**Instructions to run the workflow:**

**How to Run the DreamWeaver AI Image Generator Workflow (n8n + OpenRouter + S3 + Google Sheets)**

**🛠️ Prerequisites**

* n8n installed and running (locally or on a cloud VM like GCP) — preferably inside Docker.
* Domain name connected (e.g., DuckDNS or custom domain).
* SSL Certificate installed (e.g., Certbot) for secure HTTPS access.
* OpenRouter API key for the AI model.
* Hugging Face API Key (optional) if using Hugging Face models instead.
* S3-compatible Cloud Storage credentials (like Cloudflare R2).
* Google account linked for Google Sheets integration.

**🔧 Step-by-Step Setup Instructions**

**Load the Workflow:**

* Import the existing Chat\_AI\_Agent\_Text\_to\_Image\_Convertor\_N8N.json workflow.

The workflow components:

* When Chat Message Received
* AI Agent (OpenRouter Chat Model + Simple Memory)
* If Node (Check valid/invalid prompt)
* HTTP Request (Call model)
* S3 Upload (Save generated images)
* Google Sheets (Log prompts and image URLs)
* Final Output (Confirmation to user)

**Explanation for each Node:**

| **Node** | **Configuration Details** |
| --- | --- |
| **When Chat Message Received** | Trigger based on chat input. |
| **AI Agent (Tools Agent)** | Connect OpenRouter Chat Model + Simple Memory. |
| **OpenRouter Chat Model** | API Key authentication, connect to a model |
| **Simple Memory** | Enable short-term memory (retain last 5 messages). |
| **If Node** | Condition: if $json.output contains fallback message like "Sorry", route accordingly. |
| **Build Final Prompt** | Add enhancements to the prompt for better generation (e.g., "hyperrealistic, soft lighting"). |
| **HTTP Request** | Call Hugging Face Image Generation API (POST method). |
| **S3 Upload** | Set credentials, upload binary file (data), bucket name like n8n. |
| **Google Sheets Append** | Connect to Google Sheets, map columns (Prompt, Image URL). |
| **Final Output** | Send the final confirmation back with the image URL or fallback message. |

**Set Credentials:**

* **Google Sheets OAuth2 Credentials**: For appending prompt + URL.
* **S3 Storage Credentials**: For uploading generated images.
* **OpenRouter API Key**: For Chat Model and/or HTTP Request.
* **Hugging Face API Key**: For using Hugging Face endpoints.

**List of Crendentials:**

**OpenRouter Account API key:**

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

**Hugging Face API Key:**

hf\_yzgaOXPgKUeCPCGjwWjkwapDIabUlnZpgR

**Cloudflare S3 account:**

S3 Endpoint:

<https://8dad9ab9a76ded150197384ab9150ba8.r2.cloudflarestorage.com>

Access Key ID:

ad098dbcf865a0e53b261be7bf68d48e

Secret Access Key:

sk-proj-UEyzDcLxekfn3ZfxLHs8KtTbzkGvXomAIljy2B3NR5QwDWTrud-rwXDWRmnlhUZvHQgJAHd71cT3BlbkFJK5187sLWcQWl\_a91Vx\_rlScVm2cCARCZxmdxJo8RvtoChHVBzcxdtujVnyk0VjxWppbiqrtskA

**Google Sheets account**

OAuth Redirect URL:

<http://localhost:5678/rest/oauth2-credential/callback>

Client ID:

686640765282-je5ded8tddei6nhenvq9tct35vei2l0j.apps.googleusercontent.com

Client Secret:

key=AIzaSyDgfVuvDhEuEjmpQ5yPcyZsIVYmuwW-\_2U

**How to Run the WhatsApp AI Chatbot Workflow**

**1. Pre-requisites**

✅ **n8n Instance** running (self-hosted on GCP or any other server).  
✅ **Docker installed** if using containerized deployment.  
✅ **WhatsApp Business Cloud API account** (Meta/Facebook Developer account).  
✅ **OpenAI API Key** for LLM (e.g., GPT-4o-mini model).  
✅ **SSL Certificate** setup (for secure webhook callbacks — Let's Encrypt recommended).  
✅ **Domain Name** pointing to your n8n server (e.g., via DuckDNS or Google Domains).

**2. Environment Setup**

* Make sure your **n8n** server is running and reachable via **HTTPS** (port 443).
* Install required credentials:
  + OpenAI Credentials (for LLM usage).
  + WhatsApp Cloud API Credentials (for sending and receiving WhatsApp messages).

**3. Import the Workflow**

1. Open your n8n dashboard.
2. Click on ➡️ **Import** → **Upload JSON**.
3. Upload the file you created: **WhatsApp\_AI\_Chatbot.json**.
4. Save the workflow with a recognizable name, e.g., WhatsApp AI Chatbot.

**4. Explanation for each Nodes**

* **WhatsApp Message Received Node**:
  + Set up the correct webhook in WhatsApp Business API settings (Meta Developer Console).
  + Make sure the webhook URL is:  
    https://thivyadhanasegaran-n8n.duckdns.org/
* **AI Agent Node**:
  + Connect it to the OpenAI Chat Model node and Simple Memory node properly.
* **OpenAI Chat Model Node**:
  + Ensure the correct OpenAI model is selected (e.g., gpt-4o-mini) and the API Key is correctly configured.
* **Simple Memory Node**:
  + Configure a session key linked to the WhatsApp sender ID:  
    {{ $('WhatsApp Message Received').item.json.contacts[0].wa\_id }}
* **WhatsApp Business Cloud Node**:
  + Set your Sender Phone Number ID.
  + Use expression for the outgoing text:  
    Success {{ $json.output }}

**5. Activate the Workflow**

* Toggle the workflow to **Active** in n8n.
* Test sending a WhatsApp message to your Business number.
* If setup correctly, your chatbot will respond with AI-generated answers!

**6. Quick Architecture Summary**

📥 WhatsApp Message ➔ 🤖 AI Agent (OpenAI + Simple Memory) ➔ 📤 WhatsApp Reply

* Incoming WhatsApp messages trigger the AI Agent.
* The AI Agent uses OpenAI model + session memory to generate a smart response.
* The AI reply is sent back to the user on WhatsApp automatically.

**Credentials to setup Nodes:**

**OpenAI Account Node**

API Key:

sk-proj-msyjxg0ER\_0FtMVbHOLy1UFTuN3UQLw5UEm8kctvIMRXTAGsvANl3H-1QVt8DAj8xgE0wEzO47T3BlbkFJvZDwXn48TISIIEjwmtgECCRIl4sIO9iah2AUIF0E3TnIw7cvkL6FB80\_W2xTiTFpXl5DTEMbkA

Base URL:

<https://api.openai.com/v1>

**WhatsApp Business Cloud Node**

Access Token:

sk-proj-UEyzDcLxekfn3ZfxLHs8KtTbzkGvXomAIljy2B3NR5QwDWTrud-rwXDWRmnlhUZvHQgJAHd71cT3BlbkFJK5187sLWcQWl\_a91Vx\_rlScVm2cCARCZxmdxJo8RvtoChHVBzcxdtujVnyk0VjxWppbiqrtskA

Business Account ID:

2469020056778424