**📘 Agent Assist Structured Response Using Vertex AI Playbooks**

This document outlines the full methodology and decisions taken to build a solution that allows Agent Assist to return **structured JSON responses** (including *answer*, *reasoning*, *quotes*, and *sources*) using **Vertex AI Playbooks** and **Dialogflow CX**.

**🧩 Goal**

To enhance Google Cloud Agent Assist with the ability to return **structured, validated responses** instead of just plain text, and display them in the Agent Assist UI during live customer-agent conversations.

**🛠️ Final Solution Architecture**

1. **Vertex AI Playbook**
   * Configured to handle all logic: fetch from GKA, validate against a data store, and generate structured output.
2. **Playbook Logic Includes:**
   * Custom **instructions** for reasoning and format
   * A **Python snippet** to transform the GKA response into a strict JSON
   * Conditional logic to handle edge cases where validation fails
3. **Agent Configuration**
   * Built under **"Build Your Own Agent"**
   * Set the **custom playbook as default** for the agent
4. **Conversation Profile**
   * Created and linked with the above agent
   * Agent is configured to use **Dialogflow CX** virtual agent

**🔄 Response Structure Returned by Playbook**

json

CopyEdit

{

"answer": "string",

"reasoning": "string",

"quotes": [

{

"quote": "string",

"url": "string",

"name": "string"

}

],

"sources": [1, 2, 3]

}

📌 This format ensures agents see the **validated answer**, its **justification**, and **all matching GKA snippets**.

**📋 Playbook Instructions (LLM Prompt)**

text

CopyEdit

1. Use the agent’s last question from the conversation context.

2. Use the answer provided by Generative Knowledge Assist.

3. Use all snippet passages, snippet URIs, and snippet titles provided by Generative Knowledge Assist.

4. Use the Code Interpreter output as your FINAL answer. Do NOT override it.

5. Validate the answer using ${TOOL:AQ&A Data Store} only. Never add extra information.

6. If you cannot validate the answer, or any required field is missing, respond with:

{

"answer": "I DO NOT KNOW",

"reasoning": "I DO NOT KNOW",

"quotes": [],

"sources": []

}

7. Always output ONLY this final strict JSON format:

{

"answer": "",

"reasoning": "",

"quotes": [

{

"quote": "",

"url": "",

"name": ""

}

],

"sources": []

}

For answer: Provide the verified final answer, or 'I DO NOT KNOW'.

For reasoning: Explain step-by-step how you validated the answer using the question, GKA output, snippets, URIs, titles and documentation. Write in neutral third-person.

For quotes: Include ALL valid snippets returned by GKA.

Each quote must have:

"quote": the snippet text

"url": source link

"name": document title

If no valid snippets, use [].

For sources: Include a matching index for each quote. If no valid sources, use [].

Never hallucinate or fabricate quotes or sources.

Always use the transformed JSON from the Code Interpreter as the final answer.

**🐍 Python Snippet Logic in Playbook**

python

CopyEdit

from typing import Any, Dict

def transform\_gka\_output(event: Dict[str, Any], context: Dict[str, Any]) -> Dict[str, Any]:

gka\_response = context["tools"]["AQ&A Data Store"]

suggestion = (

gka\_response["humanAgentSuggestionResults"][0]

.get("suggestKnowledgeAssistResponse", {})

.get("knowledgeAssistAnswer", {})

.get("suggestedQueryAnswer", {})

)

answer\_text = suggestion.get("answerText", "").strip()

snippets = suggestion.get("generativeSource", {}).get("snippets", [])

quotes\_list = []

sources\_list = []

for idx, s in enumerate(snippets):

quote\_text = s.get("snippet", "")

quote\_url = s.get("uri", "")

quote\_name = s.get("title", "")

quotes\_list.append({

"quote": quote\_text,

"url": quote\_url,

"name": quote\_name

})

sources\_list.append(idx + 1)

if not answer\_text:

answer\_text = "I DO NOT KNOW"

if not quotes\_list:

sources\_list = []

final\_json = {

"answer": answer\_text,

"reasoning": "", # Filled by LLM using instructions

"quotes": quotes\_list,

"sources": sources\_list

}

return {

"response": final\_json

}

**✅ Observations**

| **Aspect** | **Status** |
| --- | --- |
| Playbook logic working | ✅ |
| Custom validation via datastore | ✅ |
| Structured response format | ✅ |
| Response displays in Agent Assist UI (as JSON) | ✅ (using Dialogflow CX) |
| Custom UI for parsing structured output | ❌ *(To be explored)* |
| Multiple snippets & sources handled | ✅ |
| GKA override avoided | ✅ |

**⚠️ Known Limitations**

* Agent Assist UI **does not parse structured JSON** visually — displays it as raw text.
* Native Agent Assist UI **cannot render quote/source blocks cleanly**.
* Markdown formatting only partially supported.
* Using Dialogflow CX as agent shows raw JSON — **requires custom UI or integration** for pretty formatting.
* **No official way to directly plug playbooks into "Build Your Own Generator"** from Assist Generator UI.

**🧪 Future Work / To-Do**

* Explore **custom Agent Assist desktop UI** (using Five9, Salesforce, or custom iframe)
* Use webhook-based fulfillment to parse GKA + custom JSON formatting on backend
* Raise feature request to **Google Cloud** for better UI handling of structured GKA responses
* Try transforming structured JSON to Markdown within answerText for better UI rendering
* Evaluate **Dialogflow CX webhooks** as a workaround for advanced formatting

**🔗 References**

* Google Vertex AI Playbooks Documentation
* Dialogflow CX and Agent Assist Integration
* Custom Agent Assist UI
* Playbook Python Snippets
* GKA Response Format

**📎 Attachments**

* ✅ Screenshot of Agent Assist output (JSON response)
* ✅ Screenshot of Playbook configuration
* ✅ Screenshot of Conversation Profile settings
* ✅ Screenshot of GKA enabled in conversation profile
* ✅ Screenshot of Dialogflow CX configuration