**Business Analyst Agent**

**Overall Approach and Technical Choices**

My approach to this task is a **multi-step, agentic process** that leverages the strengths of large language models (LLMs) and tools to ensure accuracy, relevance, and clarity. The core of this strategy is to:

1. **Search:** Use a powerful tool like Google Search to find relevant, up-to-date information on the given topic.
2. **Generate:** Employ a foundational LLM to generate an initial response based on the search results.
3. **Verify & Synthesize:** Use another LLM in a "Synthesizer" role to critically evaluate, cross-reference, and refine the initial response against the original sources.
4. **Format:** Finally, format the output to be clear, well-structured, and properly cited.

This approach is implemented using the google.genai library, which allows for seamless integration of LLMs with grounding tools, such as Google Search. This provides a robust framework for building a system that can generate grounded and verifiable content.

**Strategy for Finding Reliable Sources**

My strategy for finding reliable, high-quality sources online is built into the core Google Search tool call. Instead of generating a generic response, the model is configured to use **grounding**. Grounding forces the model to base its response on specific, verifiable information retrieved from a web search. This is crucial for several reasons:

* **Up-to-Date Information:** It ensures the response isn't based on outdated or pre-trained data.
* **Fact-Checking:** The model can be prompted to verify facts by cross-referencing multiple sources.
* **Reliability:** By relying on search results, the model can pull information from reputable sources like established tech blogs (e.g., Atlassian, Productboard), product management resources, and academic papers, rather than relying on its internal, potentially biased knowledge.

The grounding functionality provides both the text and the corresponding source URIs, which are essential for the next step of synthesizing and citing the information.

**LLM Selection and Rationale**

I decided to use **Google's gemini-2.5-flash** model for both the initial response generation and the synthesis stage. The rationale behind this choice is twofold:

* **Speed and Efficiency:** As its name suggests, gemini-2.5-flash is optimized for speed and is an efficient choice for tasks that don't require the most complex reasoning. For generating an initial draft and then a refined synthesis, its speed allows for a faster overall processing time.
* **Multimodal Capabilities and Tool Use:** The gemini-2.5-flash model is part of the Gemini family, which is known for its advanced tool-use capabilities. It can seamlessly integrate with the Google Search tool, allowing it to perform the necessary searches and ground the initial response in external data.

**Synthesizer Agent Logic**

The core logic of the Synthesizer agent is defined in the synthesize\_and\_verify function. It operates with a clear, multi-step system prompt that guides the model to perform the following actions:

1. **Cross-reference:** The agent is instructed to directly compare the "INITIAL FINDINGS" (the first LLM's response) with the "SOURCES CONSULTED" (the citations provided).
2. **Identify Themes and Contradictions:** It looks for consistency across sources and flags any conflicting information, which is a key part of the verification process.
3. **Synthesize Key Points:** The agent's primary task is to distill the core concepts from the verified information. In this case, it's asked to clearly define the RICE and Kano models and explain their key differences. This moves the output beyond a simple summary to a structured explanation.
4. **Augment Data:** It's instructed to add any relevant information from the sources that was not included in the initial findings, ensuring the final output is comprehensive.
5. **Generate Visual Aids:** The prompt explicitly asks for tables or other visual aids to help clarify the comparison. This is a crucial step for improving the clarity and usability of the final document.
6. **Provide Citations:** Finally, the agent is responsible for ensuring the synthesized output is properly cited, maintaining the integrity of the information.

The Synthesizer acts as a quality control layer, ensuring that the final output is not just a regurgitation of search results but a well-reasoned, verified, and structured document.

**Project Setup and Execution**

To set up and run this project, follow these steps:

1. **Prerequisites:** You must have Python installed on your system.
2. **Install the Google GenAI SDK:** Open your terminal or command prompt and run the following command to install the required library:

Bash

pip install google-generativeai

1. **Get an API Key:** You need a valid API key to access the Google GenAI models. You can obtain one from the Google AI Studio website.
2. **Update the Code:** In the gemini.py file, replace the placeholder api\_key with your actual API key:

Python

client = genai.Client(api\_key="YOUR\_API\_KEY\_HERE")

1. **Run the Script:** Save the file and run it from your terminal using the following command:

Bash

python gemini.py

After a few moments, the script will execute, and the final synthesized document will be saved to a file named synthesized\_output.txt in the same directory.