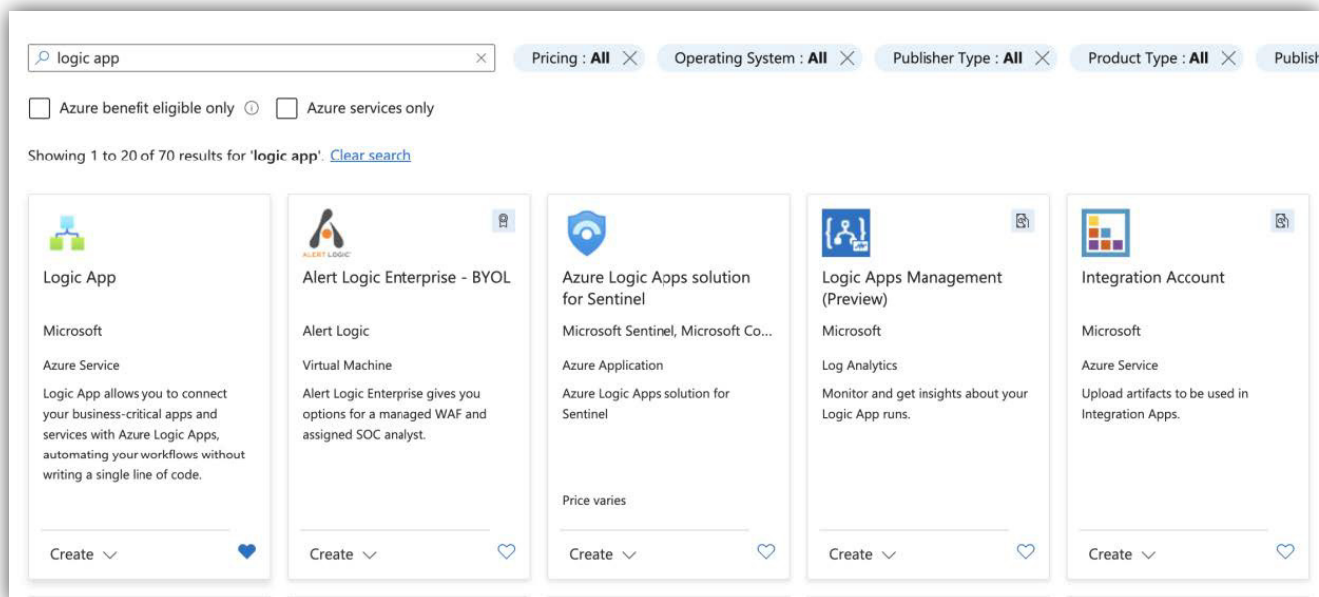
1. It must use a **Consumption** plan
2. It must be in the same subscription as your agent
3. It must have a **When a HTTP request is received** trigger with a description
4. It must have an **Response** action as well to pass back data to the agent

If you do not see your Logic App listed in your available actions, ensure it meets all of these criteria.

Creating Additional Logic Apps



Select Logic App > Create



Creating Additional Logic Apps

Select Consumption

Home > rg-aiql-prod > Marketplace >

Create Logic App

Select a hosting option

These hosting plans determine the resource allocation, scaling and pricing for your app. [Learn more about Logic App hosting options](#)

	Consumption	
Hosting plans	Multi-tenant Fully managed and easy to get started.	Workflow Service Plan Single tenant runtime with in-app connectors and scaling features.
Compute	Shared	Dedicated
Networking	Public cloud	VNET Integration
Pricing	Pay-per-operation	Per workflow service plan instance

Give the logic app a clear name and click **Review + Create**

Home > rg-aiql-prod > Marketplace > Create Logic App >

Create Logic App (Multi-tenant)

Basics Tags Review + create

Create a logic app, which lets you group workflows as a logical unit for easier management, deployment and sharing of resources. Workflows let you connect your business-critical apps and services with Azure Logic Apps, automating your workflows without writing a single line of code.

Project Details

Select a subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * ⓘ Demo

Resource Group * ⓘ rg-aiql-prod
[Create new](#)

Instance Details

Logic App name * qualifyLead ✓

Region * East US

Enable log analytics * ☐ Yes ☒ No

Creating Additional Logic Apps

Once deployed go to the logic app and click **Edit**

The screenshot shows the Azure Logic Apps portal for a logic app named 'qualifyLead'. The left sidebar contains a navigation menu with options: Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Resource visualizer, Development Tools, Settings, Monitoring, Automation, and Help. The main area displays the 'Overview' tab. At the top, there's a search bar and a toolbar with actions: Run, Refresh, Edit, Delete, Disable, Clone, Open in mobile, Export, and Provide feedback. Below the toolbar, the 'Essentials' section lists key properties: Resource group (rg-aiql-prod), Location (Sweden Central), Subscription (Demo), Subscription ID (3fec8748-79bb-4177-9f95-5f1a11369bb6), Workflow URL (https://prod-24.swedencentral.logic.azure.com:443/workflows/3...), and Tags (Add tags). To the right, a table shows additional details: Definition (1 trigger, 2 actions), Status (Enabled), Runs last 24 hours (0 successful, 0 failed), and Integration Account (--). Below this, there are tabs for 'Get started', 'Run history' (selected), 'Trigger history', and 'Metrics'. Under the 'Run history' tab, there are buttons for 'Resubmit' and 'Add filter', followed by a text input field with the placeholder 'Specify the run identifier to open monitor view directly'. At the bottom, a table header is visible with columns: Identifier, Status, Start time (Local Time), and Duration.

For requesting Smart Campaigns our logic app will need 3 parts:

Click **Add a trigger**

A button with a dark background and a white icon of a document with a downward arrow. To the right of the icon, the text 'Add a trigger' is displayed in white.

Select **When a HTTP request is received**

A card with a light blue background and a dark blue border. On the left, there is a circular icon with a globe and a document. To the right of the icon, the text 'When a HTTP request is received' is displayed in dark blue.

Creating Additional Logic Apps

Important – The HTTP request must have a description

The screenshot shows the configuration page for the 'When a HTTP request is received' trigger in Logic Apps. At the top, there's a header with a globe icon and the text 'When a HTTP request is received'. Below this is a notification box stating: 'Changing the trigger name updates the callback URL when you save the workflow.' with a close button. Underneath is a section titled 'Add a description' with a dropdown arrow. The main configuration area has four tabs: 'Parameters' (selected), 'Settings', 'Code view', and 'About'. Under the 'Parameters' tab, there are three sections: 'HTTP URL' with a text box containing 'URL will be generated after save' and a copy icon; 'Method' with a dropdown menu set to 'Default (Allow All Methods)'; and 'Request Body JSON Schema' with a large empty text area. At the bottom of the 'Request Body JSON Schema' section is a link that says 'Use sample payload to generate schema'.

Under Request Body JSON Schema, enter some version of the schema below. The only required parameter is leadID, which allows the agent to pass in the correct lead ID. The other parameters will be tokens that will be passed to Marketo Engage.

```
{
  "type": "object",
  "properties": {
    "aiSummary": {
      "description": "AI generated summary of why the lead is qualified",
      "type": "string"
    },
    "leadScore": {
      "description": "score 1-100 for the lead dec",
      "type": "string"
    },
    "leadID": {
      "description": "Marketo lead id",
      "type": "integer"
    }
  }
}
```

Creating Additional Logic Apps

Next add an action and select Request Campaign

If you haven't created a Marketo connection yet you will need to create one with your munchkin ID, client ID and client secret. Check out the [authentication documentation](#) here.

Create connection

Request Campaign

Create a new connection

Connection Name *

demo-sandbox

MunchkinID * ⓘ

123-abc-456

Client ID * ⓘ

Client ID

Client Secret * ⓘ

Client Secret

Request Campaign

Parameters

Settings

Code view

Testing

About

The Id Of The Campaign To Trigger *

1239

Advanced parameters

Showing 2 of 2

Show all

Clear all

Leads

Id - 1

leadID x

Add new item

Tokens

Name - 1

aiSummary

Value - 1

aiSummary x

Name - 2

leadScore

Value - 2

leadScore x

Add new item

+

Response

+

Ongoing improvement loop

1. Review **Sales feedback** and explainability notes attached to qualified leads.
2. Tune **ICP** and **Instructions** (e.g., adjust how owner vs buyer is detected).
3. Expand **Tools** (e.g., add “Data Health Review”, “Prospecting” later).
4. Gradually reduce person-in-the-loop rules as the system's confidence and accuracy improve.

Quick Checklist

1. Azure Account Created
2. Marketo API Credentials Obtained
3. Azure Resource Group Created
4. Azure AI Studio Workspace & Agent Deployed
5. Agent Instructions Defined
6. ICP Document Uploaded to Vector Store
7. Program & Tokens Created
8. Watch List Created
9. Smart Campaigns Created & Activated (Qualify, Disqualify, Nurture, HR)
10. enrichLeadData Logic App Deployed & Configured
11. getLeadActivity Logic App Deployed & Configured
12. requestMarketoCampaign Logic App Deployed & Configured
13. All Logic Apps Registered as Tools in Agent
14. Trigger Logic App Deployed & Scheduled
15. Manual End-to-End Test Completed Successfully
16. Evaluation Run with Batch of Leads Completed

Adobe