Carnegie Mellon University

Lecture 4: LLM Application Design (Cont’d)

SPRING 2025

MOHAMED FARAG

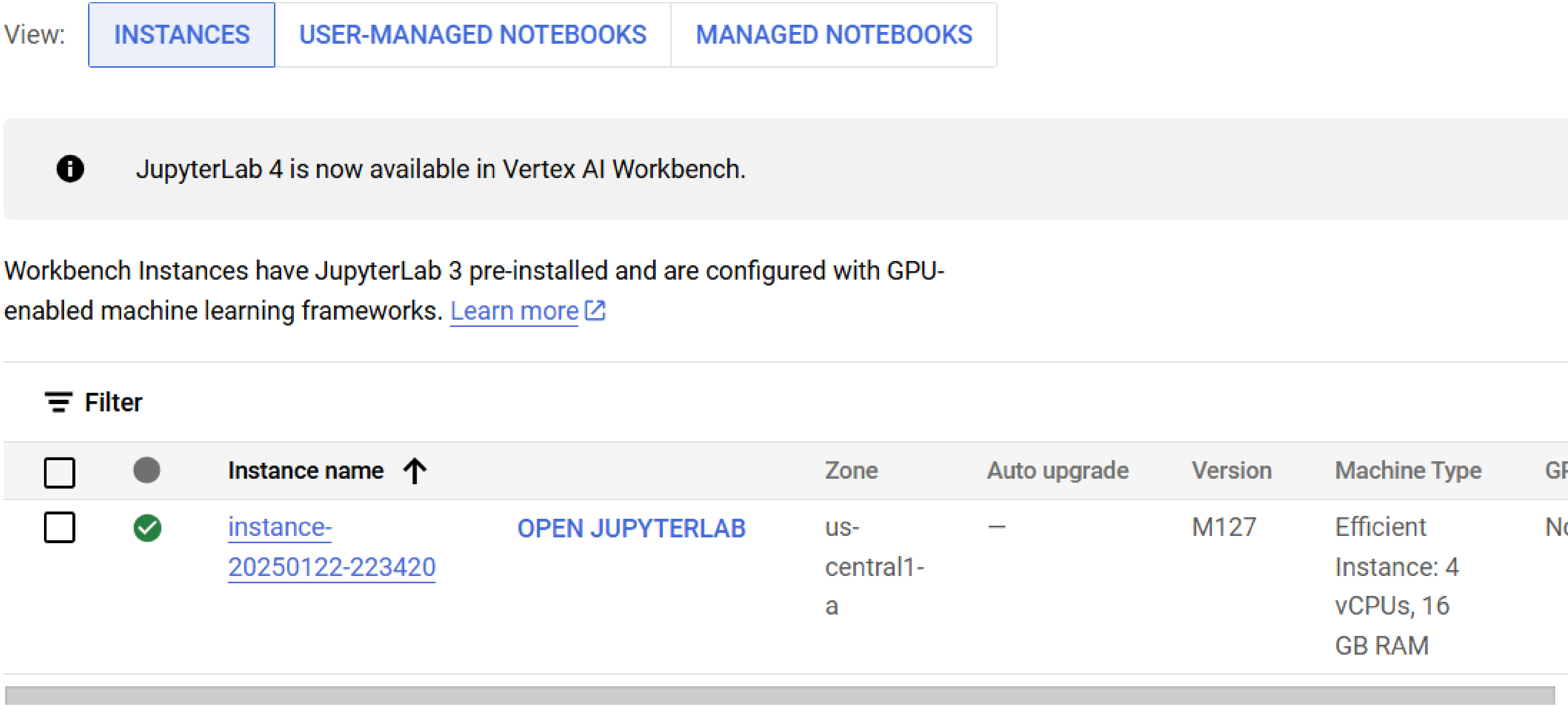
FARAG@CMU.EDU

Agenda

- Create Your Vertex AI Cloud Environment
- Generate Vertex AI Credentials
- FlowiseAI Components
- FlowiseAI Exercises

Create Your Vertex AI Cloud Environment

Hint: Enable Compute Engine and Vertex AI APIs

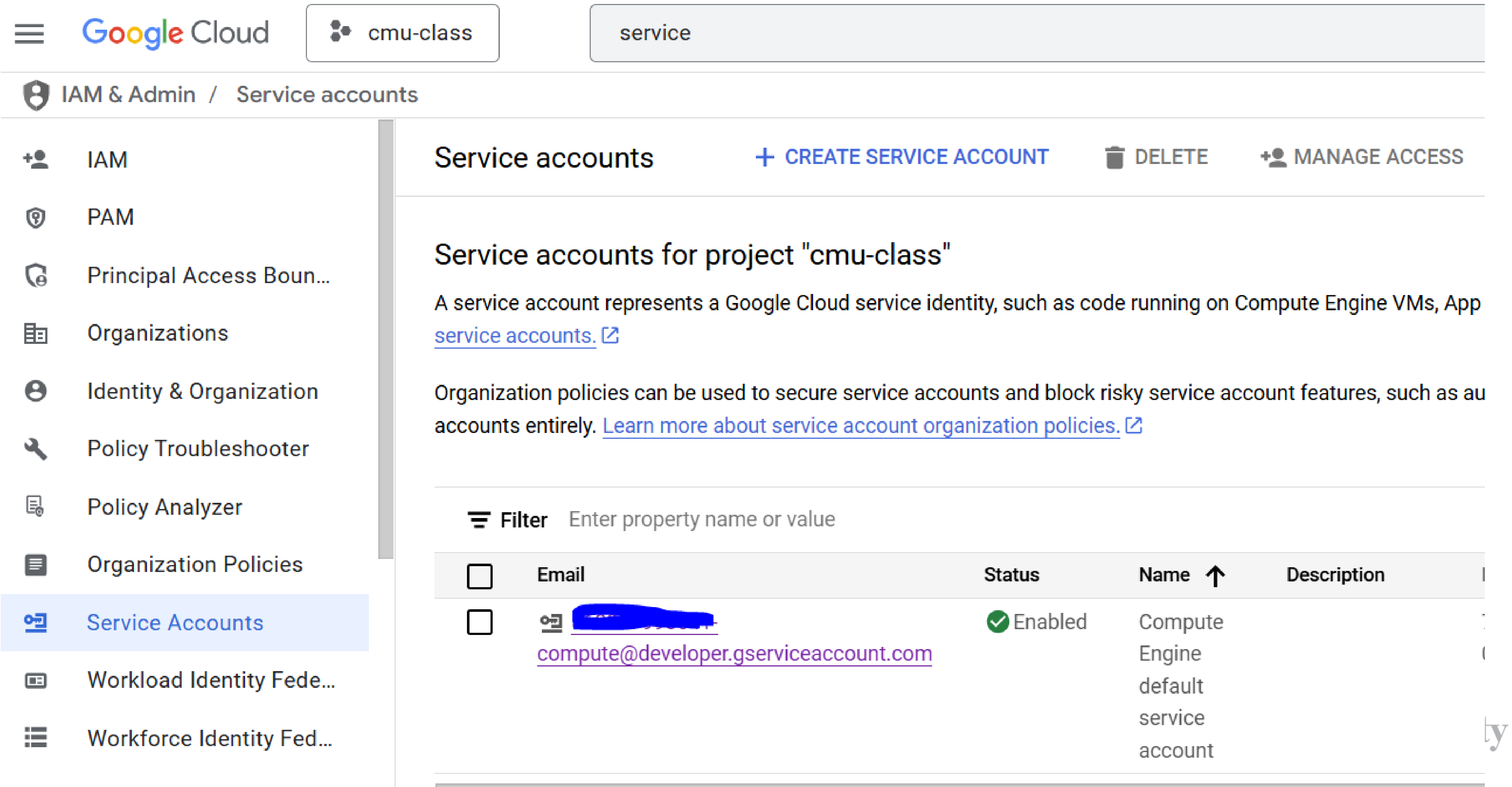


JupyterLab 4 is now available in Vertex AI Workbench.

Workbench Instances have JupyterLab 3 pre-installed and are configured with GPU-enabled machine learning frameworks. Learn more

□	●	Instance name ↑		Zone	Auto upgrade	Version	Machine Type	GP
□	✓	instance-20250122-223420	OPEN JUPYTERLAB	us-central1-a	—	M127	Efficient Instance: 4 vCPUs, 16 GB RAM	No

Generate Vertex API Credentials for FlowiseAI



Service accounts

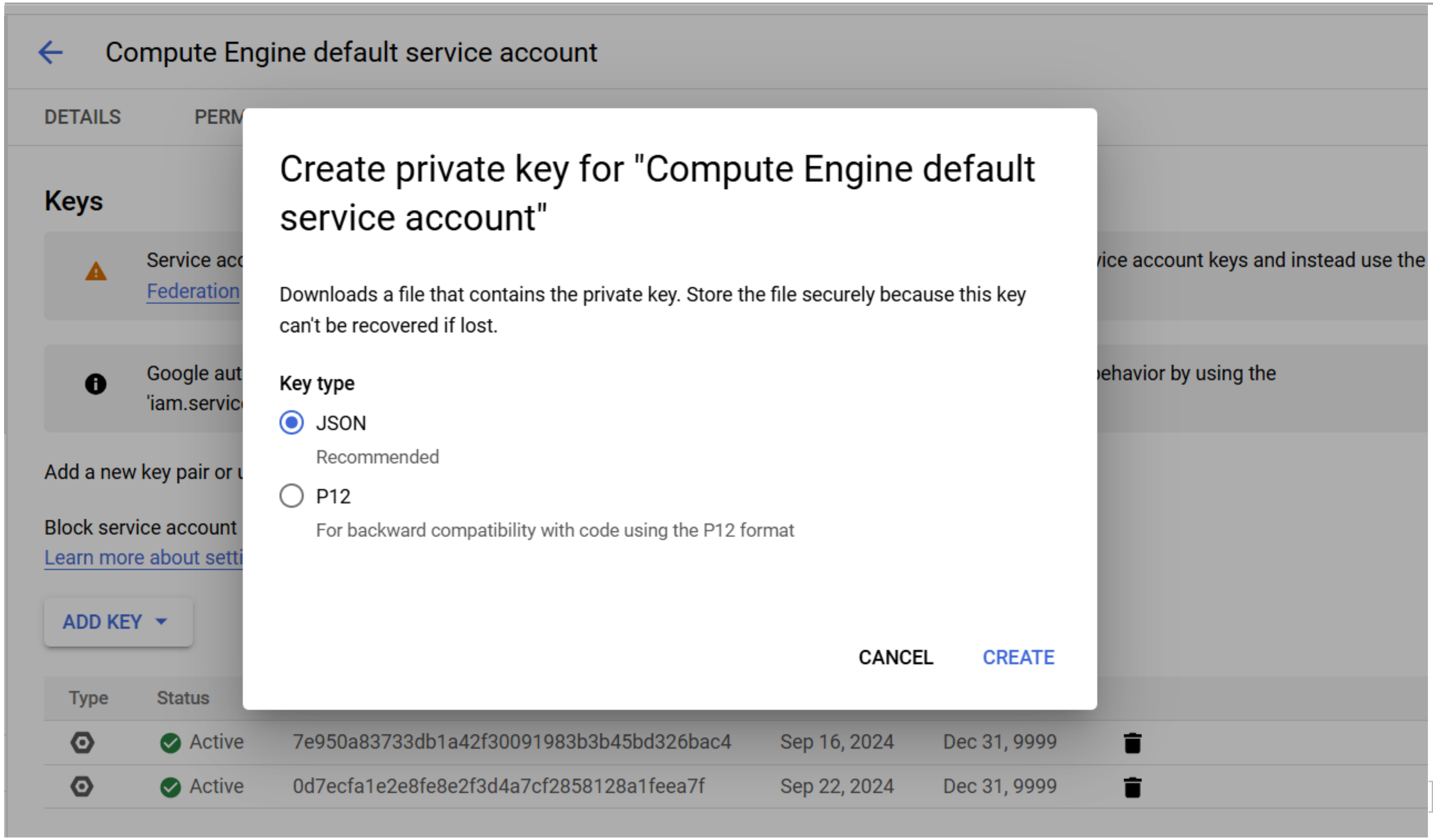
Service accounts for project "cmu-class"

A service account represents a Google Cloud service identity, such as code running on Compute Engine VMs, App
service accounts.

Organization policies can be used to secure service accounts and block risky service account features, such as au
accounts entirely. Learn more about service account organization policies.

□	Email	Status	Name ↑	Description
□	[redacted] compute@developer.gserviceaccount.com	✓ Enabled	Compute Engine default service account	

Create new JSON key and paste JSON file contents into your FlowiseAI credentials



Compute Engine default service account

Keys

Service acc… ice account keys and instead use the
Federation

Google aut… ehavior by using the
'iam.servic

Add a new key pair or u

Block service account
Learn more about setti

Create private key for "Compute Engine default service account"

Downloads a file that contains the private key. Store the file securely because this key can't be recovered if lost.

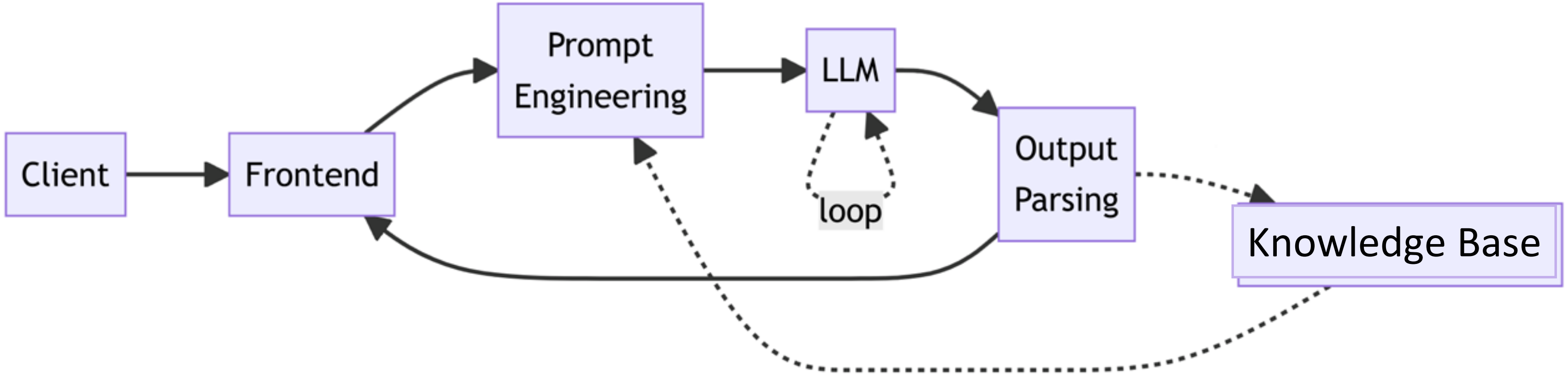
Key type

● JSON
Recommended

○ P12
For backward compatibility with code using the P12 format

Type	Status				
	Active	7e950a83733db1a42f30091983b3b45bd326bac4	Sep 16, 2024	Dec 31, 9999	
	Active	0d7ecfa1e2e8fe8e2f3d4a7cf2858128a1feea7f	Sep 22, 2024	Dec 31, 9999	

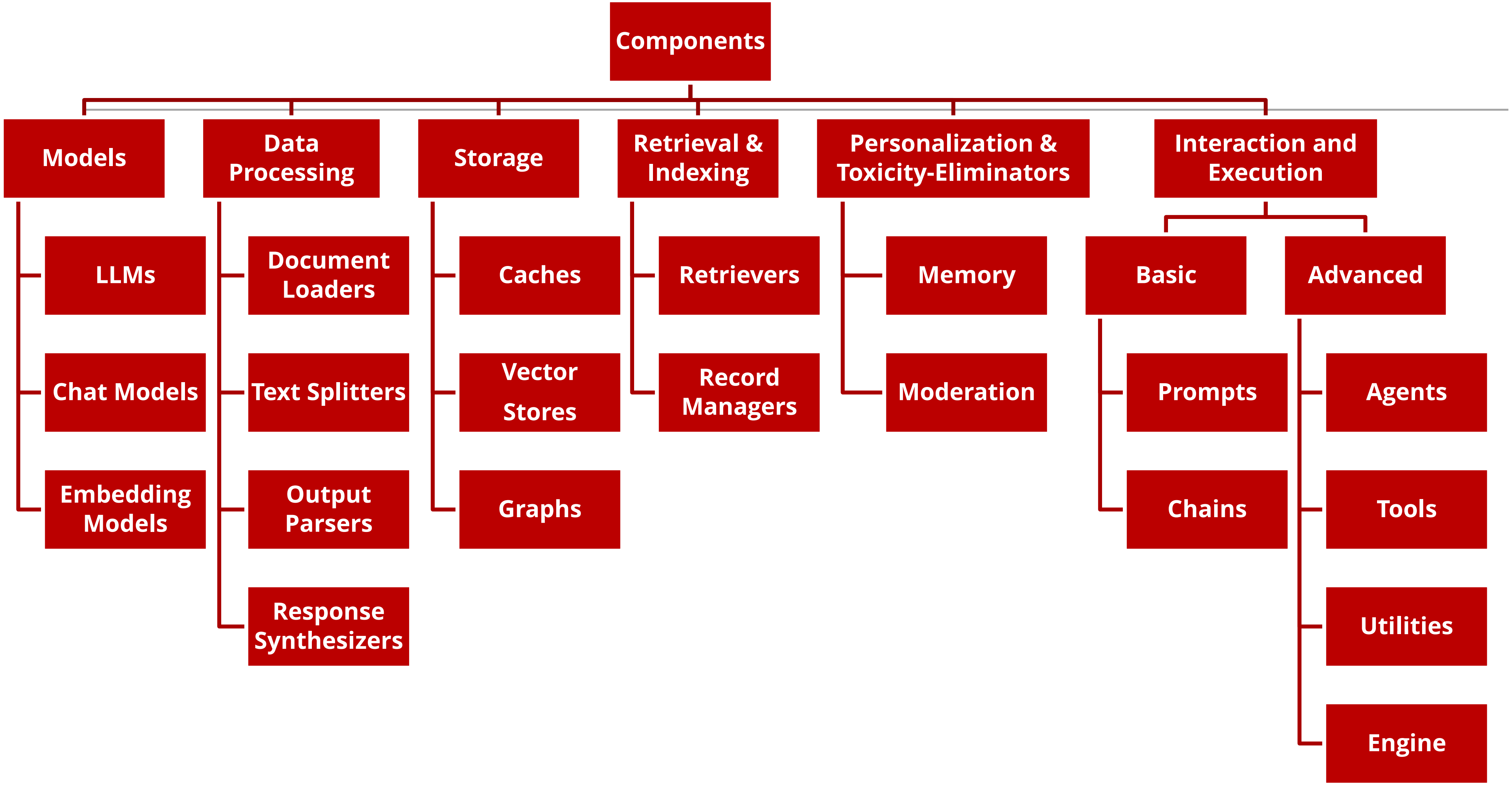
LLM Applications – General Idea



What if we would like to:

- Leverage external documents to enhance the capabilities of the LLM.
- Incorporate previous conversations into the LLM's responses.
- Integrate multiple prompts or functionalities for a cohesive experience
- Use a custom functionality that is not offered out of the box.

FlowiseAI Components



- Components
  - Models
    - LLMs
    - Chat Models
    - Embedding Models
  - Data Processing
    - Document Loaders
    - Text Splitters
    - Output Parsers
    - Response Synthesizers
  - Storage
    - Caches
    - Vector Stores
    - Graphs
  - Retrieval & Indexing
    - Retrievers
    - Record Managers
  - Personalization & Toxicity-Eliminators
    - Memory
    - Moderation
  - Interaction and Execution
    - Basic
      - Prompts
      - Chains
    - Advanced
      - Agents
      - Tools
      - Utilities
      - Engine

FlowiseAI Components (Cont’d)

Category	Nodes	Purpose
Models	LLMs	Large Language Models for natural language understanding and generation
	Chat Models	Specialized models for conversations
	Embedding Models	Transforming data into vector space for machine learning
Data Processing	Document Loaders	Nodes for ingesting and parsing documents
	Text Splitters	Divide text into smaller chunks for processing
	Output Parsers	Convert model outputs into structured formats
	Response Synthesizer	Combines outputs into cohesive and structured responses (from LlamaIndex)

FlowiseAI Components (Cont'd)

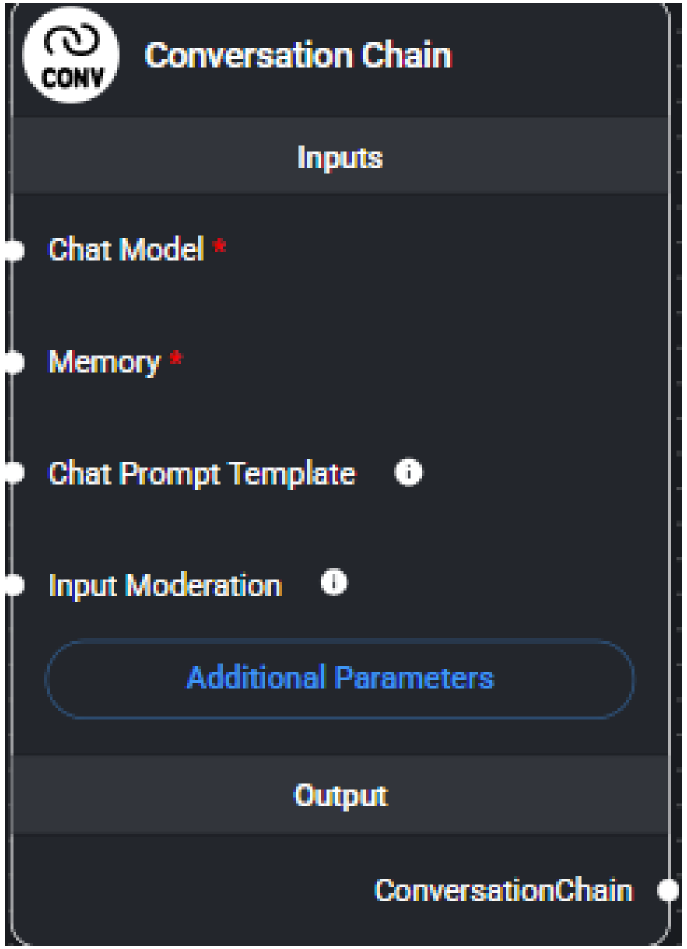
Category	Nodes	Purpose
Storage	Caches	Temporary storage to speed up recurring processes
	Vector Stores	Store and retrieve vector representations of data efficiently
	Graphs	Work with graph-based structures for complex relationships
Retrieval & Indexing	Retrievers	Fetch relevant data from storage for a query
	Record Managers	Organize and manage saved records in a database
Personalization & Toxicity-Eliminators	Memory	Retains context and past interactions
	Moderation	Monitor and restricts inappropriate content

FlowiseAI Components - Chains

Chains are leveraged to assemble modular components into versatile and reusable pipelines. Chains have different types including:

- Conversation Chain: Uses memory to keep track of previous interactions
- Retrieval QA Chain: Retrieves relevant information from a single knowledge base
- Conversational Retrieval QA Chain: Retrieves relevant information from a single knowledge base and optionally uses memory to keep track of previous interactions as well.
- Multi Retrieval QA Chain: Combines multiple retrievers.

Chain Example – Conversation Chain



Conversation Chain

Inputs

Chat Model *

Memory *

Chat Prompt Template

Input Moderation

Output

ConversationChain

In your opinion,
why would you use a Chain?

FlowiseAI Components - Agents

Agents are autonomous software entities designed to perform actions and execute tasks.

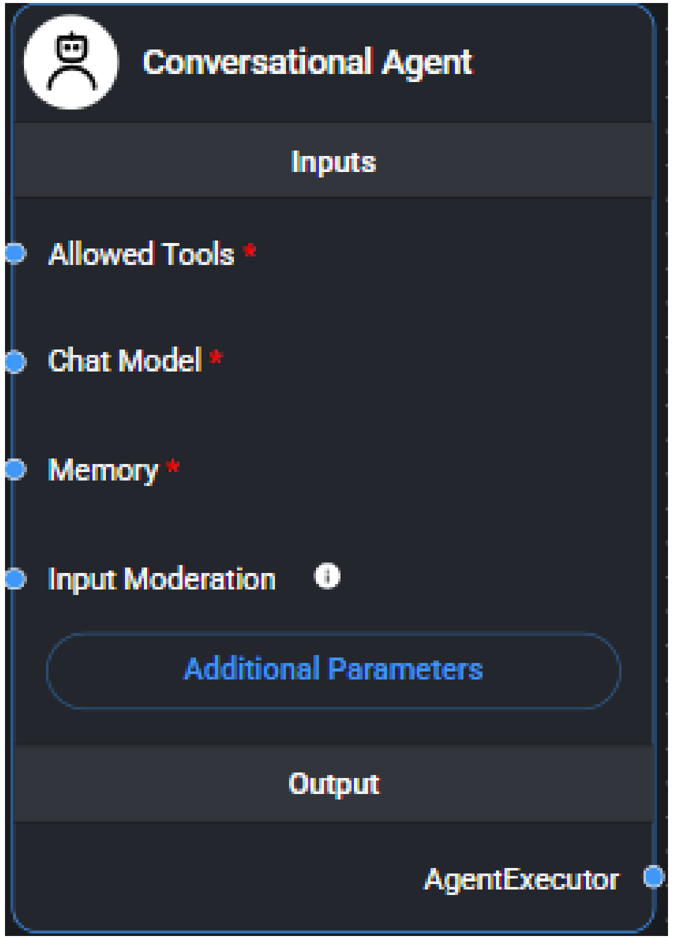
- What are the main differences between agents and chains?

Agents – Cont'd

Agents are autonomous software entities designed to perform actions and execute tasks.

- What are the main differences between agents and chains?
  - Agents orchestrate chains while chains orchestrate lower-level modules.
  - Agents can take actions and perform tasks.

Agent Example

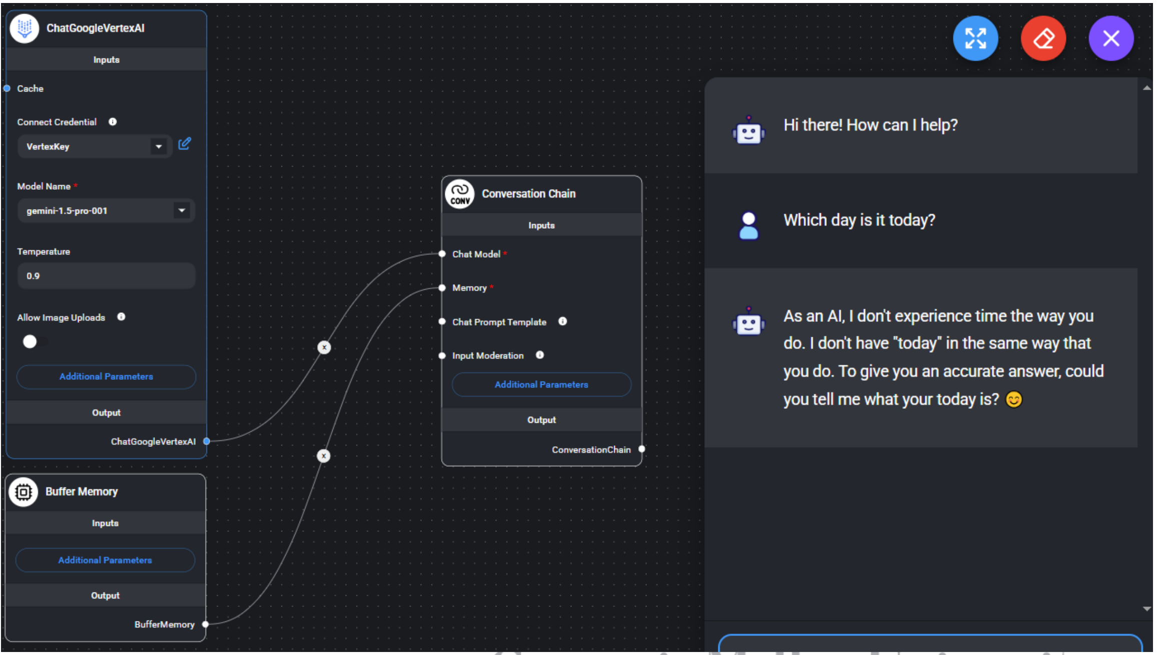


FlowiseAI Components - Tools

- Tools offer modular interfaces that enable agents to connect with external services such as databases and APIs.
- Toolkits organize tools that utilize shared resources.
- Example tools include Web Browser, Calculator, etc.
- You can create your custom tools as well.

Example

- We need to a tool to find today’s date or weather since LLMs can’t find this information on their own.
- Suggestions: Use https://open-meteo.com/ OR https://openweathermap.org to create your custom tool or agent.



ChatGoogleVertexAI

Inputs

Cache

Connect Credential: VertexKey

Model Name *: gemini-1.5-pro-001

Temperature: 0.9

Allow Image Uploads

Additional Parameters

Output

ChatGoogleVertexAI

Buffer Memory

Inputs

Additional Parameters

Output

BufferMemory

Conversation Chain

Inputs

Chat Model *

Memory *

Chat Prompt Template

Input Moderation

Additional Parameters

Output

ConversationChain

Hi there! How can I help?

Which day is it today?

As an AI, I don't experience time the way you do. I don't have "today" in the same way that you do. To give you an accurate answer, could you tell me what your today is? 😊

General LLM Application Design Guidelines

- Document Loaders → Text Splitters → Embeddings/Vector Stores
- Chat Models/LLMs → Output Parsers
- Prompts → LLM/Chat Model Chains
- Tools → Agents
- Moderation → Models, Chains, Agents, etc.

Exercises

- Perform Q&A on PDF file.
- Build Custom Tool for retrieving weather by Lat/Long

Readings

- Create your custom Tool in Flowise AI: https://docs.flowiseai.com/integrations/langchain/tools/custom-tool
  - Important reading for the project and HW-1.