

Agents

Authors: Julia Wiesinger, Patrick Marlow
and Vladimir Vuskovic

Google

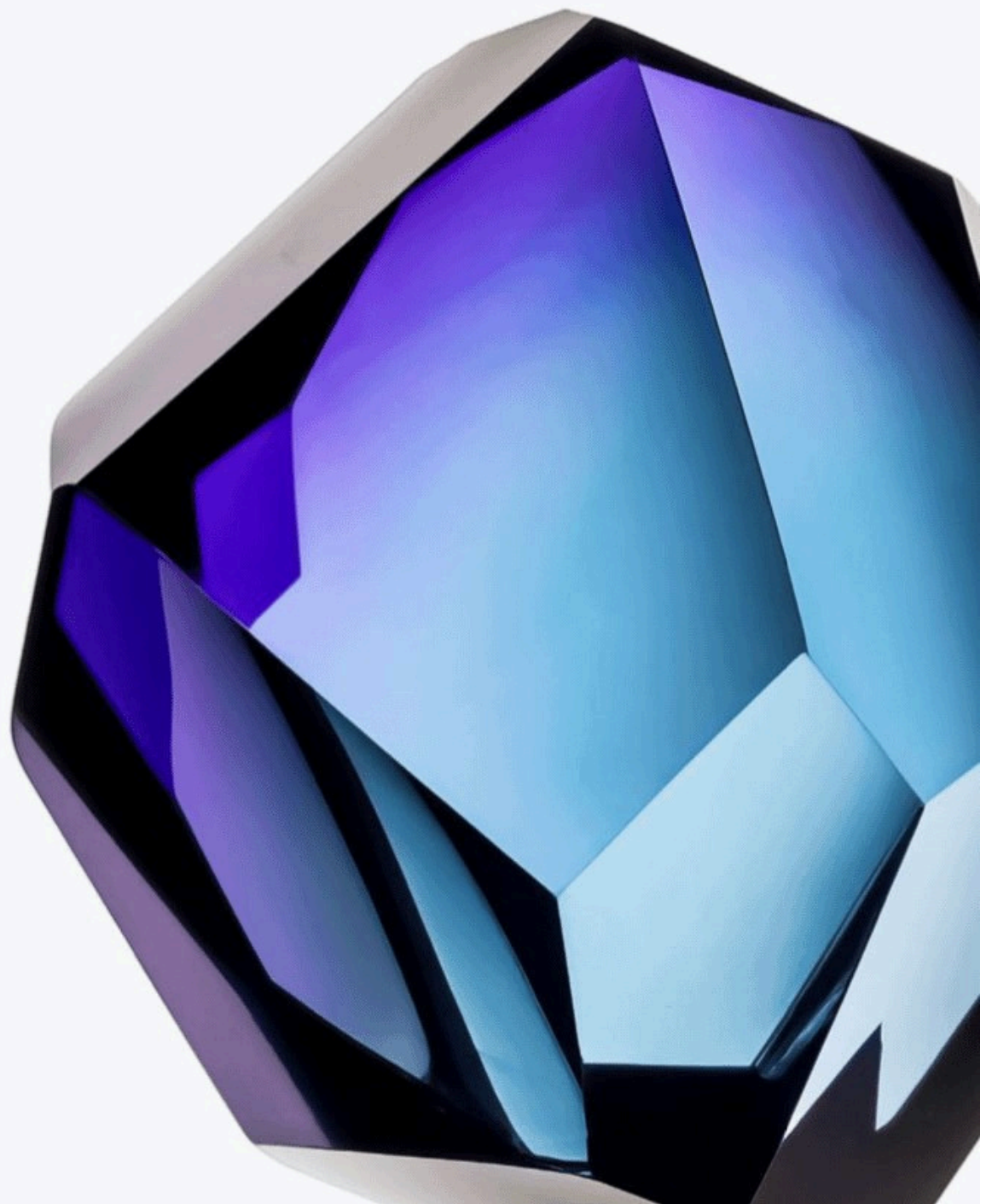



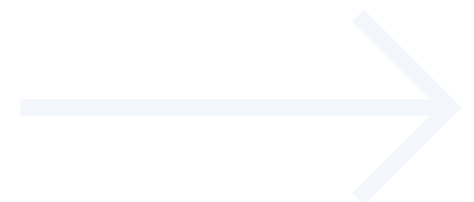
Table of contents

Introduction	4
What is an agent?	5
The model	6
The tools	7
The orchestration layer	7
Agents vs. models	8
Cognitive architectures: How agents operate	8
Tools: Our keys to the outside world	12
Extensions	13
Sample Extensions	15
Functions	18
Use cases	21
Function sample code	24
Data stores	27
Implementation and application	28
Tools recap	32
Enhancing model performance with targeted learning	33
Agent quick start with LangChain	35
Production applications with Vertex AI agents	38
Summary	40
Endnotes	42

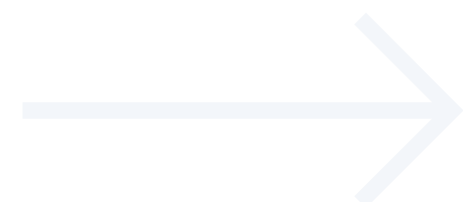




This combination of reasoning, logic, and access to external information that are all connected to a Generative AI model invokes the concept of an agent.

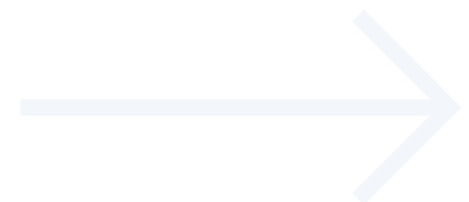


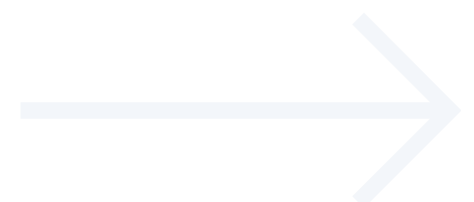
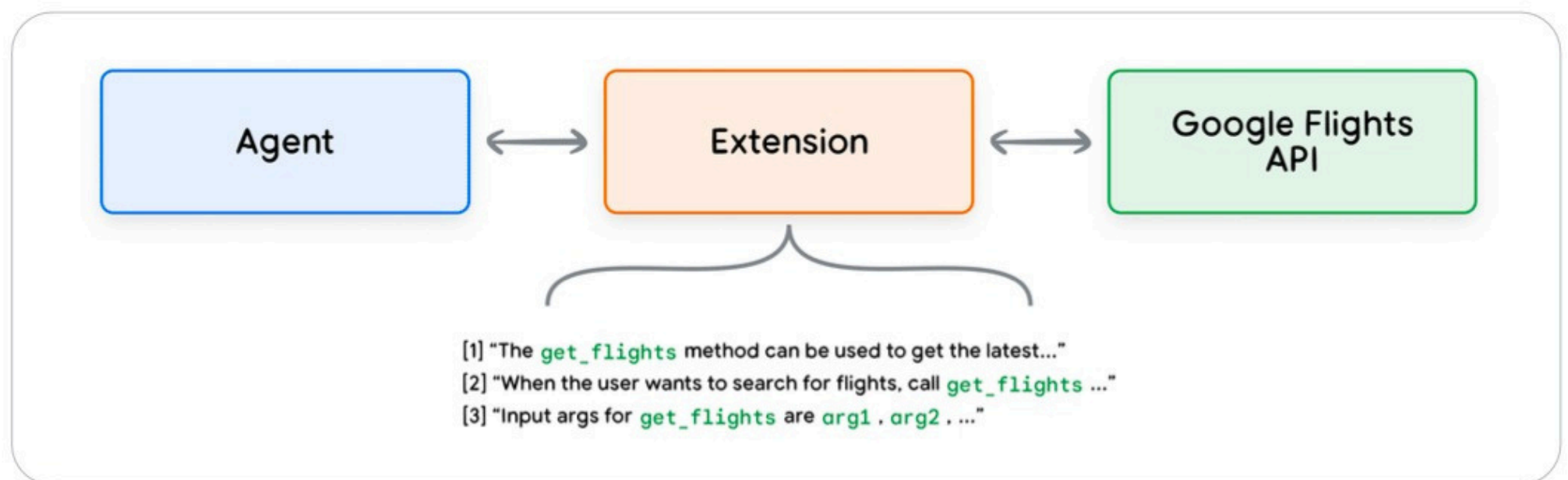
Models	Agents
Knowledge is limited to what is available in their training data.	Knowledge is extended through the connection with external systems via tools
Single inference / prediction based on the user query. Unless explicitly implemented for the model, there is no management of session history or continuous context. (i.e. chat history)	Managed session history (i.e. chat history) to allow for multi turn inference / prediction based on user queries and decisions made in the orchestration layer. In this context, a 'turn' is defined as an interaction between the interacting system and the agent. (i.e. 1 incoming event/ query and 1 agent response)
No native tool implementation.	Tools are natively implemented in agent architecture.
No native logic layer implemented. Users can form prompts as simple questions or use reasoning frameworks (CoT, ReAct, etc.) to form complex prompts to guide the model in prediction.	Native cognitive architecture that uses reasoning frameworks like CoT, ReAct, or other pre-built agent frameworks like LangChain.



The orchestration layer

The orchestration layer describes a cyclical process that governs how the agent takes in information, performs some internal reasoning, and uses that reasoning to inform its next action or decision. In general, this loop will continue until an agent has reached its goal or a stopping point. The complexity of the orchestration layer can vary greatly depending on the agent and task it's performing. Some loops can be simple calculations with decision rules, while others may contain chained logic, involve additional machine learning algorithms, or implement other probabilistic reasoning techniques. We'll discuss more about the detailed implementation of the agent orchestration layers in the cognitive architecture section.





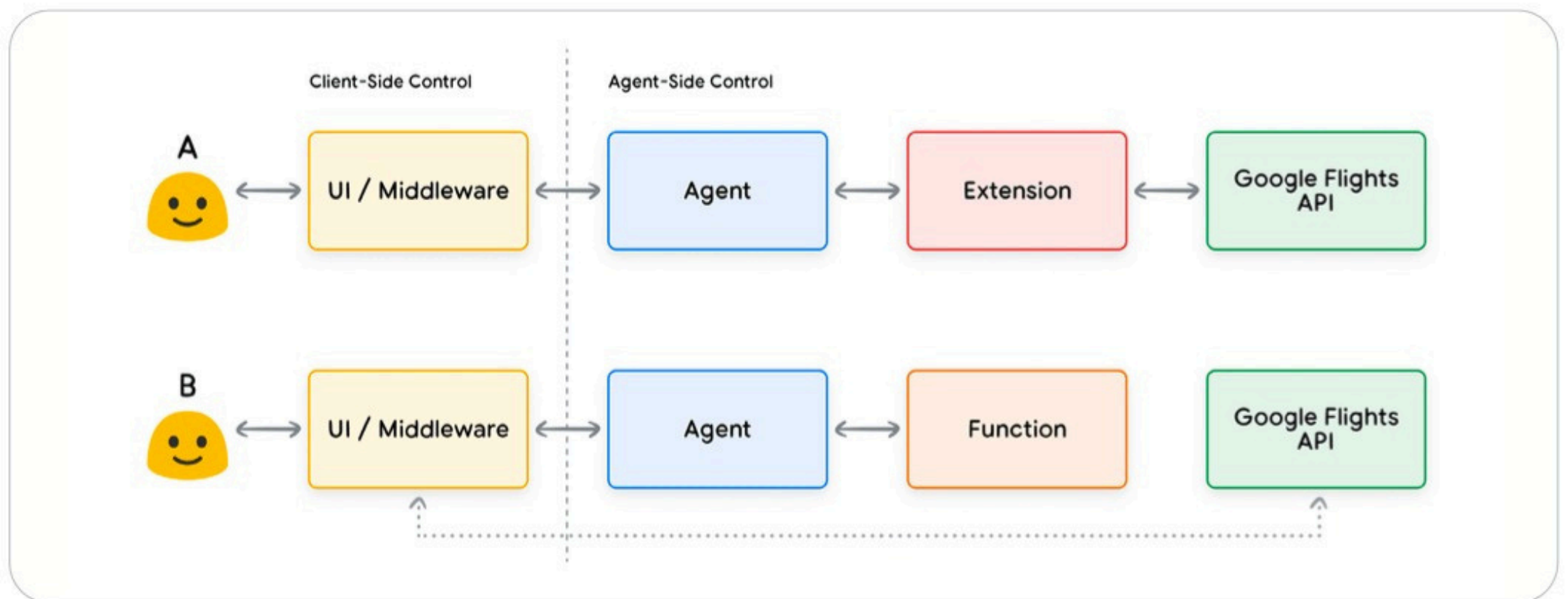
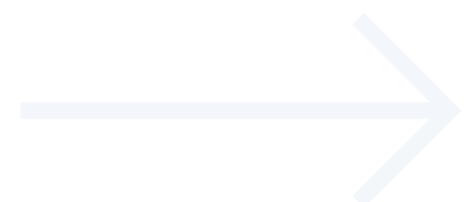
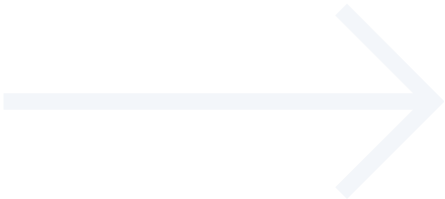
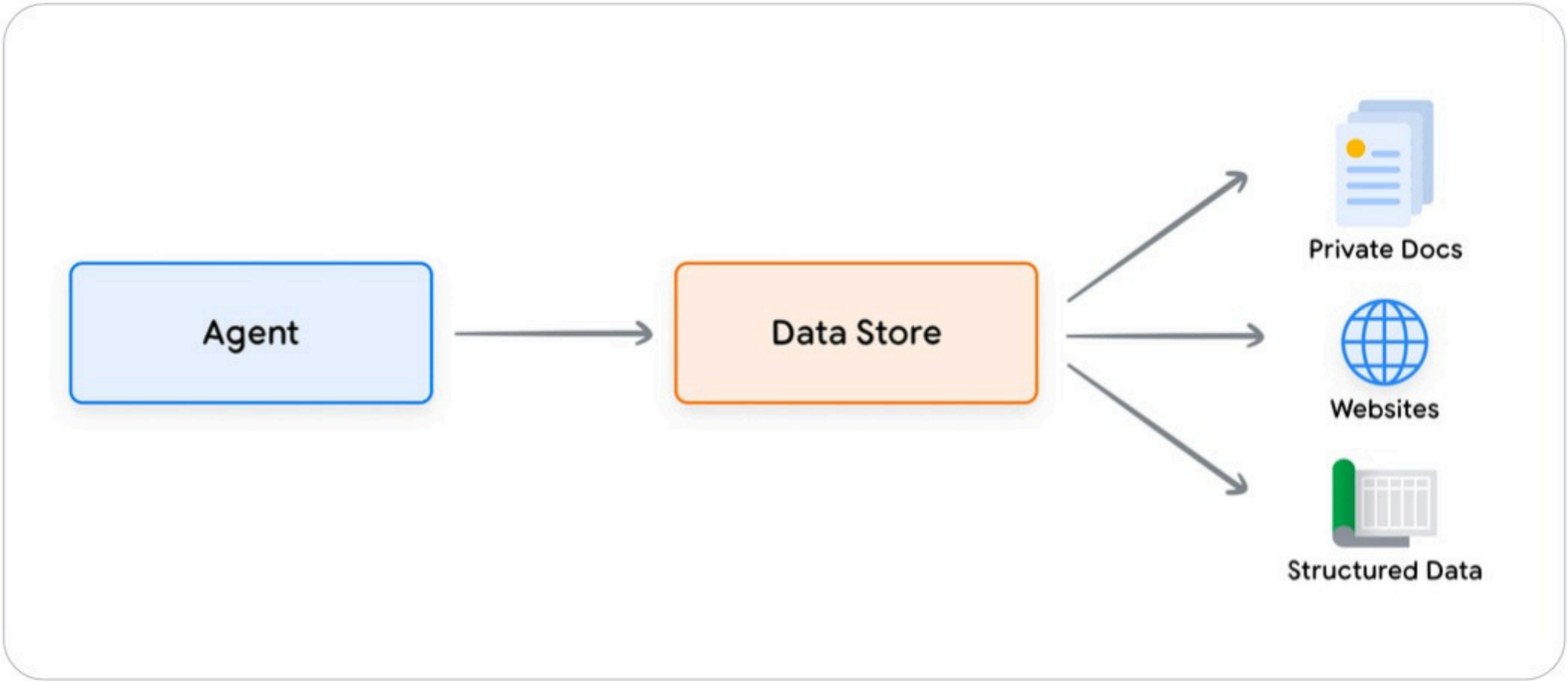


Figure 8. Delineating client vs. agent side control for extensions and function calling





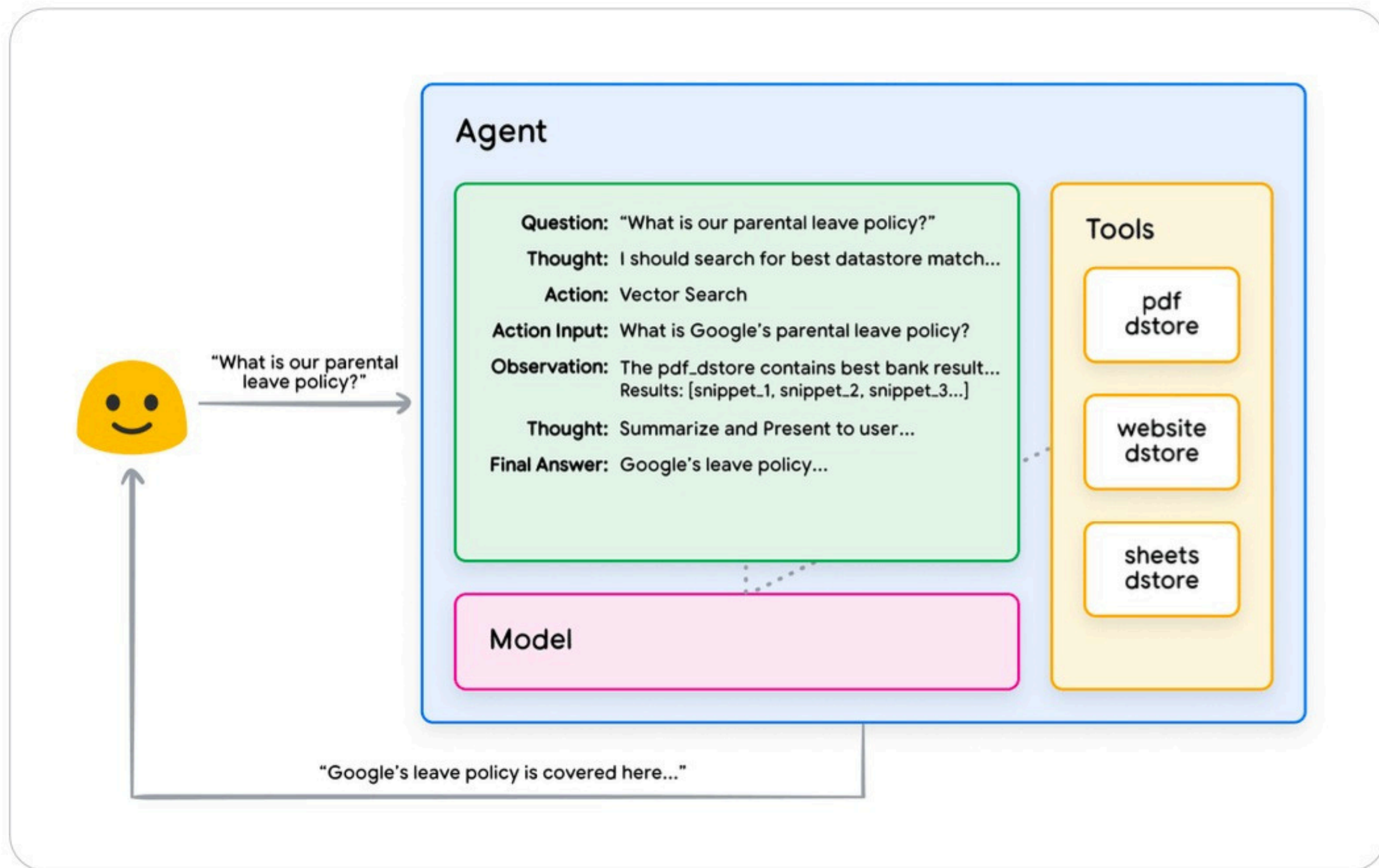


Figure 14. Sample RAG based application w/ ReAct reasoning/planning

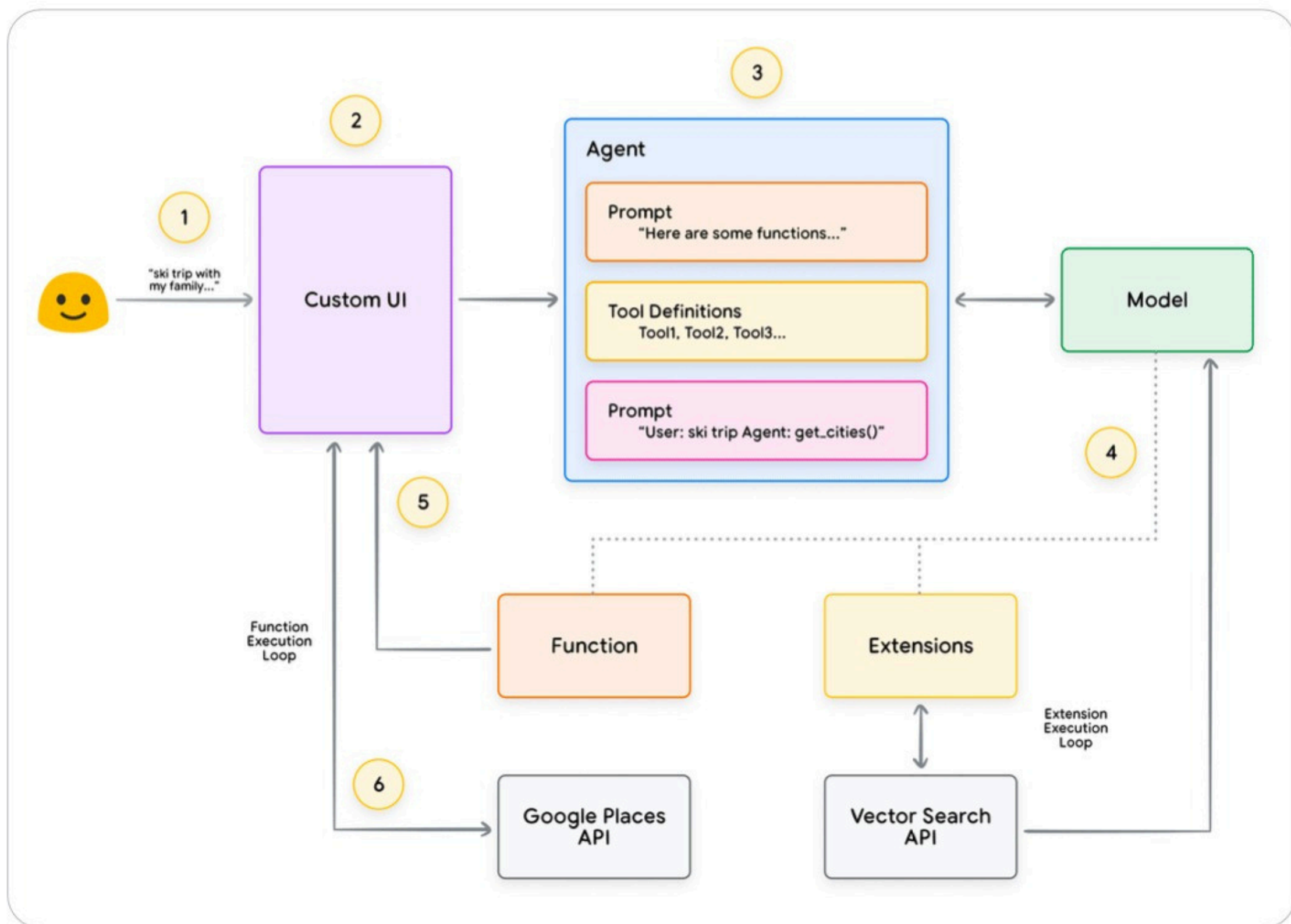


Figure 15. Sample end-to-end agent architecture built on Vertex AI platform

The End
Follow Naveed Sarwar