COMMUNICATION BRIDGE FOR WEBSITE DEVELOPMENT TEAM

Abstract

This project, titled **“COMMUNICATION BRIDGE FOR WEBSITE DEVELOPMENT TEAM,”** is developed by the **Connection Team** to automate and streamline collaboration between the **Meetly Team** and the **Website Development Team**.

The **Connection Team** acts as an intelligent bridge — receiving meeting transcripts from the Meetly Team via Gmail, analyzing them with **AWS Bedrock (Claude 3.5 Sonnet)** through an automated **n8n workflow**, and converting them into standardized **Website Instruction JSONs**. These structured instructions are then sent to the Website Team through a webhook for immediate use in website creation.

If the Website Team requires additional assets such as images or voice scripts, their JSON-based requests are automatically formatted into readable emails and sent back to the Meetly Team. Every step, including AI analysis and data exchange, is securely logged into **AWS DynamoDB** via **Lambda** and **API Gateway**.

By implementing this workflow, the **Connection Team** eliminates manual communication gaps, improves coordination, and establishes an efficient, AI-driven automation pipeline for website development projects.

Project Overview

The **Connection Team Workflow** automates and streamlines communication between three key teams involved in website development — **Meetly Team**, and **Website Development Team**.

- **Meetly Team** conducts client meetings and sends the project transcript (in JSON format) via Gmail.

- The **Connection Team**, through an automated **n8n workflow**, extracts the JSON attachment and analyzes it using an **AI Agent powered by AWS Bedrock (Claude 3.5 Sonnet)**.

- The AI interprets the client's use case, identifies the website category (e.g., hospital, education, restaurant), and generates a structured **Website Instruction JSON** containing sections, themes, and required content.

- The **Website Team** receives this instruction automatically through a **webhook endpoint**, enabling them to begin website creation instantly.

- If the Website Team needs additional media assets (images, voice scripts, etc.), their JSON request is automatically converted to a plain-text email and sent back to the **Meetly Team** for fulfillment.

- Every step of the workflow — from receiving transcripts to AI analysis, data exchange, and email communication — is logged into **AWS DynamoDB** using **Lambda** and **API Gateway** for traceability and monitoring.

This system transforms what was once a **manual, error-prone coordination process** into an **intelligent, automated, and fully traceable communication bridge**, ensuring faster project turnaround and improved collaboration among all teams.

Objective

The primary objective of the “Communication Bridge for Website Development Team” project is to design and implement an intelligent automation workflow that eliminates manual coordination between the Meetly Team and the Website Development Team.

Developed by the Connection Team, this system acts as a smart intermediary layer that:

- Automatically receives client meeting transcripts shared by the Meetly Team in JSON format via Gmail.

- Uses AI (AWS Bedrock – Claude 3.5 Sonnet) to interpret and transform the transcripts into a standardized website instruction format, enabling the Website Team to begin development without delays.

- Automatically handles return communication, such as image or voice script requests, by converting technical JSON responses into readable emails for the Meetly Team.

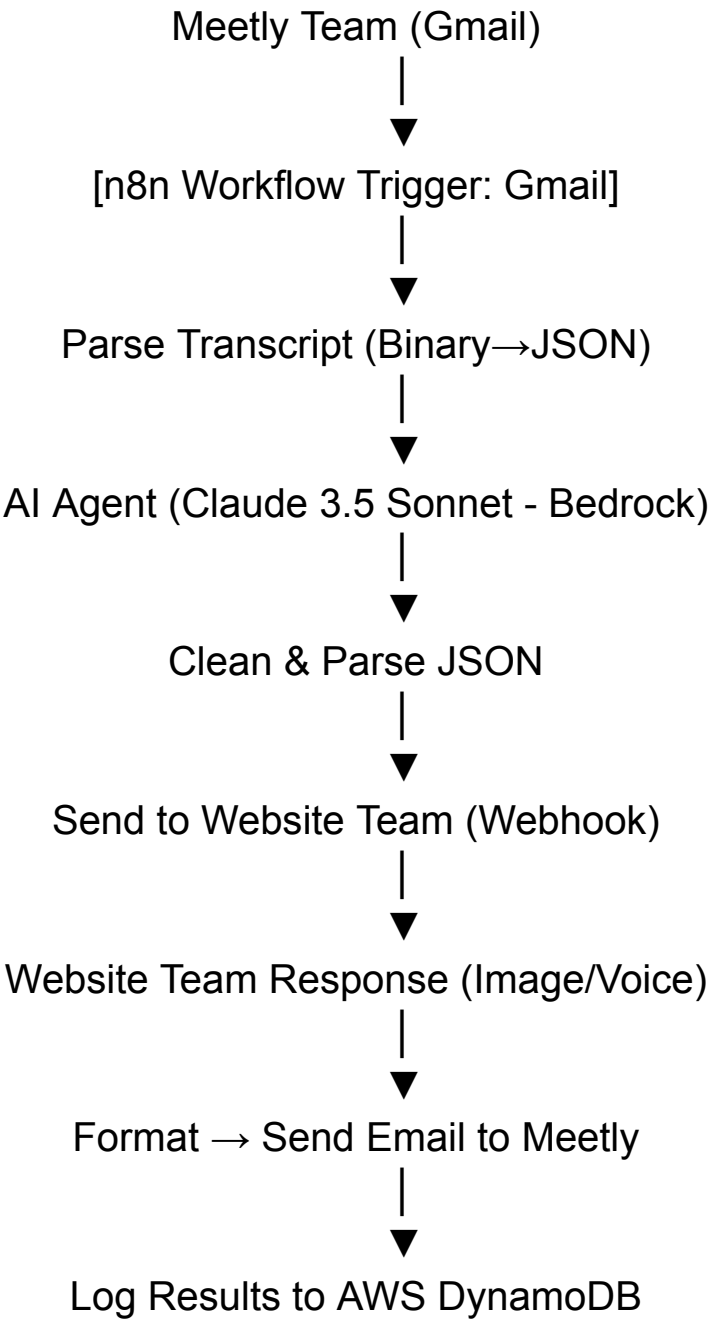
- Ensures transparency and reliability by logging every workflow step (AI output, data exchange, and communication flow) into AWS DynamoDB through Lambda and API Gateway integration.

By achieving these objectives, the Connection Team’s automation workflow establishes a seamless, AI-powered communication channel, minimizing errors, improving clarity, and significantly reducing the time taken to transfer client requirements between teams.

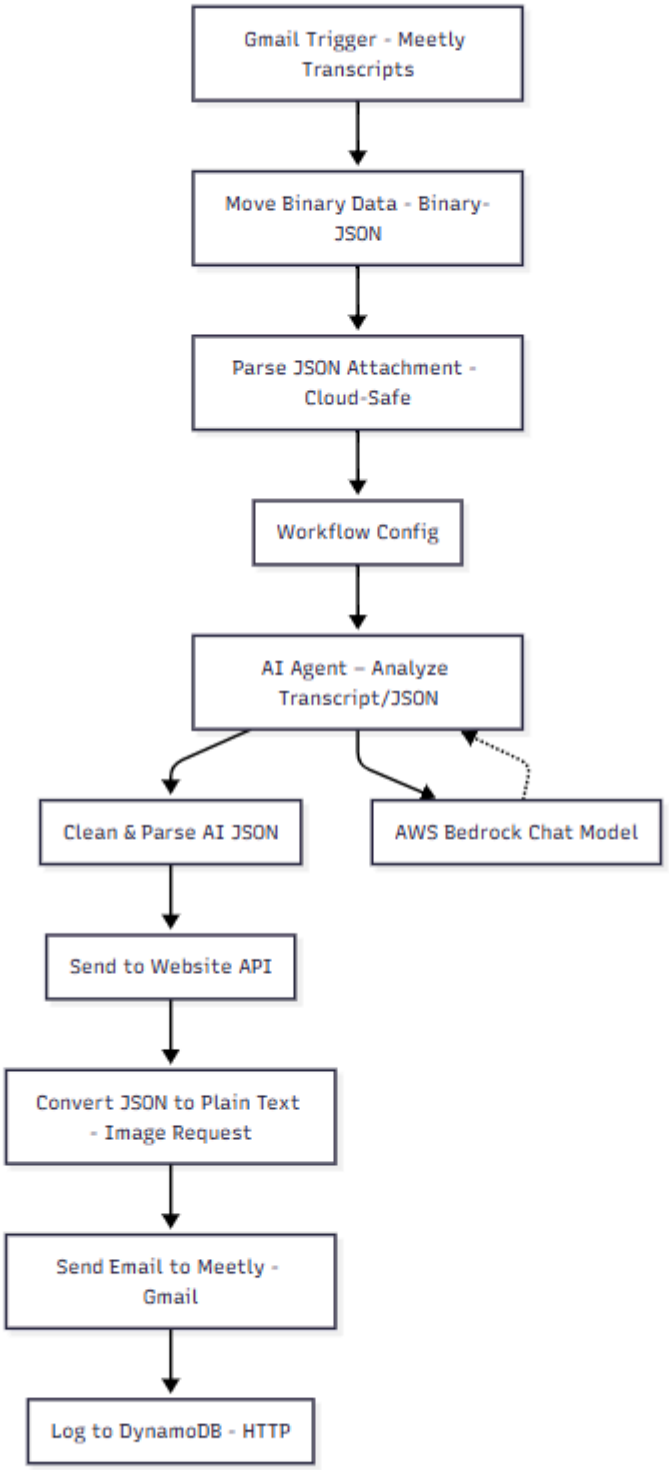
Technologies and Tools Used

Category	Tool / Service	Purpose
Workflow Automation	**n8n**	Acts as the main orchestration tool to automate the entire workflow between Gmail, AI, and APIs.
Cloud AI	**AWS Bedrock (Claude 3.5 Sonnet)**	Analyzes meeting transcripts and converts them into structured website instruction JSONs.
Email Service	**Gmail API (n8n Node)**	Used to automatically receive client transcripts and send formatted emails to the Meetly Team.
Database	**AWS DynamoDB**	Stores detailed logs of workflow executions, AI analysis, and team communications for auditing.
Serverless Function	**AWS Lambda**	Writes and processes log data into DynamoDB via API Gateway.
API Management	**AWS API Gateway**	Provides a secure HTTP endpoint for communication between n8n and AWS Lambda.
Webhooks	**n8n Cloud Webhook / Website API**	Enables real-time data exchange between the workflow and the Website Development Team.
Scripting Languages	**JavaScript & Python (n8n Code Nodes)**	Used within the workflow for JSON parsing, transformation, and plain-text message generation.

Architecture Diagram



Mermaid diagram



Step-by-Step Implementation Process

Step 1: Gmail Trigger – Meetly Transcripts

- Automatically monitors Gmail for new emails from the Meetly Team.
- Downloads attached transcript files (trans.json) in real-time.
- Initiates the workflow whenever a valid transcript email is received.

Step 2: Move Binary Data (Binary → JSON)

- Converts the attached binary JSON file into text-readable format.
- Makes the data accessible for parsing and AI processing.
- Bridges raw Gmail attachments with structured data handling in n8n.

Step 3: Parse JSON Attachment (Cloud-Safe)

- Validates the JSON file to ensure it's properly structured.
- Removes unwanted spaces or invalid characters.
- Prepares the data for accurate AI analysis in later steps.

Step 4: Workflow Config

- Stores dynamic URLs like `websiteApiUrl` and `logApiUrl`.
- Centralizes configuration for easy updates or environment changes.
- Ensures all connected APIs stay flexible and maintainable.

Step 5: AI Agent – Analyze Transcript / JSON

- Uses **Claude 3.5 Sonnet (AWS Bedrock)** to interpret the meeting transcript.
- Extracts use case details such as website type, theme, and sections.
- Generates structured “Website Instruction JSON” for the Website Team.

Step 6: Clean & Parse AI JSON

- Cleans the AI output by removing unwanted characters or formatting.
- Extracts valid JSON blocks and normalizes key fields.
- Ensures a consistent structure like `website_type`, `business_name`, and `sections_required`.

Step 7: Send to Website API

- Sends the processed JSON to the Website Team’s webhook endpoint.
- Shares complete website instructions automatically.
- Waits for and receives their response about required assets or voice scripts.

Step 8: Convert JSON to Plain Text (Image Request)

- Converts the Website Team’s asset request JSON into readable email text.
- Lists image and voice script requirements clearly for communication.
- Generates formatted subject and body for the Meetly Team email.

Step 9: Send Email to Meetly (Gmail)

- Sends the formatted message to the Meetly Team via Gmail.
- Notifies them about all required images and voice scripts.
- Completes the feedback loop between the Website and Meetly Teams.

Step 10: Log to DynamoDB (HTTP)

- Provides a central dashboard for Connection Team activity tracking.
- Posts workflow logs to AWS DynamoDB through API Gateway + Lambda.
- Enables audit tracking and process monitoring for the Connection Team.

Code Snippets for Key Components

➢ **AI Agent Prompt (LangChain Node)**

You are an intelligent website analyzer working in the Connection Team.

You receive structured JSON input that describes a website use case (e.g., hospital, restaurant, education platform, e-commerce, NGO, or corporate site).

Your job is to:
1. Detect the main domain or topic of the website based on its title, sections, and keywords.

2. Summarize the website's purpose.
3. Produce a standardized "website instruction JSON" that can be sent directly to the Website Building Team.

Title rule:
If the input JSON already includes a title or business_name, keep it EXACTLY as written — do not shorten, modify, or reformat.

Domain detection guidance:
- If words like "menu", "chef", "dining", "food", "restaurant", or "reservation" appear → topic = "Restaurant".
- If words like "hospital", "doctor", "patient", "healthcare", or "medical" → topic = "Hospital".
- If words like "course", "education", "student", "learning", "academy" → topic = "Education".
- If words like "product", "shop", "cart", "store", "e-commerce" → topic = "E-commerce".
- If unclear, set topic = "Business/Organization".

Output rules:
- Return only valid JSON (no explanations, markdown, or code fences).
- If some details are missing, infer reasonable defaults.
- Keep section names descriptive but concise.
- Always include 3–6 relevant images in "images_required".

Output Format:

```
{
  "mode": "website_instruction",
  "topic": "<detected website topic>",
  "summary": "<one-sentence overview>",
  "website": {
      "title": "<exact title from input>",
      "theme": "<color scheme or design style>",
      "sections": [
      { "name": "<section name>", "details": "<brief description>" }
      ],
      "images_required": [
      "<list of image ideas>"
      ]
  }
}
```

Input JSON:
{{$json}}

➢ **Clean & Parse AI JSON**

```
// Step 1: Extract the raw string from AI output
let raw = $json.output || $json.text || $json.data || JSON.stringify($json);

// Step 2: Find and extract only the JSON portion
const firstBrace = raw.indexOf('{');
const lastBrace = raw.lastIndexOf('}');
if (firstBrace === -1 || lastBrace === -1) {
    return [{ json: { error: 'No JSON block found in AI output', raw_output: raw } }];
```

```
}
const jsonString = raw.slice(firstBrace, lastBrace + 1);

// Step 3: Parse the extracted JSON safely
let parsed;
try {
  parsed = JSON.parse(jsonString);
} catch (err) {
   return [{ json: { error: 'Invalid JSON format', details: err.message,
raw_json: jsonString } }];
}

// Step 4: Extract the fields
const website = parsed.website || {};
const topicRaw = parsed.topic || '';
const title = website.title || parsed.title || 'Unnamed Website';
const sections = (website.sections || []).map(s => s.name ||
'').filter(Boolean);

// Step 5: Normalize the topic
let website_type = topicRaw.toLowerCase();
if (website_type.includes('hospital') ||
website_type.includes('healthcare')) website_type = 'hospital';
else if (website_type.includes('education')) website_type =
'education';
else if (website_type.includes('restaurant')) website_type =
'restaurant';
else website_type = 'business';

// Step 6: Build final Website API payload
const result = {
  website_type,
  business_name: title,
  sections_required: sections
};
```

```
// Step 7: Return
return [{ json: result }];
```

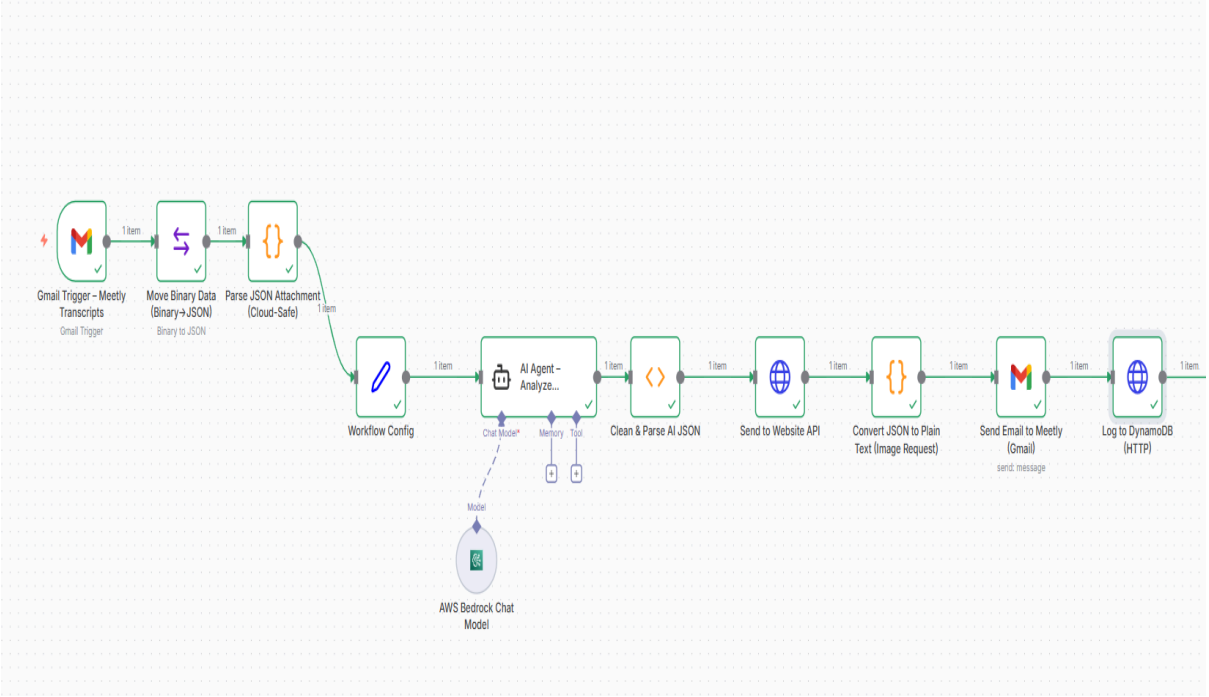
➢ **Log to DynamoDB (HTTP)**

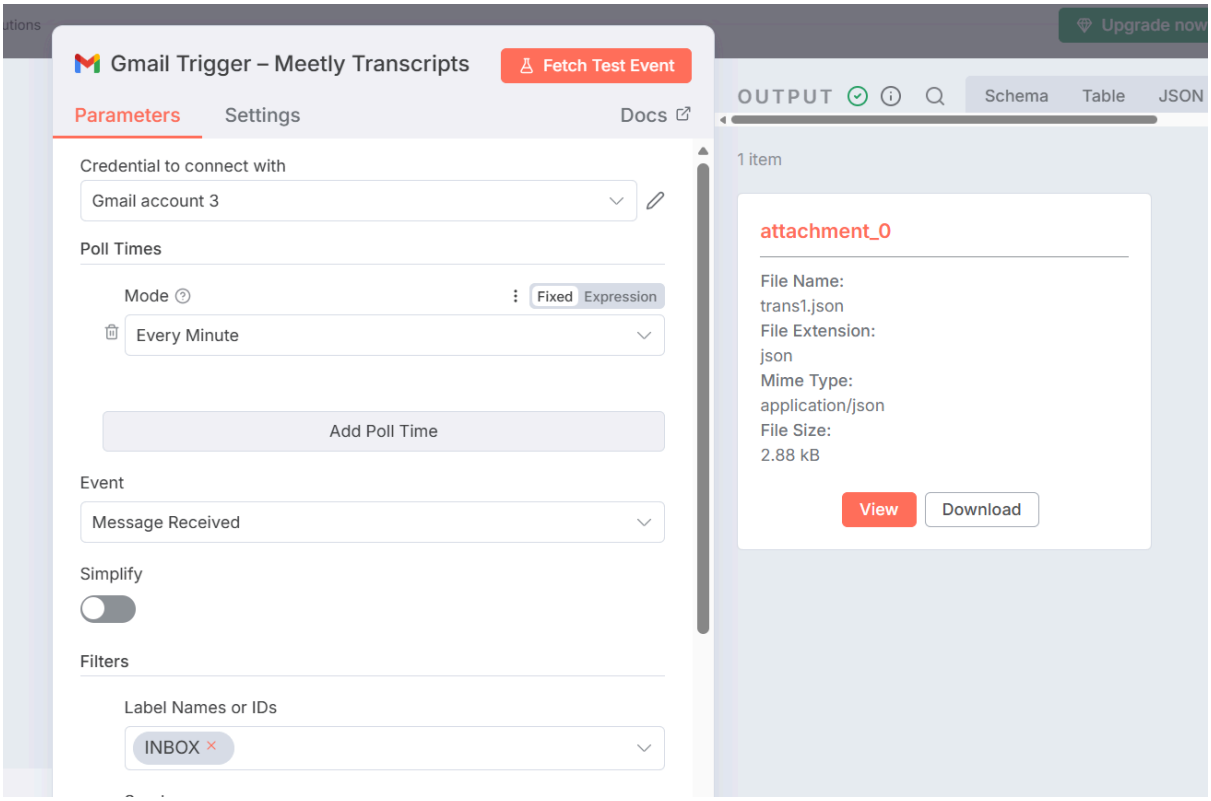
Js code:

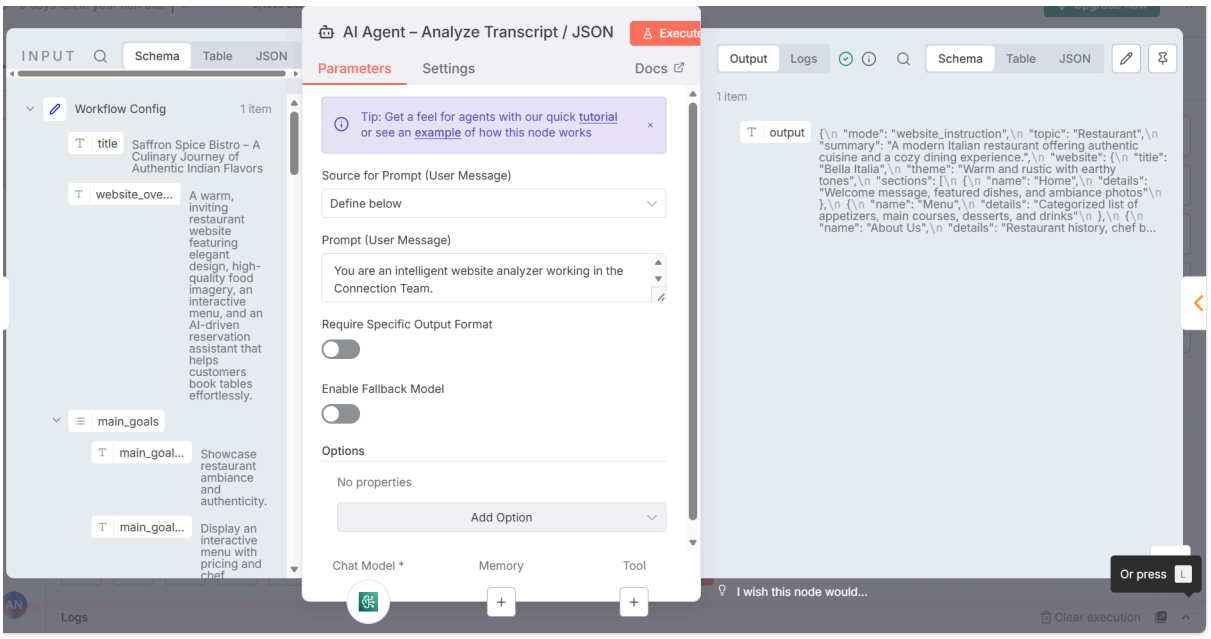
```
{{ JSON.stringify({
  executionId: $execution.id,
  workflowName: $workflow.name || 'Connection-Team-Workflow',
  timestamp: new Date().toISOString(),
              website_type:       $node["Clean       &       Parse       AI
JSON"].json.website_type,
            business_name:       $node["Clean       &       Parse       AI
JSON"].json.business_name,
  bedrock_model: $node["AWS Bedrock Chat Model"].json.model,
    ai_output_summary:  $node["AI  Agent  –  Analyze  Transcript  /
JSON"].json.output,
  status: 'completed'
}) }}
```

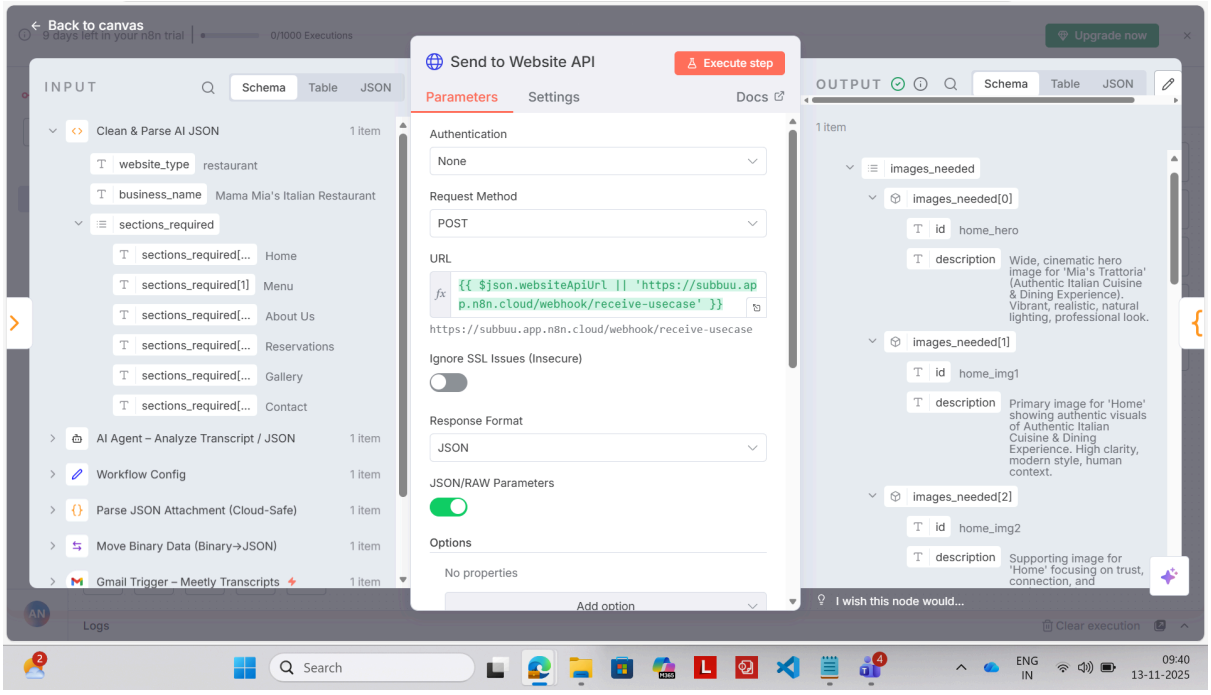
Screenshots

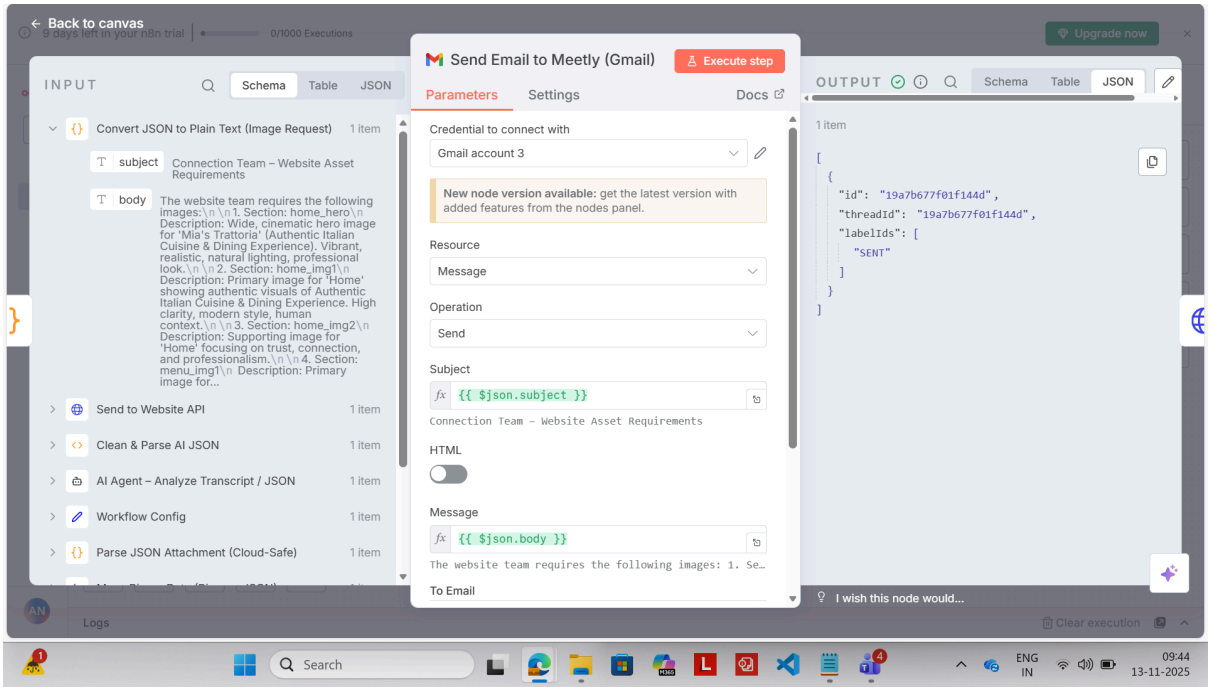
Workflow

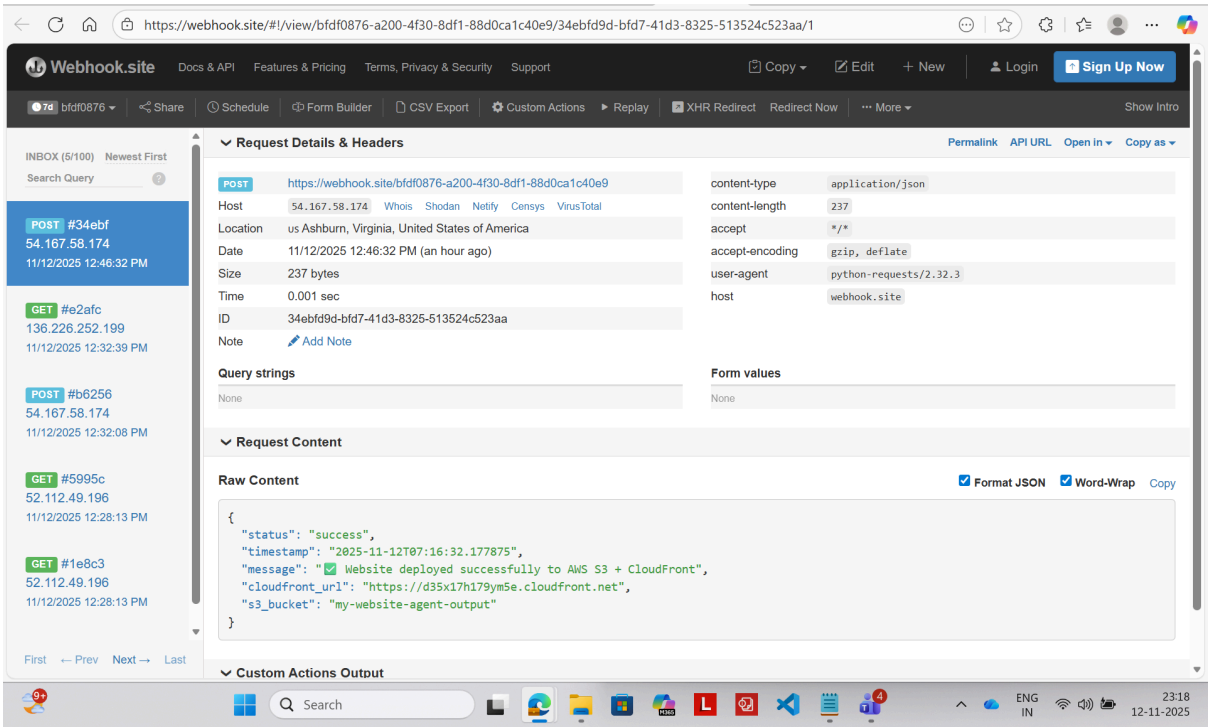




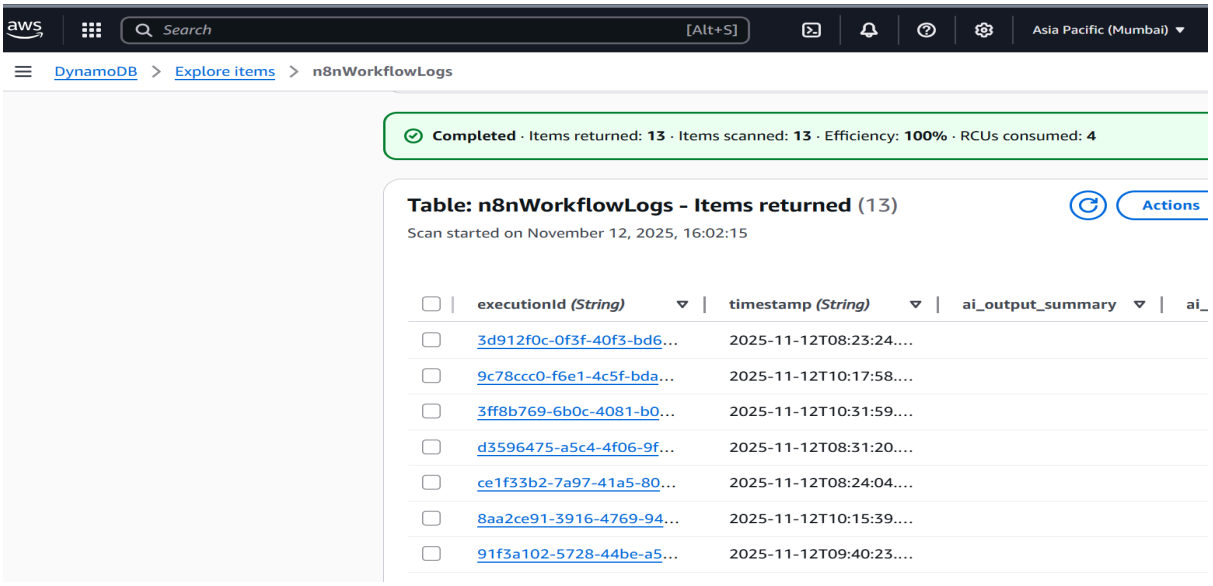

INPUT
Schema
Table
JSON
Workflow Config
1 item
title
Saffron Spice Bistro – A Culinary Journey of Authentic Indian Flavors
website_ove...
A warm, inviting restaurant website featuring elegant design, high-quality food imagery, an interactive menu, and an AI-driven reservation assistant that helps customers book tables effortlessly.
main_goals
main_goal...
Showcase restaurant ambiance and authenticity.
main_goal...
Display an interactive menu with pricing and chef
AI Agent – Analyze Transcript / JSON
Execute
Parameters
Settings
Docs
Tip: Get a feel for agents with our quick tutorial or see an example of how this node works
Source for Prompt (User Message)
Define below
Prompt (User Message)
You are an intelligent website analyzer working in the Connection Team.
Require Specific Output Format
Enable Fallback Model
Options
No properties
Add Option
Chat Model *
Memory
Tool
Output
Logs
Schema
Table
JSON
1 item
output
{\n "mode": "website_instruction",\n "topic": "Restaurant",\n "summary": "A modern Italian restaurant offering authentic cuisine and a cozy dining experience.",\n "website": {\n "title": "Bella Italia",\n "theme": "Warm and rustic with earthy tones",\n "sections": [\n {\n "name": "Home",\n "details": "Welcome message, featured dishes, and ambiance photos"\n },\n {\n "name": "Menu",\n "details": "Categorized list of appetizers, main courses, desserts, and drinks"\n },\n {\n "name": "About Us",\n "details": "Restaurant history, chef b...
I wish this node would...
Or press
Logs
Clear execution

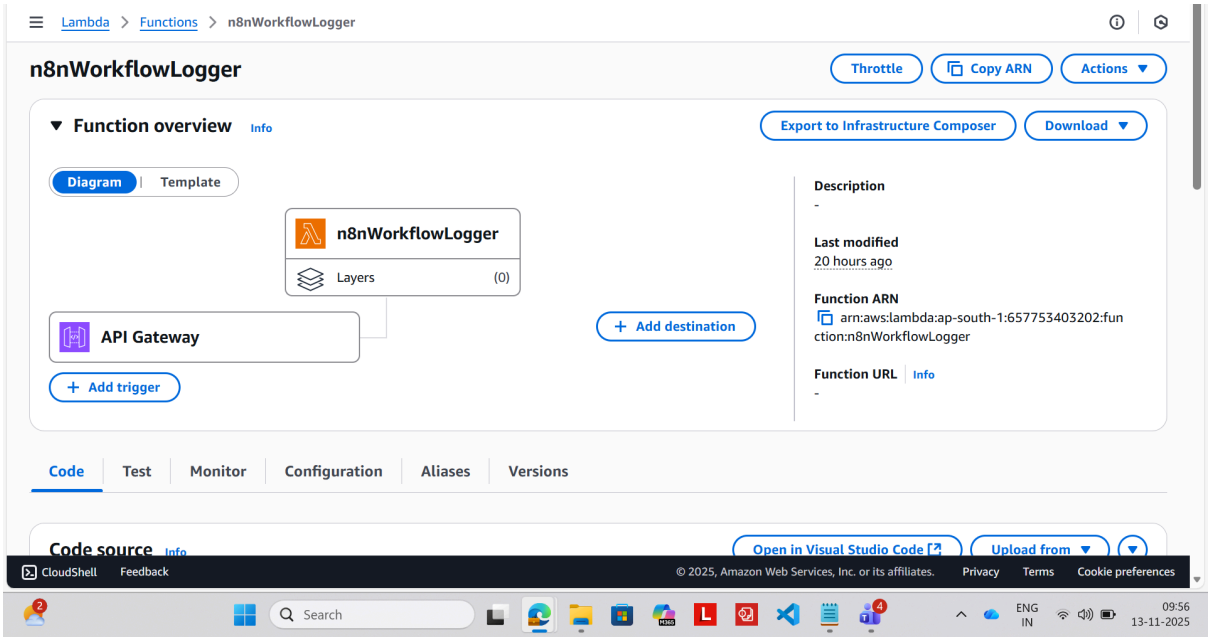

Back to canvas
9 days left in your n8n trial
0/1000 Executions
Upgrade now
INPUT
Schema
Table
JSON
Clean & Parse AI JSON
1 item
website_type
restaurant
business_name
Mama Mia's Italian Restaurant
sections_required
sections_required[...
Home
sections_required[1]
Menu
sections_required[...
About Us
sections_required[...
Reservations
sections_required[...
Gallery
sections_required[...
Contact
AI Agent – Analyze Transcript / JSON
1 item
Workflow Config
1 item
Parse JSON Attachment (Cloud-Safe)
1 item
Move Binary Data (Binary→JSON)
1 item
Gmail Trigger – Meetly Transcripts
1 item
Send to Website API
Execute step
Parameters
Settings
Docs
Authentication
None
Request Method
POST
URL
{{ $json.websiteApiUrl || 'https://subbuu.app.n8n.cloud/webhook/receive-usecase' }}
https://subbuu.app.n8n.cloud/webhook/receive-usecase
Ignore SSL Issues (Insecure)
Response Format
JSON
JSON/RAW Parameters
Options
No properties
Add option
OUTPUT
Schema
Table
JSON
1 item
images_needed
images_needed[0]
id
home_hero
description
Wide, cinematic hero image for 'Mia's Trattoria' (Authentic Italian Cuisine & Dining Experience). Vibrant, realistic, natural lighting, professional look.
images_needed[1]
id
home_img1
description
Primary image for 'Home' showing authentic visuals of Authentic Italian Cuisine & Dining Experience. High clarity, modern style, human context.
images_needed[2]
id
home_img2
description
Supporting image for 'Home' focusing on trust, connection, and
I wish this node would...
Logs
Clear execution
Search
ENG
IN
09:40
13-11-2025

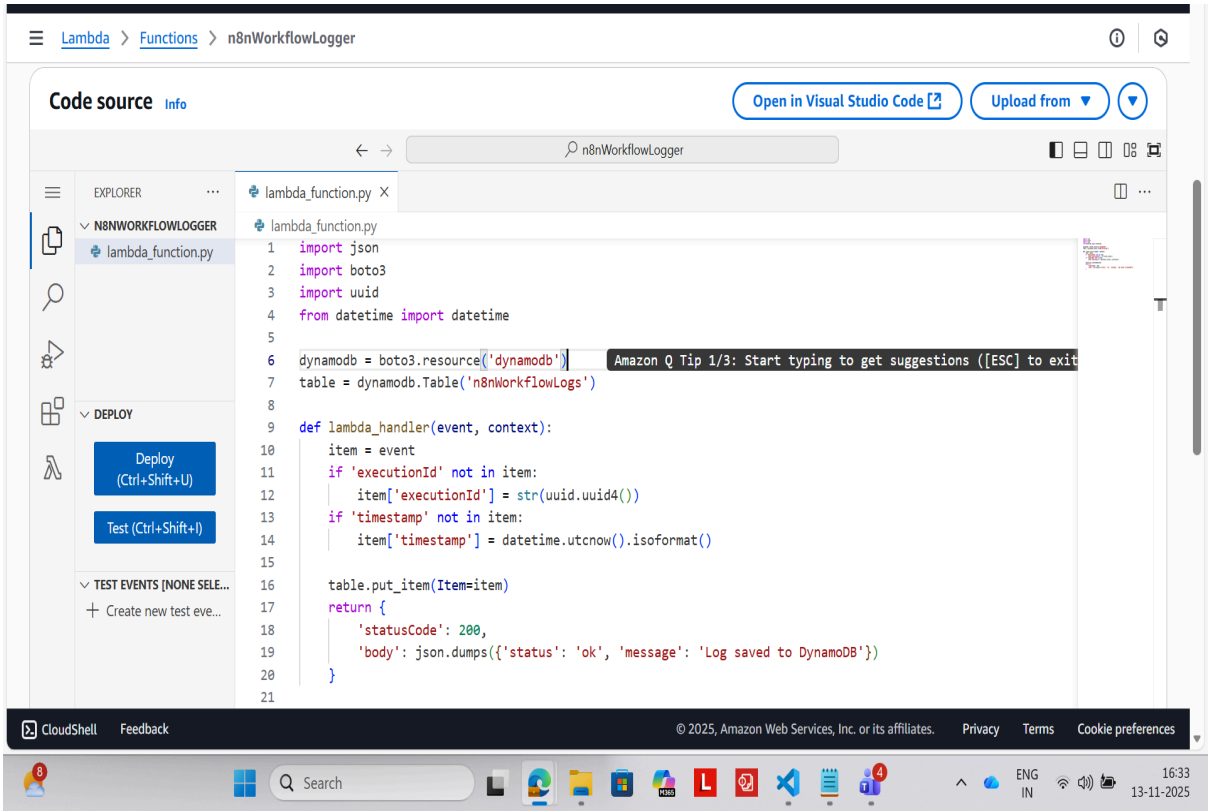

Back to canvas
Upgrade now
INPUT
Schema
Table
JSON
Convert JSON to Plain Text (Image Request)
1 item
subject
Connection Team – Website Asset Requirements
body
Send to Website API
Clean & Parse AI JSON
AI Agent – Analyze Transcript / JSON
Workflow Config
Parse JSON Attachment (Cloud-Safe)
Send Email to Meetly (Gmail)
Execute step
Parameters
Settings
Docs
Credential to connect with
Gmail account 3
New node version available: get the latest version with added features from the nodes panel.
Resource
Message
Operation
Send
Subject
{{ $json.subject }}
Connection Team – Website Asset Requirements
HTML
Message
{{ $json.body }}
To Email
OUTPUT
1 item
"id": "19a7b677f01f144d",
"threadId": "19a7b677f01f144d",
"labelIds": [
"SENT"
I wish this node would...
Logs
Clear execution
Search
ENG
IN
09:44
13-11-2025

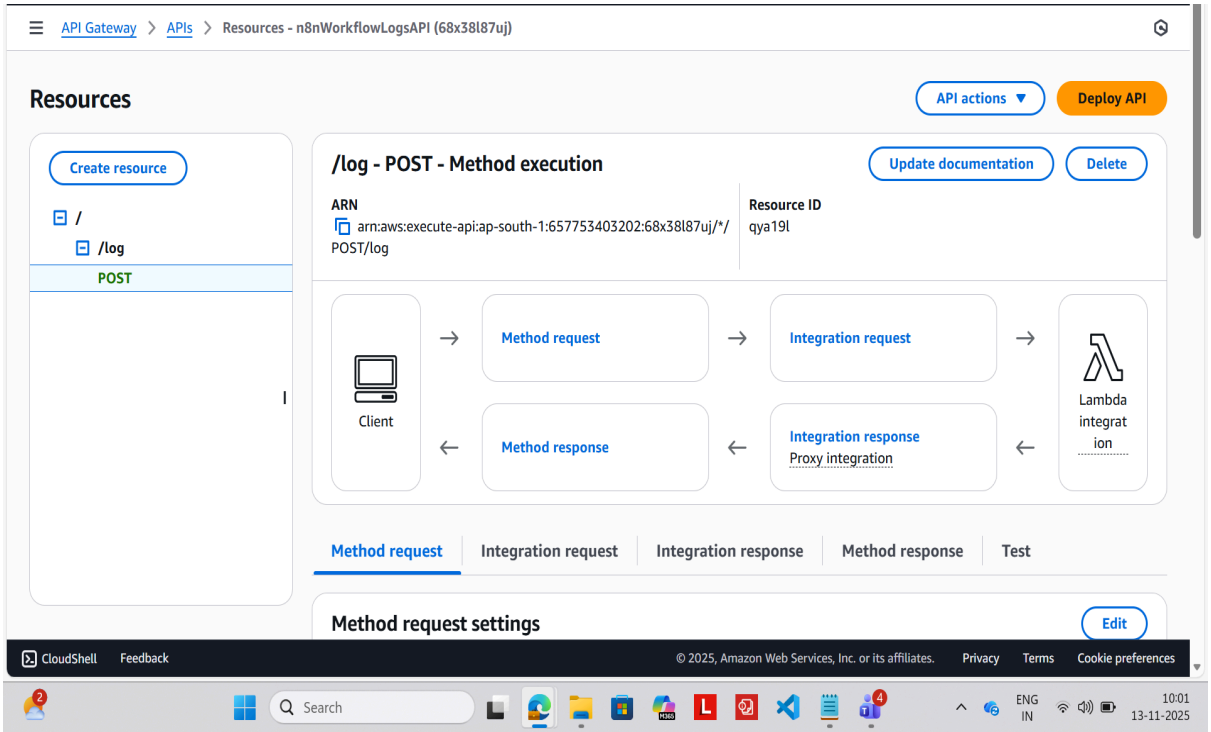

https://webhook.site/#!/view/bfdf0876-a200-4f30-8df1-88d0ca1c40e9/34ebfd9d-bfd7-41d3-8325-513524c523aa/1
Webhook.site
Docs & API
Features & Pricing
Terms, Privacy & Security
Support
Copy
Edit
New
Login
Sign Up Now
bfdf0876
Share
Schedule
Form Builder
CSV Export
Custom Actions
Replay
XHR Redirect
Redirect Now
More
Show Intro
INBOX (5/100)
Newest First
Search Query
POST #34ebf
54.167.58.174
11/12/2025 12:46:32 PM
GET #e2afc
136.226.252.199
11/12/2025 12:32:39 PM
POST #b6256
54.167.58.174
11/12/2025 12:32:08 PM
GET #5995c
52.112.49.196
11/12/2025 12:28:13 PM
GET #1e8c3
52.112.49.196
11/12/2025 12:28:13 PM
First
Prev
Next
Last
Request Details & Headers
Permalink
API URL
Open in
Copy as
POST https://webhook.site/bfdf0876-a200-4f30-8df1-88d0ca1c40e9
Host 54.167.58.174 Whois Shodan Netify Censys VirusTotal
Location us Ashburn, Virginia, United States of America
Date 11/12/2025 12:46:32 PM (an hour ago)
Size 237 bytes
Time 0.001 sec
ID 34ebfd9d-bfd7-41d3-8325-513524c523aa
Note Add Note
content-type application/json
content-length 237
accept */*
accept-encoding gzip, deflate
user-agent python-requests/2.32.3
host webhook.site
Query strings
None
Form values
None
Request Content
Raw Content
Format JSON
Word-Wrap
Copy
{
"status": "success",
"timestamp": "2025-11-12T07:16:32.177875",
"message": "✅ Website deployed successfully to AWS S3 + CloudFront",
"cloudfront_url": "https://d35x17h179ym5e.cloudfront.net",
"s3_bucket": "my-website-agent-output"
}
Custom Actions Output
Search
ENG
IN
23:18
12-11-2025

AWS Configuration






Lambda > Functions > n8nWorkflowLogger
Code source Info
Open in Visual Studio Code
Upload from
n8nWorkflowLogger
EXPLORER
lambda_function.py
N8NWORKFLOWLOGGER
lambda_function.py
DEPLOY
Deploy (Ctrl+Shift+U)
Test (Ctrl+Shift+I)
TEST EVENTS [NONE SELE...
Create new test eve...
import json
import boto3
import uuid
from datetime import datetime
dynamodb = boto3.resource('dynamodb')
Amazon Q Tip 1/3: Start typing to get suggestions ([ESC] to exit
table = dynamodb.Table('n8nWorkflowLogs')
def lambda_handler(event, context):
item = event
if 'executionId' not in item:
item['executionId'] = str(uuid.uuid4())
if 'timestamp' not in item:
item['timestamp'] = datetime.utcnow().isoformat()
table.put_item(Item=item)
return {
'statusCode': 200,
'body': json.dumps({'status': 'ok', 'message': 'Log saved to DynamoDB'})
}
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG IN
16:33 13-11-2025


API Gateway > APIs > Resources - n8nWorkflowLogsAPI (68x38l87uj)
Resources
API actions
Deploy API
Create resource
/
/log
POST
/log - POST - Method execution
Update documentation
Delete
ARN
arn:aws:execute-api:ap-south-1:657753403202:68x38l87uj/*/POST/log
Resource ID
qya19l
Client
Method request
Integration request
Lambda integration
Method response
Integration response
Proxy integration
Method request
Integration request
Integration response
Method response
Test
Method request settings
Edit
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG IN
10:01 13-11-2025

Conclusion

The **Communication Bridge for Website Development Team** successfully automates the end-to-end collaboration between the Meetly Team and the Website Development Team through an intelligent, AI-powered workflow.

By acting as the **automation and intelligence layer**, the Connection Team's workflow ensures that meeting transcripts are instantly analyzed, transformed into structured website instructions, and shared seamlessly with the development team.

This system eliminates manual intervention, reduces communication delays, and provides complete visibility through automated logging in AWS DynamoDB.
Using tools like **n8n**, **AWS Bedrock**, **Lambda**, and **API Gateway**, the workflow delivers a scalable, reliable, and efficient coordination framework.

Overall, the project demonstrates how **AI-driven automation can streamline cross-team communication**, improve accuracy, and accelerate the website development lifecycle.

Future Enhancements

- Add **automatic prompt optimization** based on historical AI results.
- Extend DynamoDB logging for AI token usage cost tracking.
- Deploy workflow as **Docker container** via AWS ECS/ECR for higher scalability.

Final Thoughts

"AI won't just automate workflows — it will manage entire ecosystems of communication."

Github Repository :
Sujitha-976/COMMUNICATION-BRIDGE-FOR-WEBSITE-DEVELOPMENT-TEAM