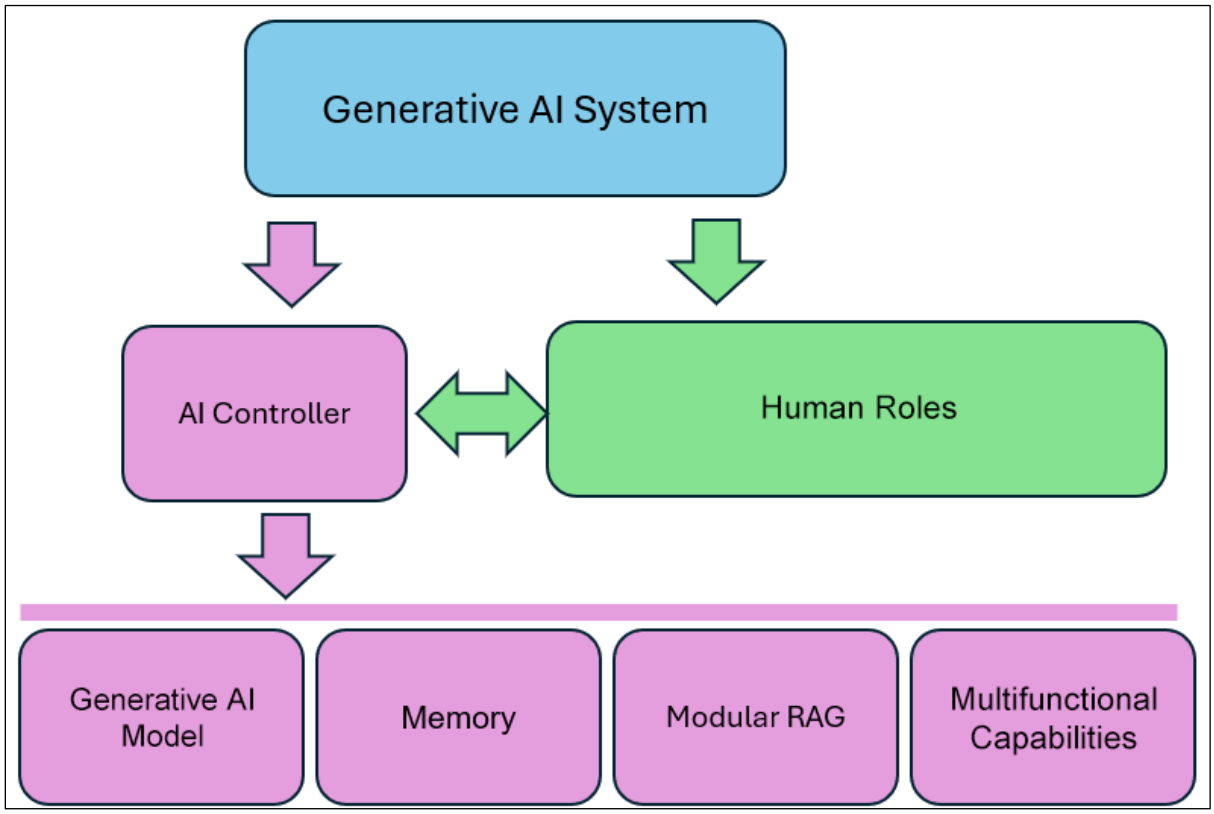
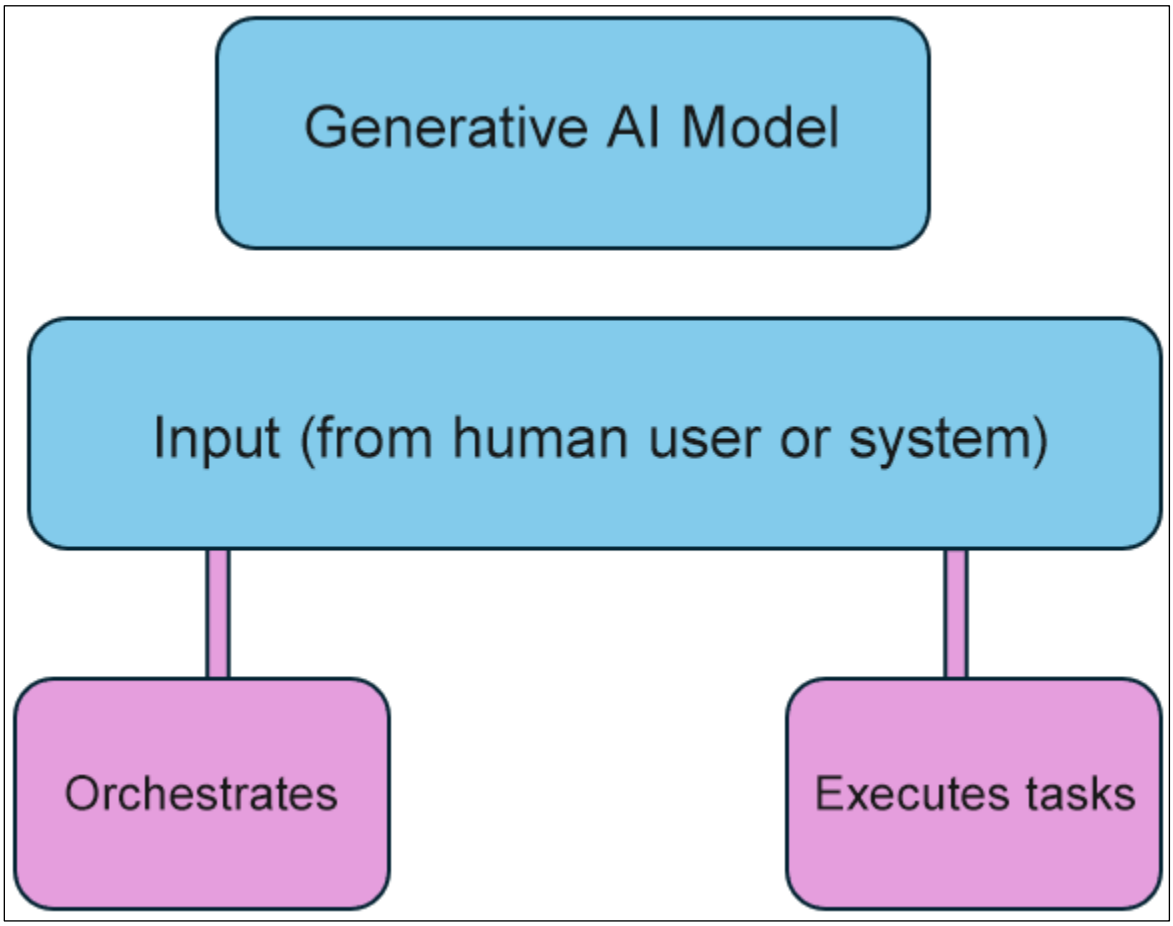
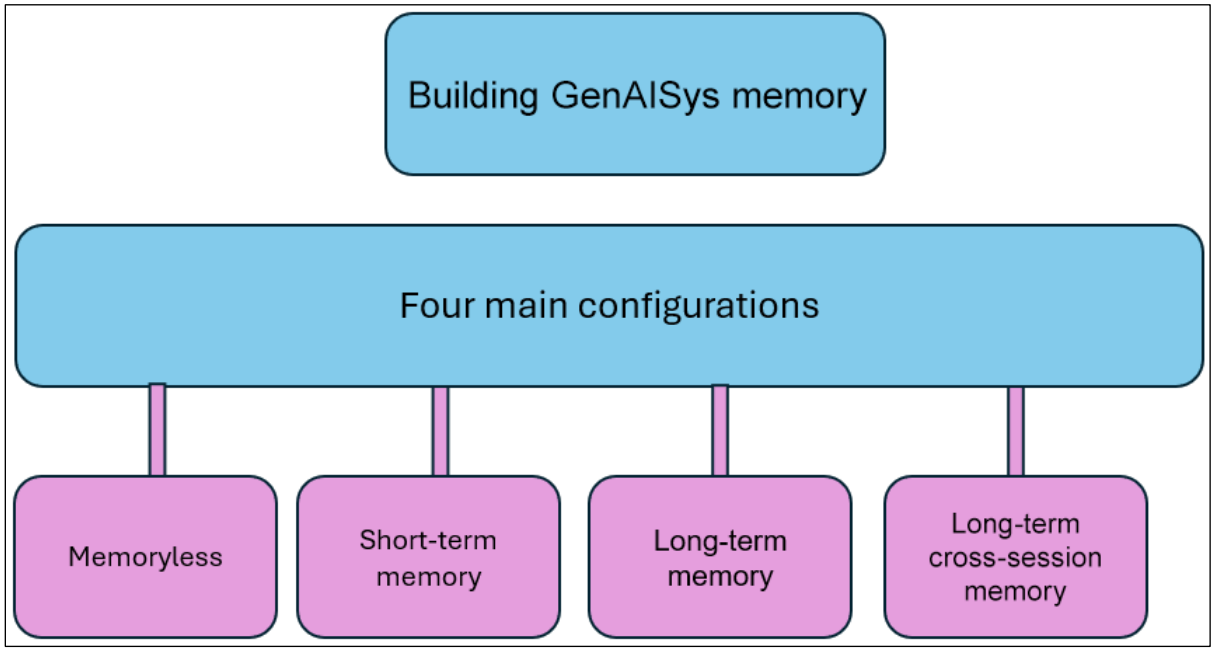
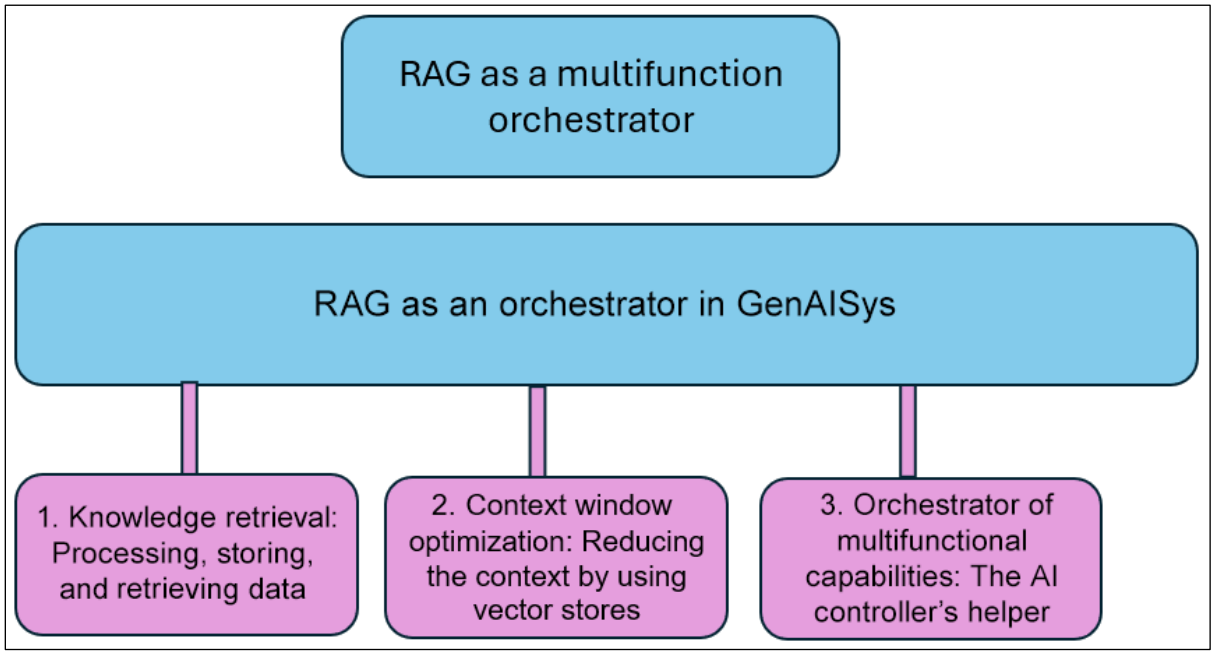
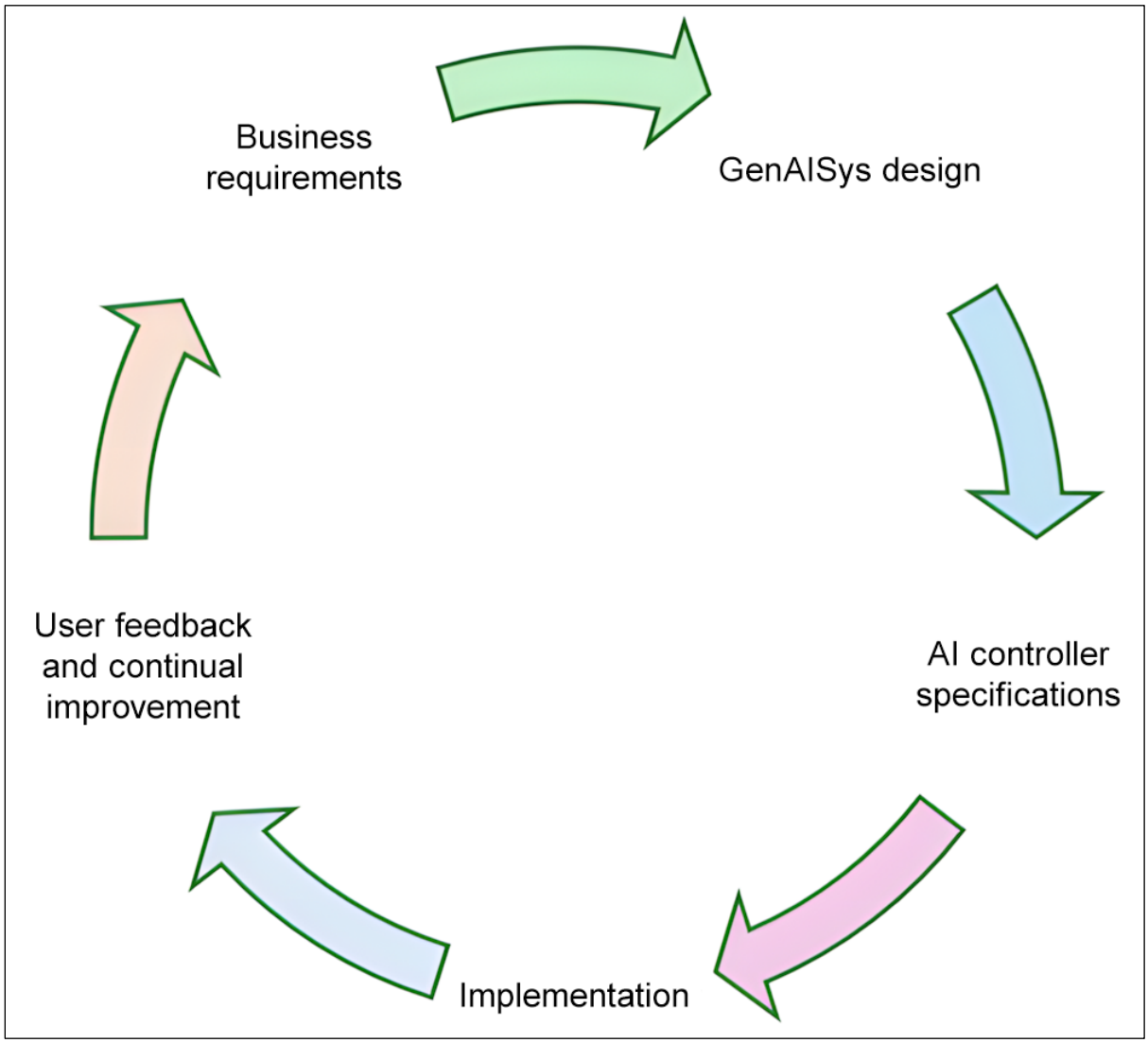
Chapter 1: Defining a Business-Ready Generative AI System



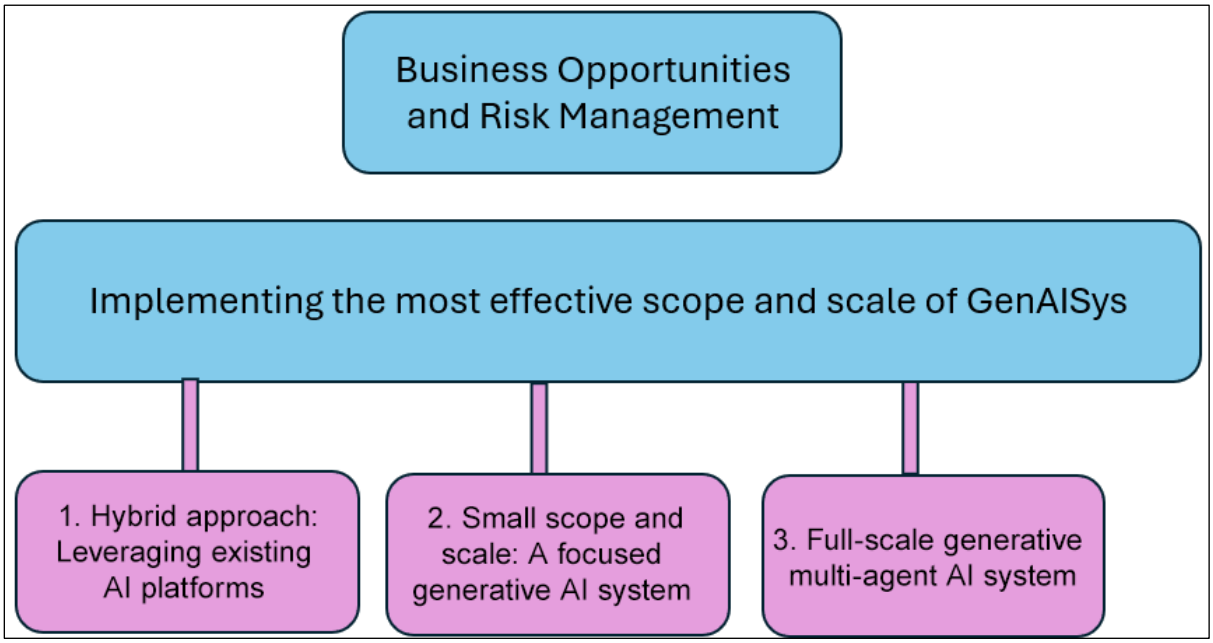

Generative AI Model
Input (from human user or system)
Orchestrates
Executes tasks


Building GenAISys memory
Four main configurations
Memoryless
Short-term memory
Long-term memory
Long-term cross-session memory


RAG as a multifunction orchestrator
RAG as an orchestrator in GenAISys
1. Knowledge retrieval: Processing, storing, and retrieving data
2. Context window optimization: Reducing the context by using vector stores
3. Orchestrator of multifunctional capabilities: The AI controller's helper

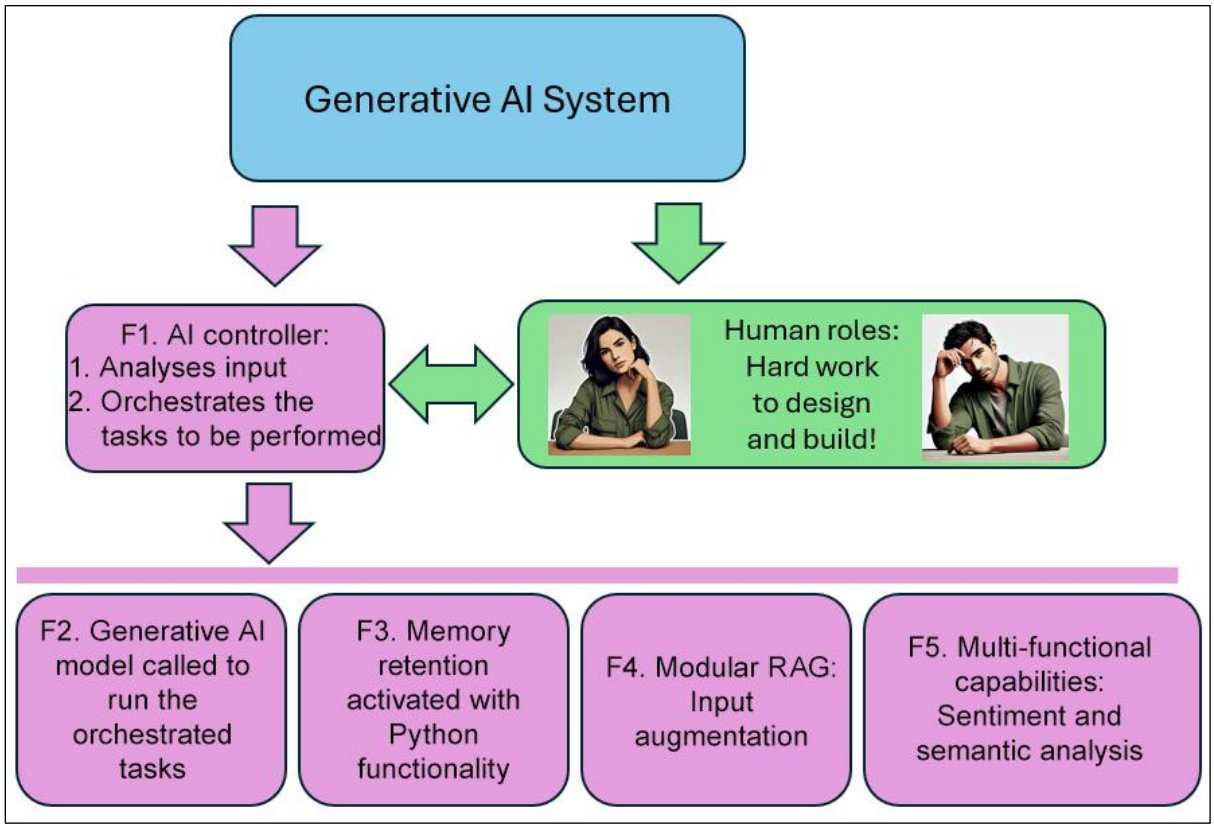

Business requirements
GenAISys design
AI controller specifications
Implementation
User feedback and continual improvement

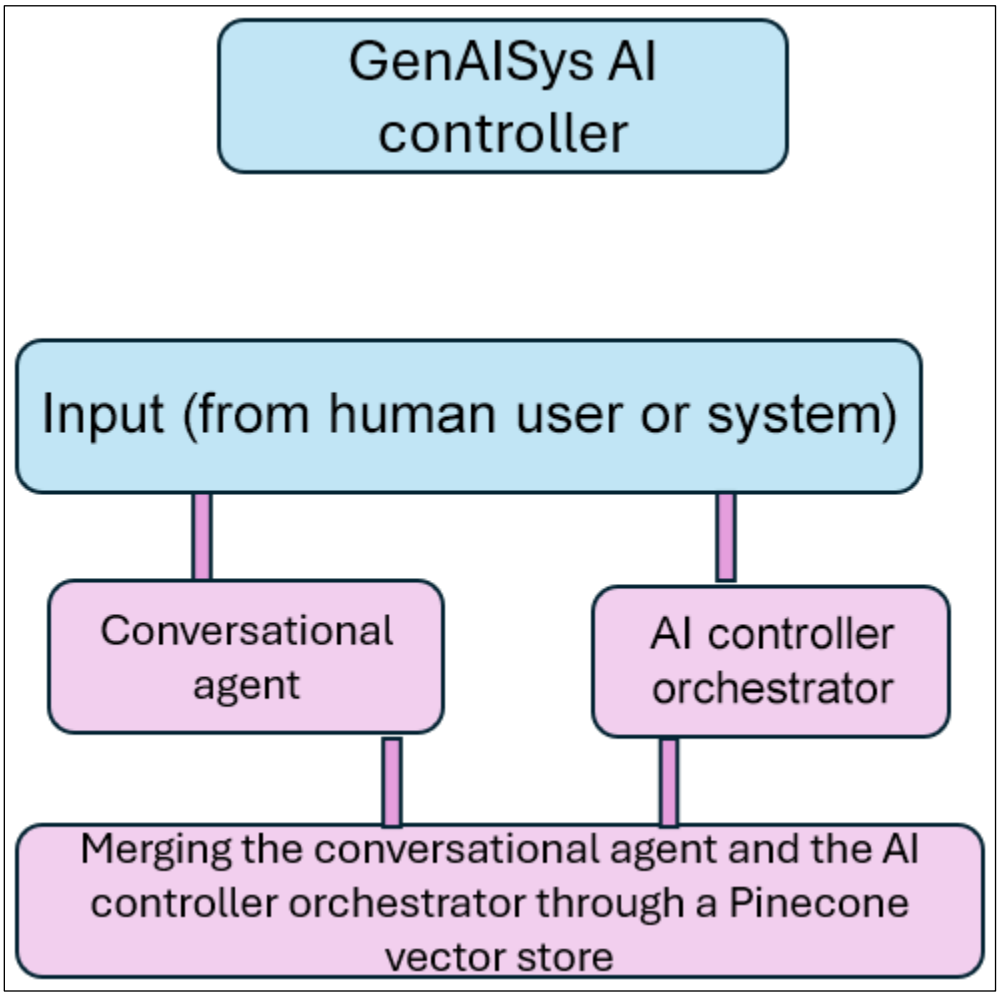
Role	Model	Controller	Pipelines	UI/UX	Security	DevOps	Prompts	QA
MLE	R/A	C	C	I	I	I	R/A	C
DE	C	C	R/A	I	C	I	I	C
BE	I	R/A	R	C	I	R	I	C
FE	I	I	I	R/A	I	I	I	C
UX	I	I	I	A/R	I	I	I	C
CE	C	C	C	I	C	R/A	I	C
SE	I	I	C	I	R/A	I	I	C
DPO	I	I	C	I	R	I	I	I
QAE	C	C	C	C	C	C	C	R/A

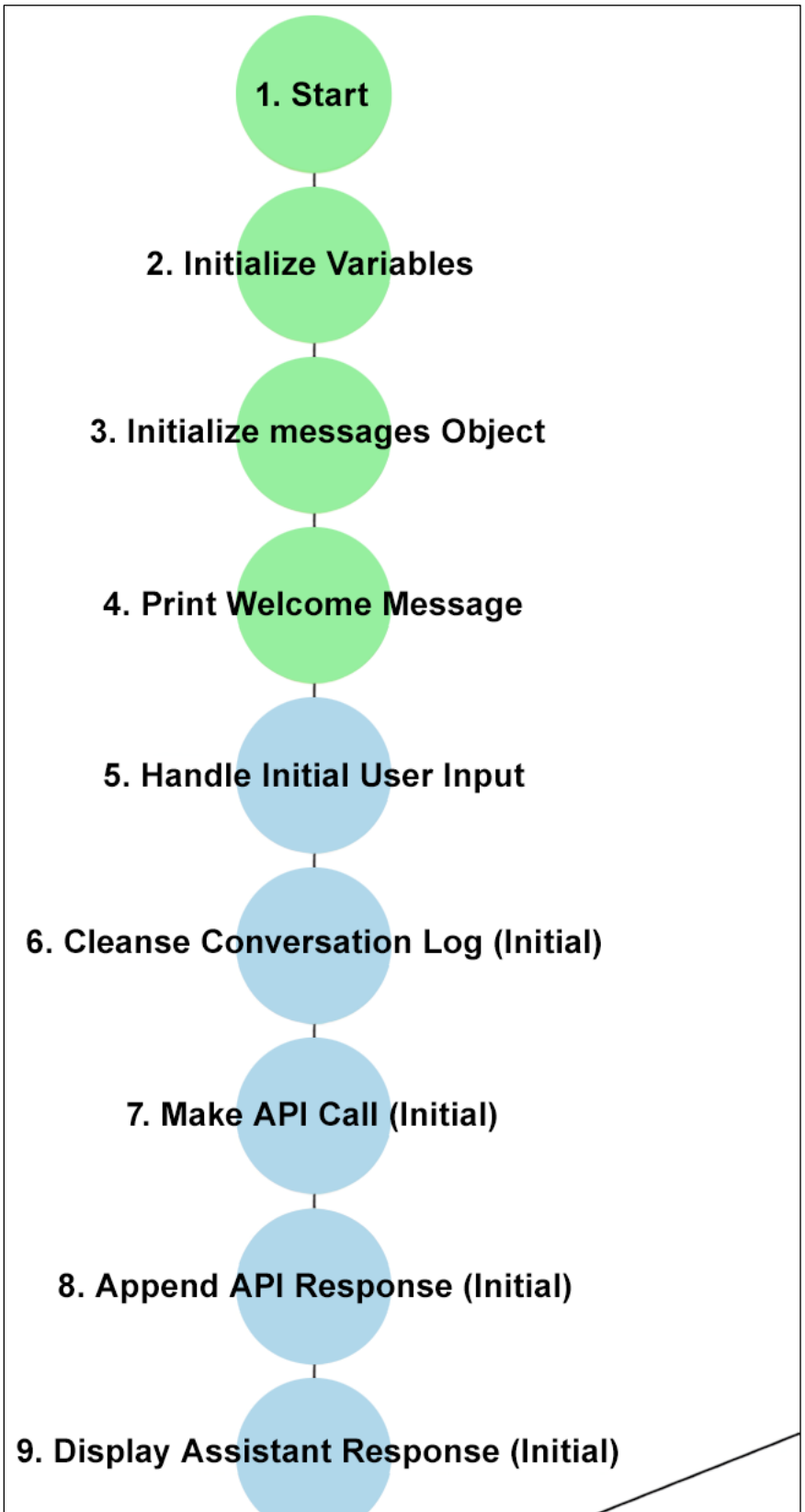


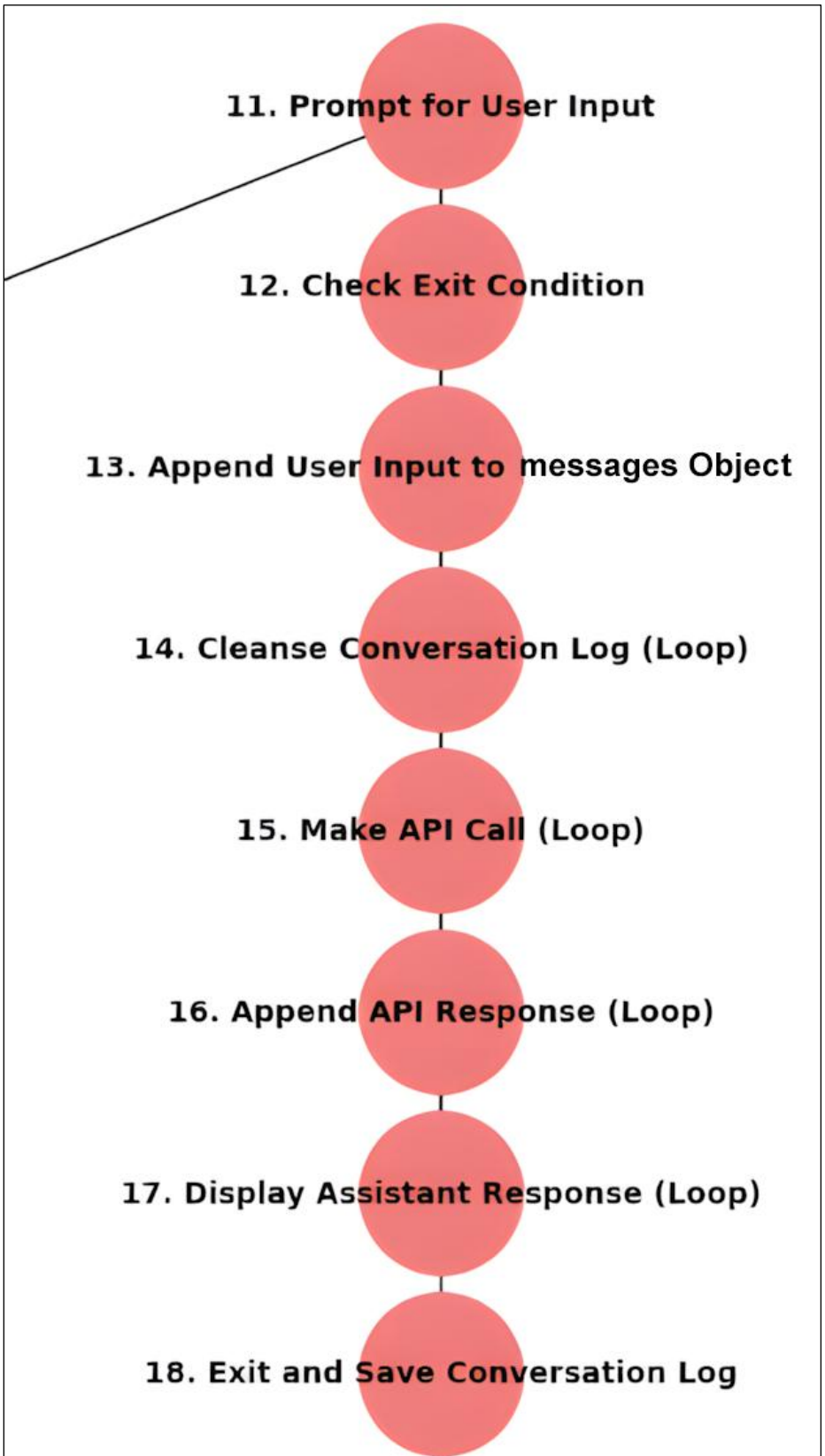

Notebook access
Name
Value
Actions
API_KEY
Add new secret

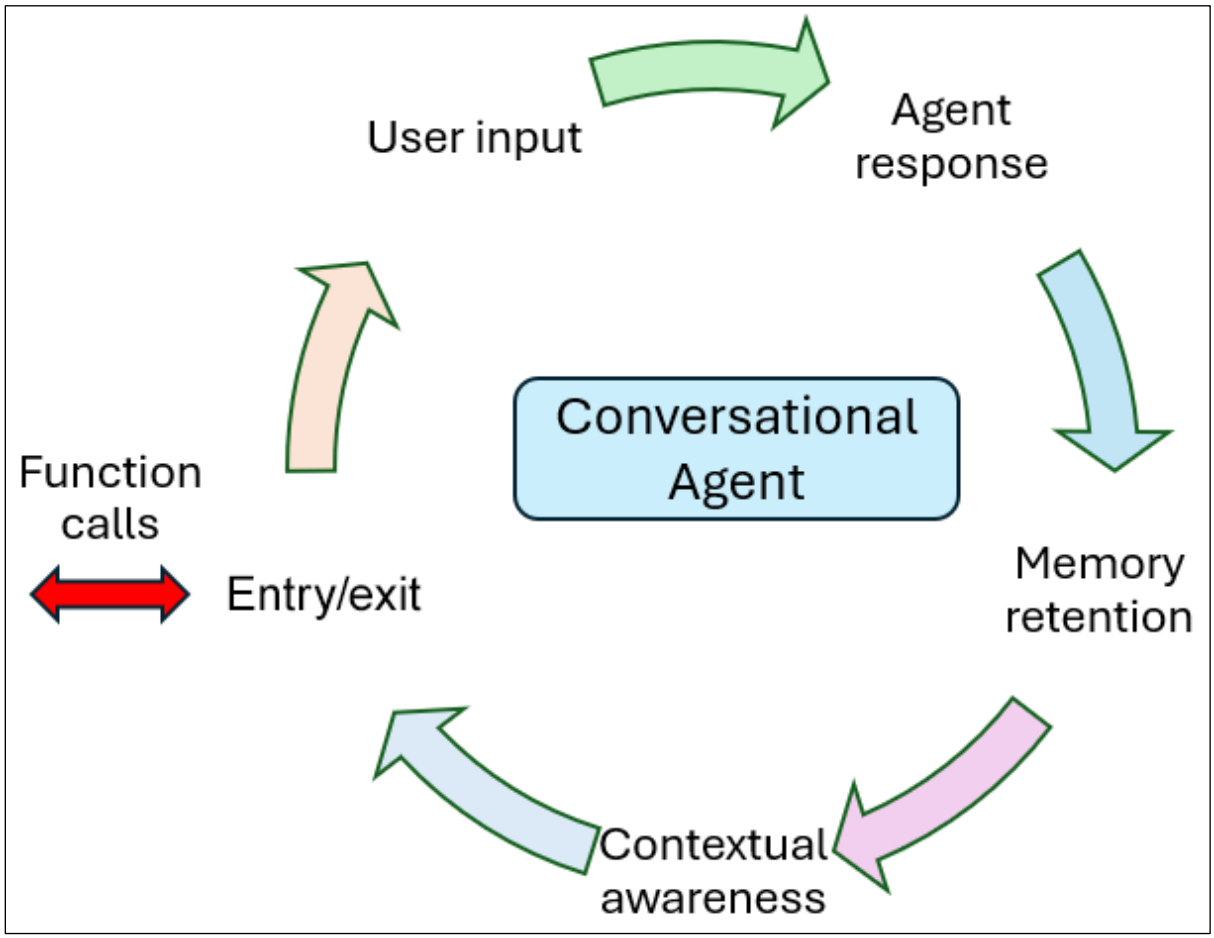
Chapter 2: Building the Generative AI Controller

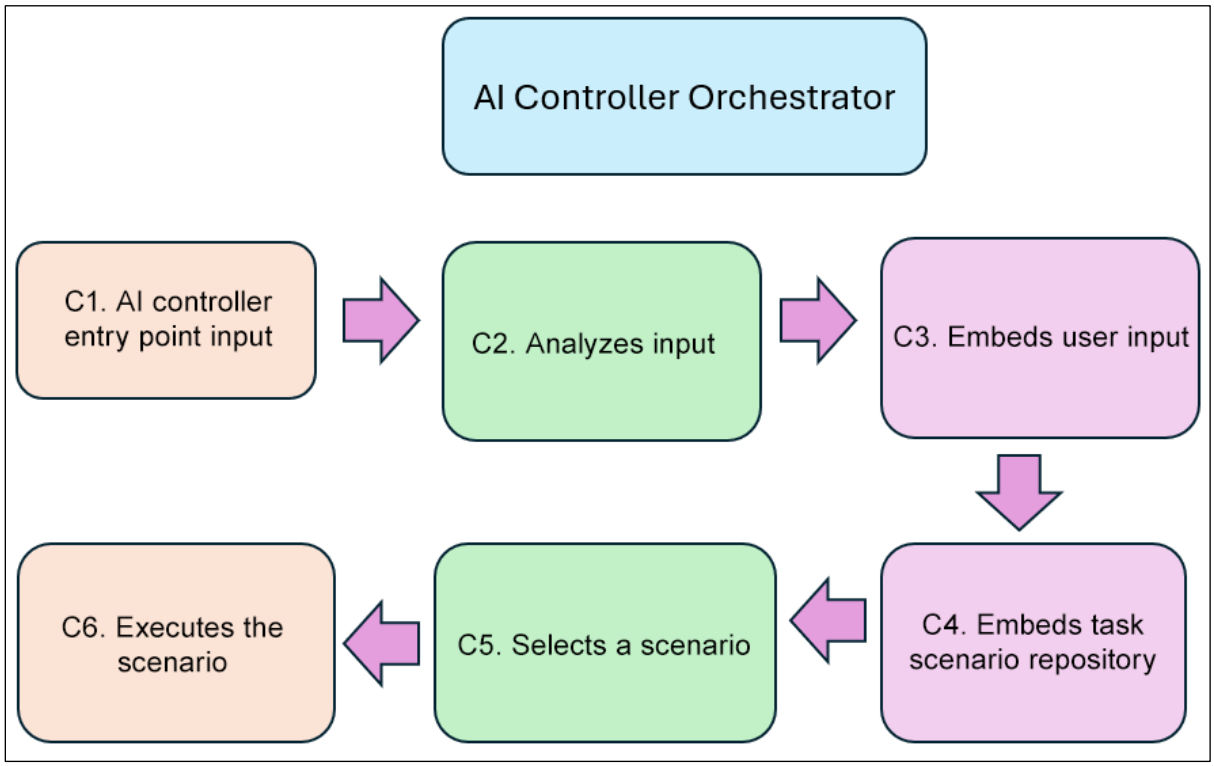


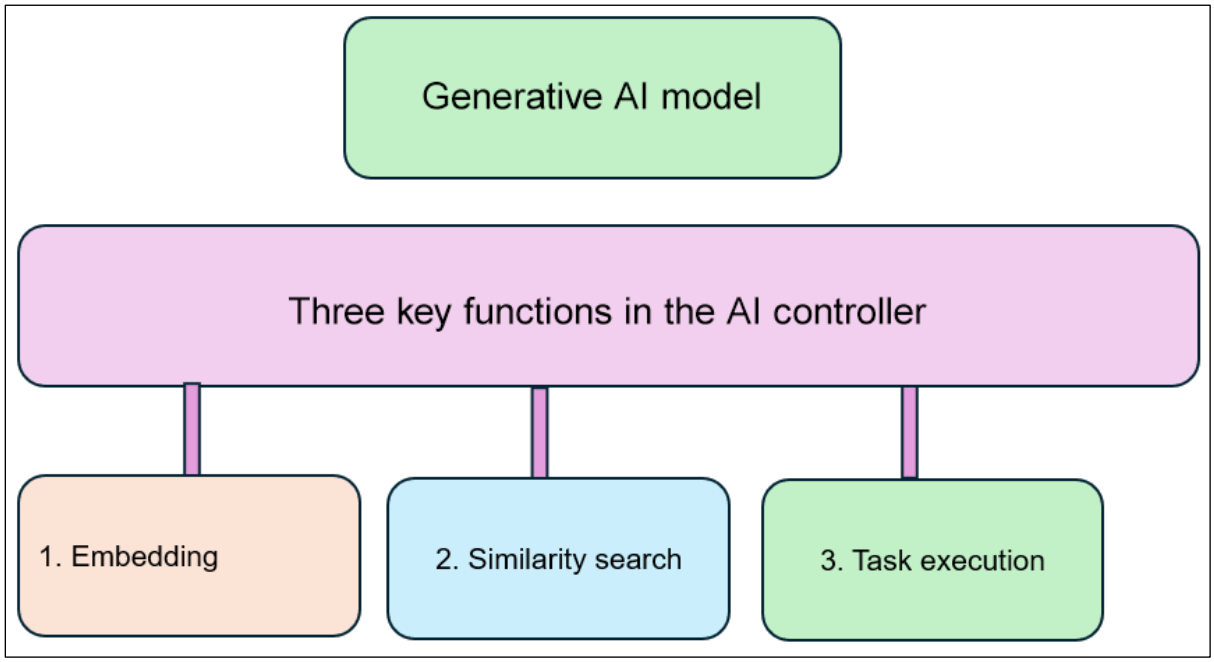

GenAISys AI controller
Input (from human user or system)
Conversational agent
AI controller orchestrator
Merging the conversational agent and the AI controller orchestrator through a Pinecone vector store

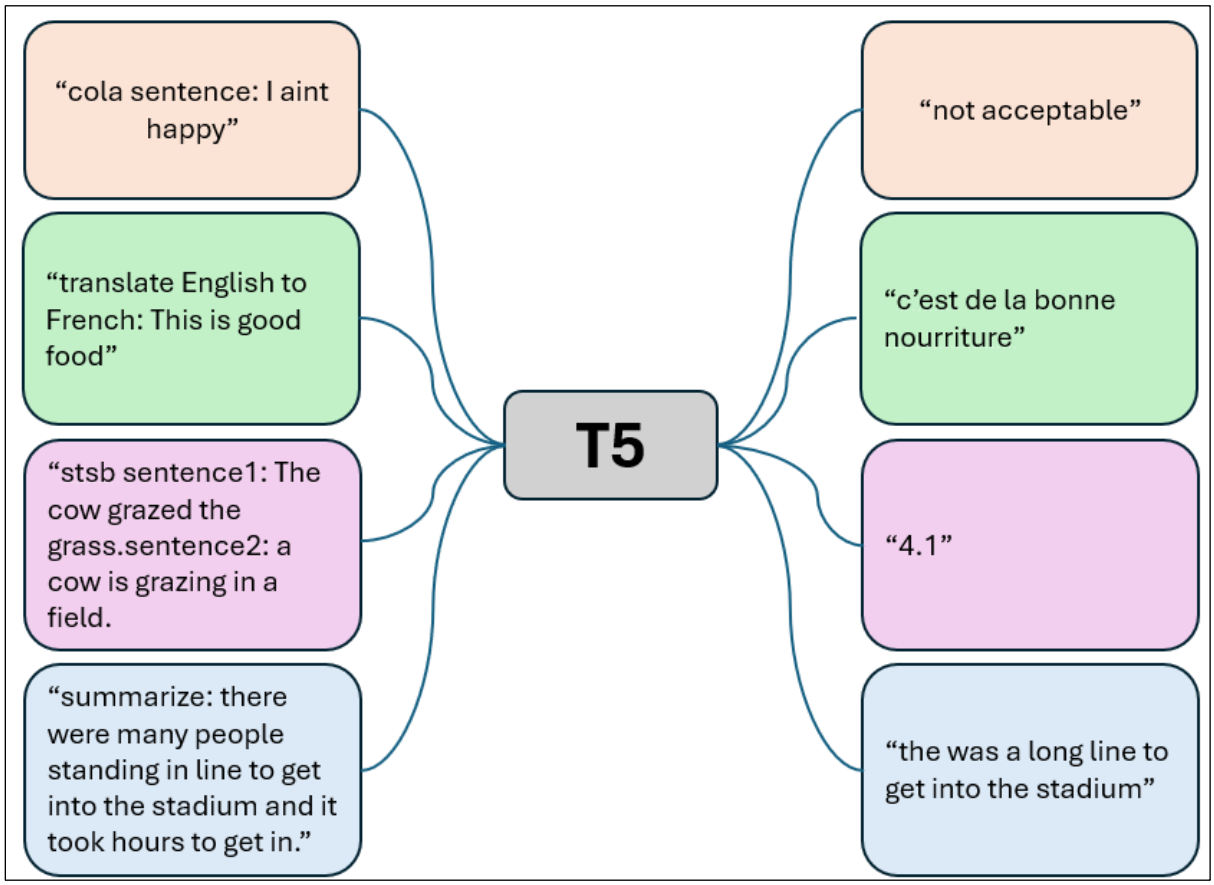

1. Start
2. Initialize Variables
3. Initialize messages Object
4. Print Welcome Message
5. Handle Initial User Input
6. Cleanse Conversation Log (Initial)
7. Make API Call (Initial)
8. Append API Response (Initial)
9. Display Assistant Response (Initial)

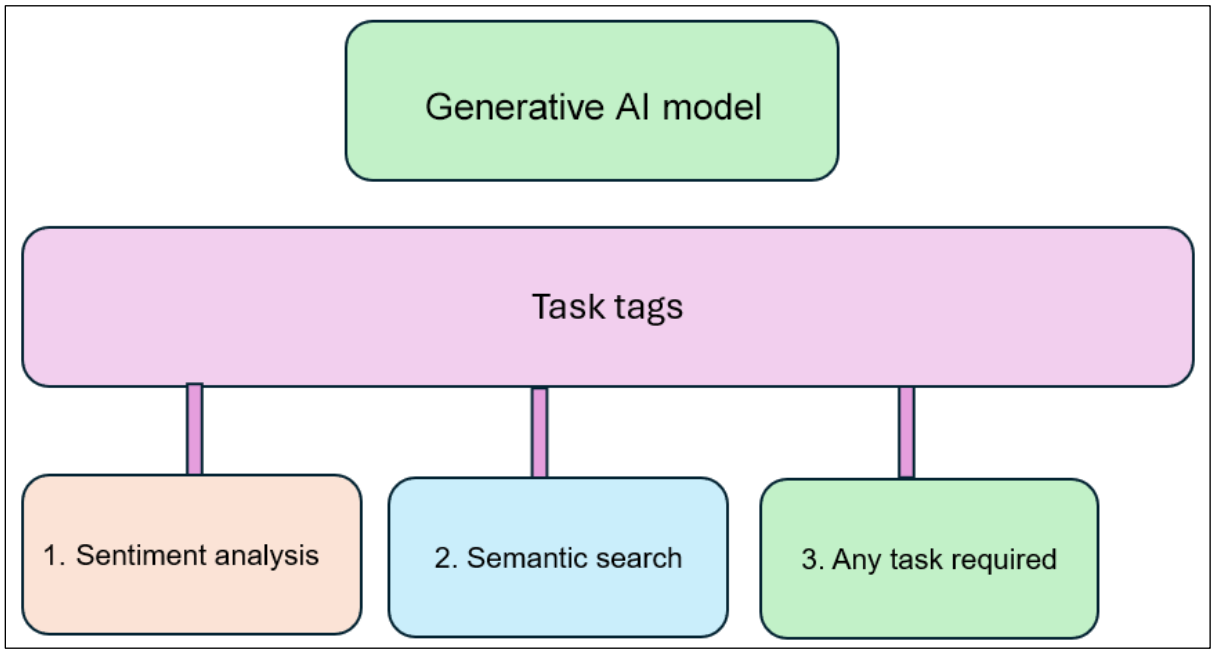

11. Prompt for User Input
12. Check Exit Condition
13. Append User Input to messages Object
14. Cleanse Conversation Log (Loop)
15. Make API Call (Loop)
16. Append API Response (Loop)
17. Display Assistant Response (Loop)
18. Exit and Save Conversation Log


User input
Agent response
Conversational Agent
Memory retention
Contextual awareness
Function calls
Entry/exit

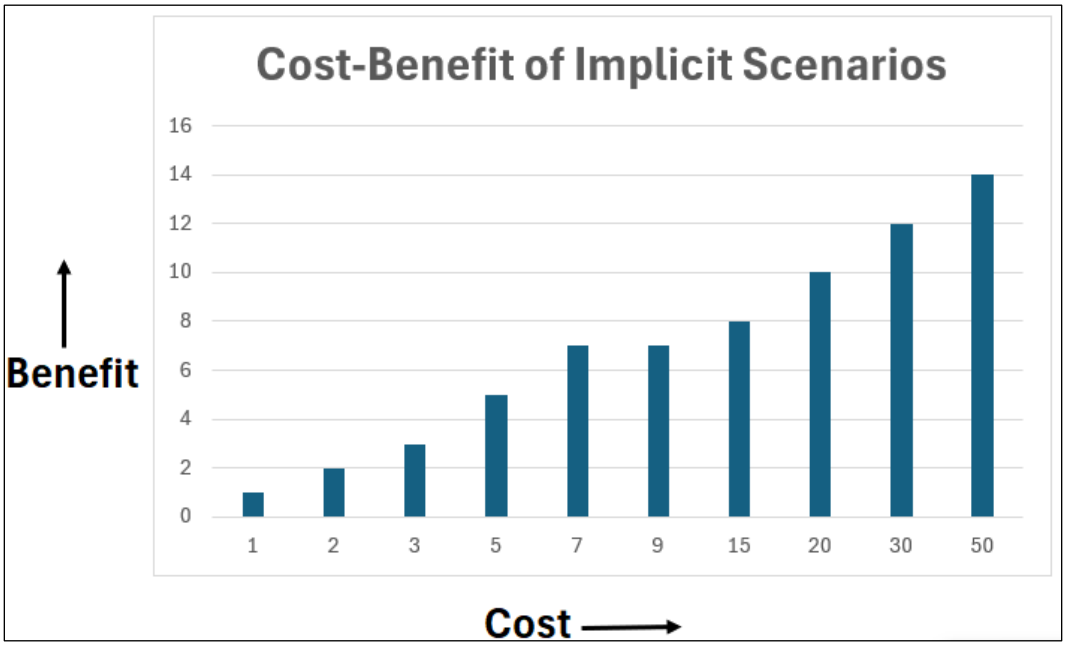

AI Controller Orchestrator
C1. AI controller entry point input
C2. Analyzes input
C3. Embeds user input
C4. Embeds task scenario repository
C5. Selects a scenario
C6. Executes the scenario

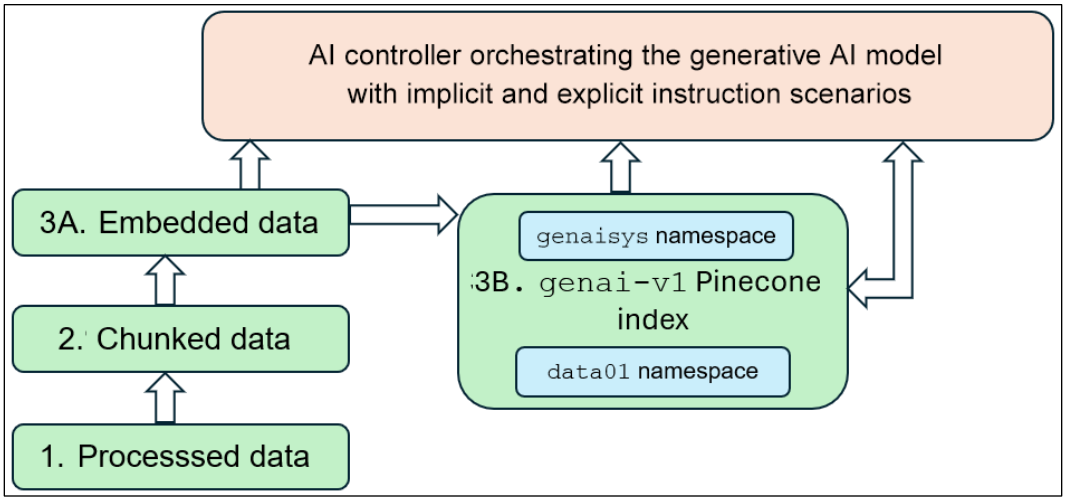

Generative AI model
Three key functions in the AI controller
1. Embedding
2. Similarity search
3. Task execution

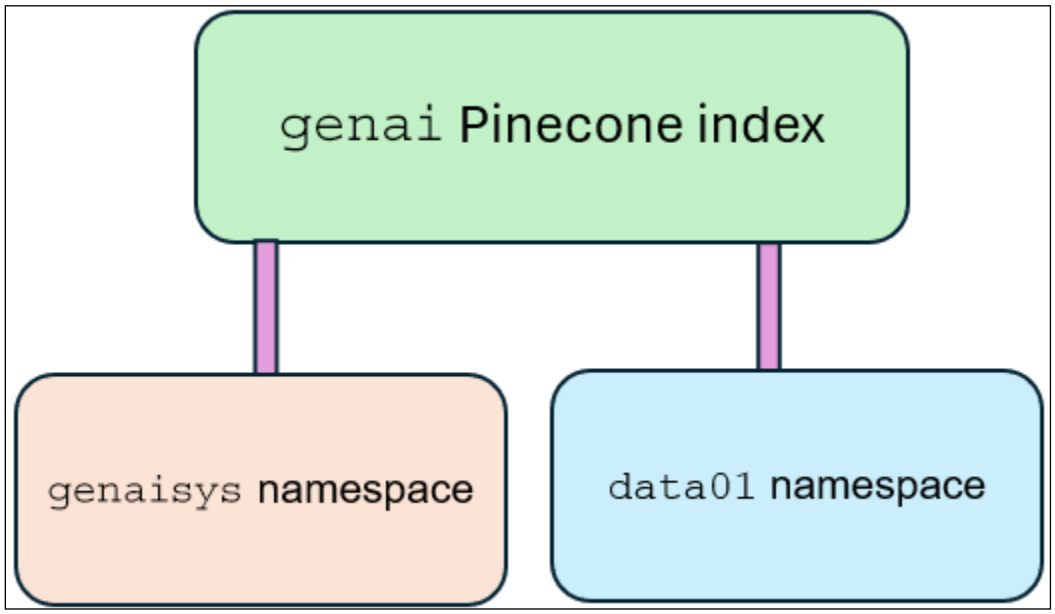

"cola sentence: I aint happy"
"translate English to French: This is good food"
"stsb sentence1: The cow grazed the grass.sentence2: a cow is grazing in a field.
"summarize: there were many people standing in line to get into the stadium and it took hours to get in."
T5
"not acceptable"
"c'est de la bonne nourriture"
"4.1"
"the was a long line to get into the stadium"


Generative AI model
Task tags
1. Sentiment analysis
2. Semantic search
3. Any task required

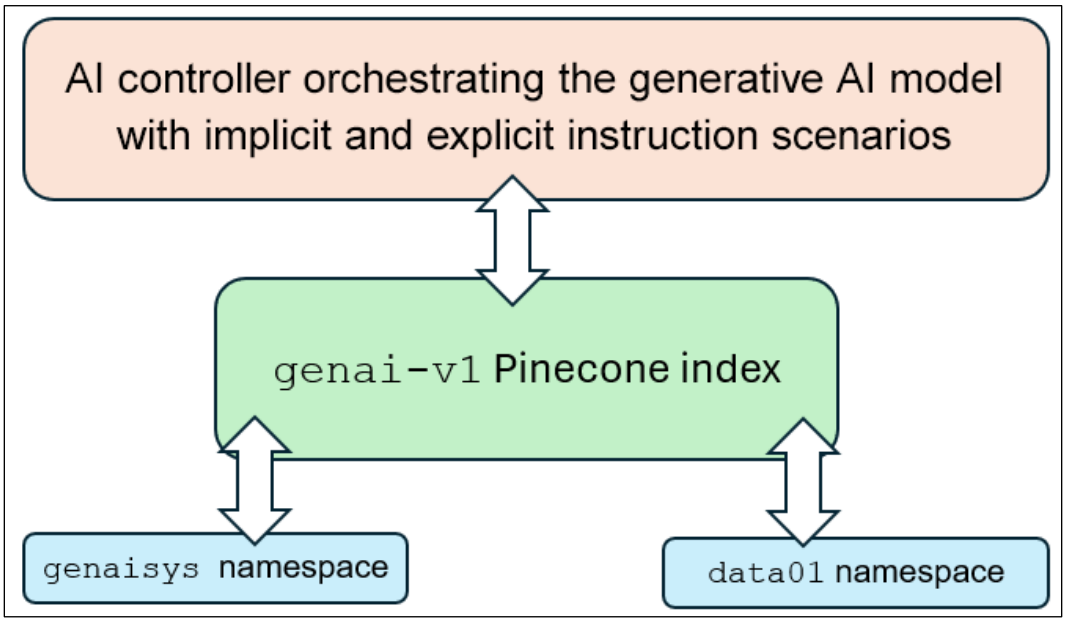
Chapter 3: Integrating Dynamic RAG into the GenAISys

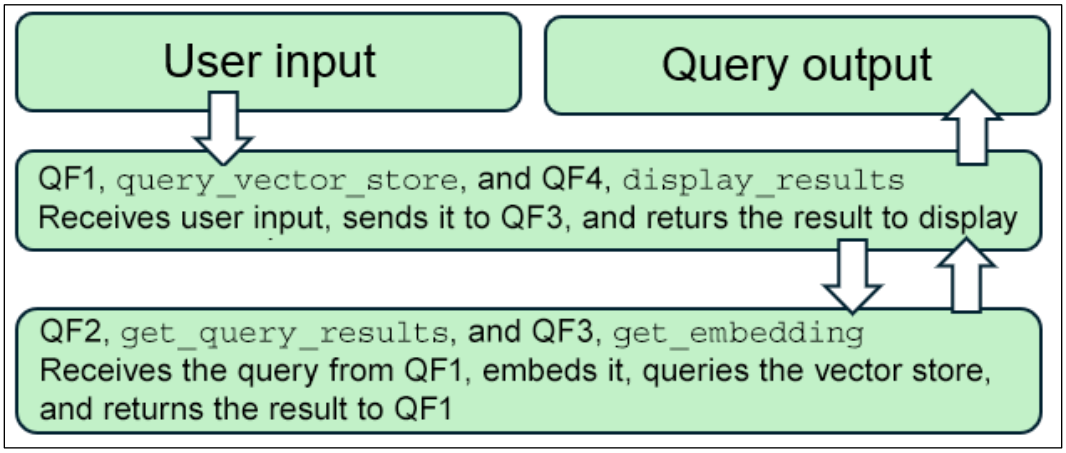




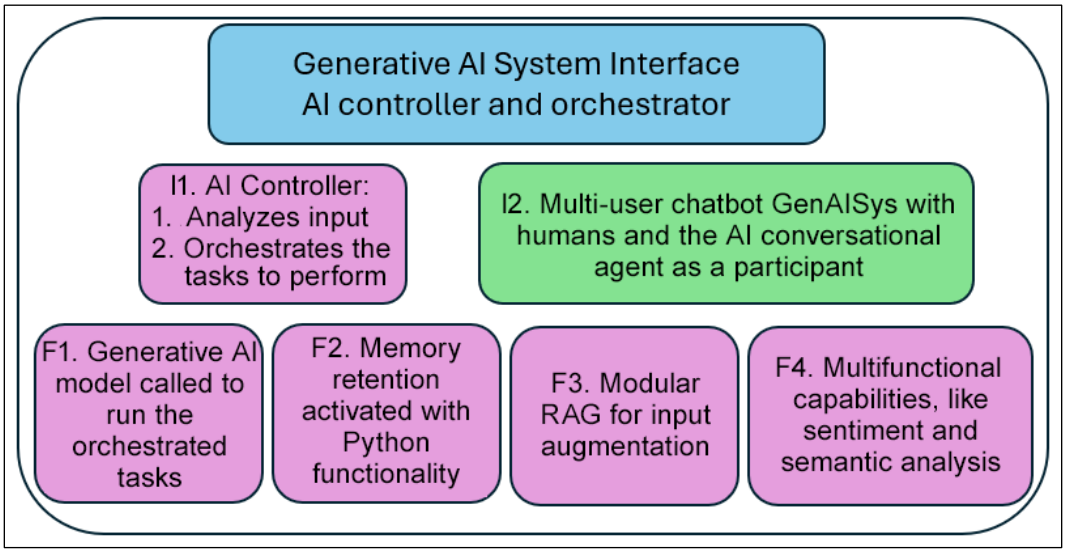


Cost over time ⓘ Download

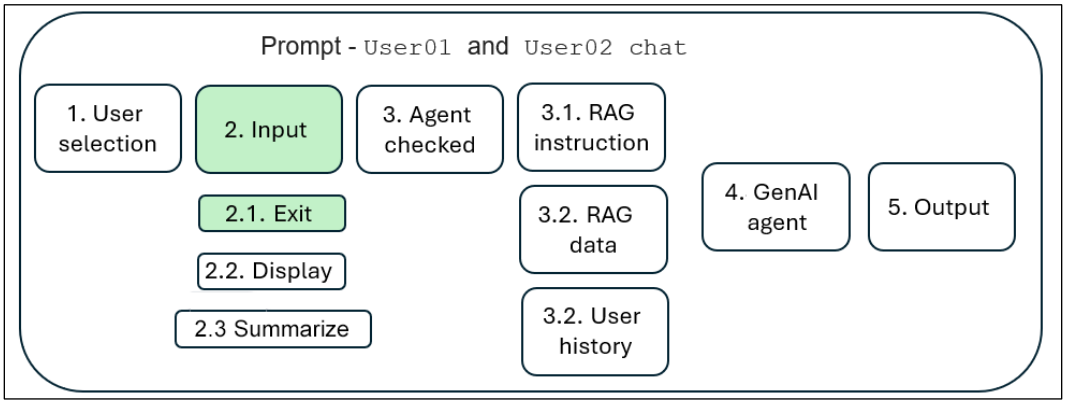



User input
Query output
QF1, query_vector_store, and QF4, display_results
Receives user input, sends it to QF3, and returs the result to display
QF2, get_query_results, and QF3, get_embedding
Receives the query from QF1, embeds it, queries the vector store, and returns the result to QF1

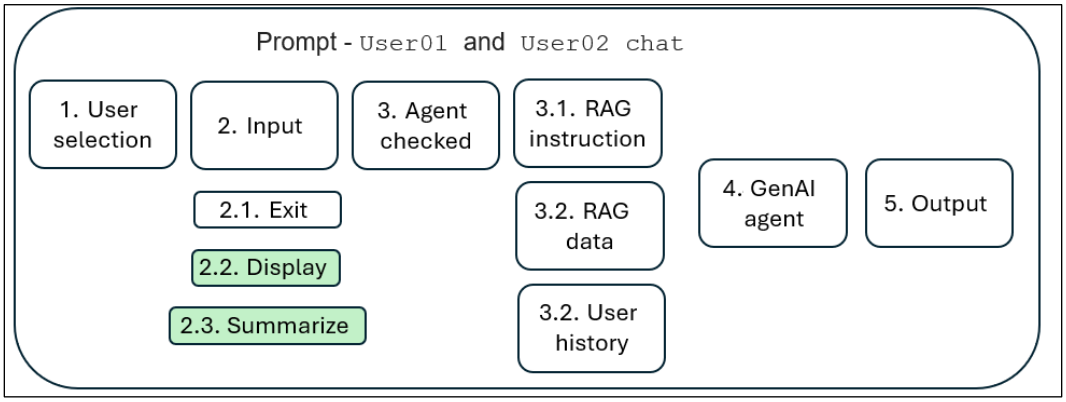
Chapter 4: Building the AI Controller Orchestration Interface

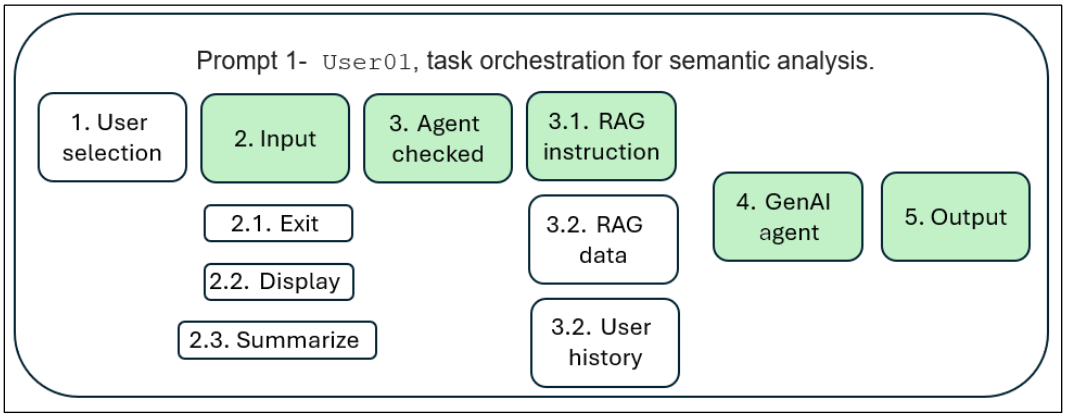


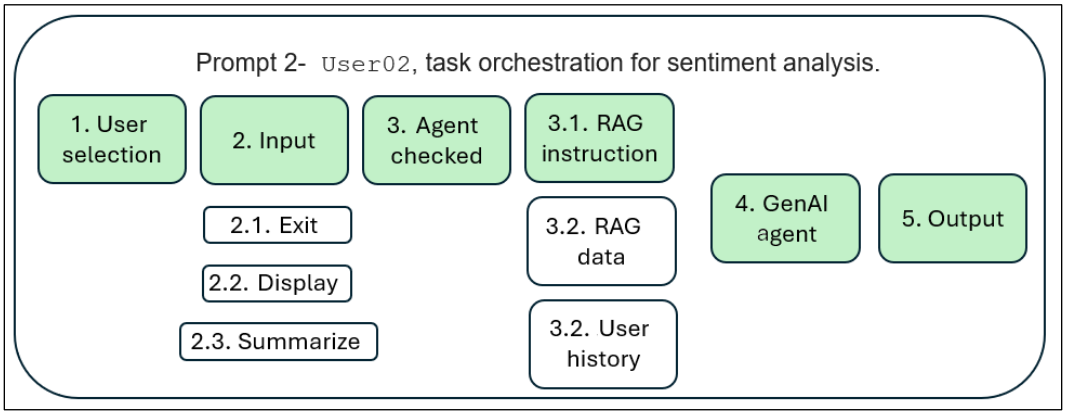


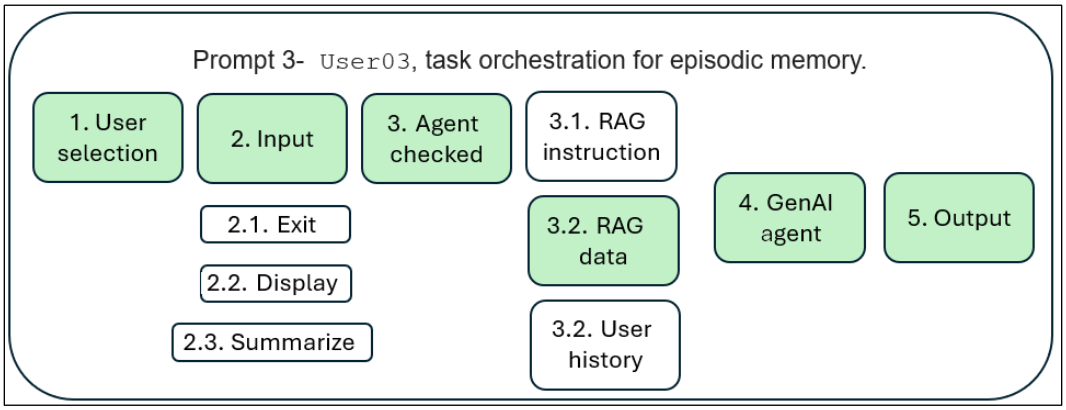


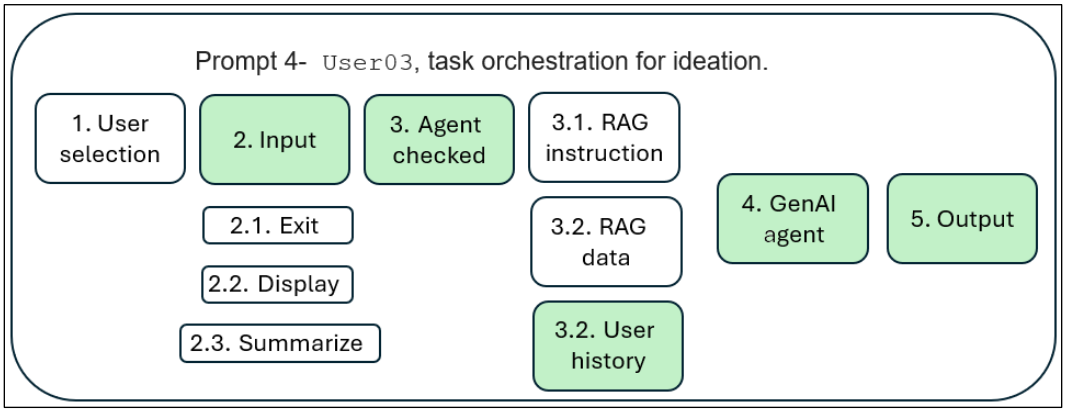

```
Conversation ended. Saving history...
Conversation history saved to conversation_history.json.
History saved. Proceed to the next cell.
```

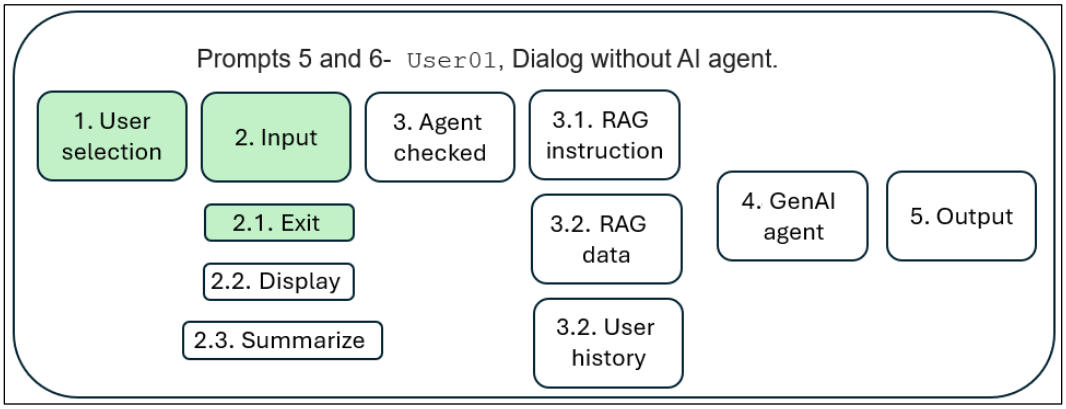




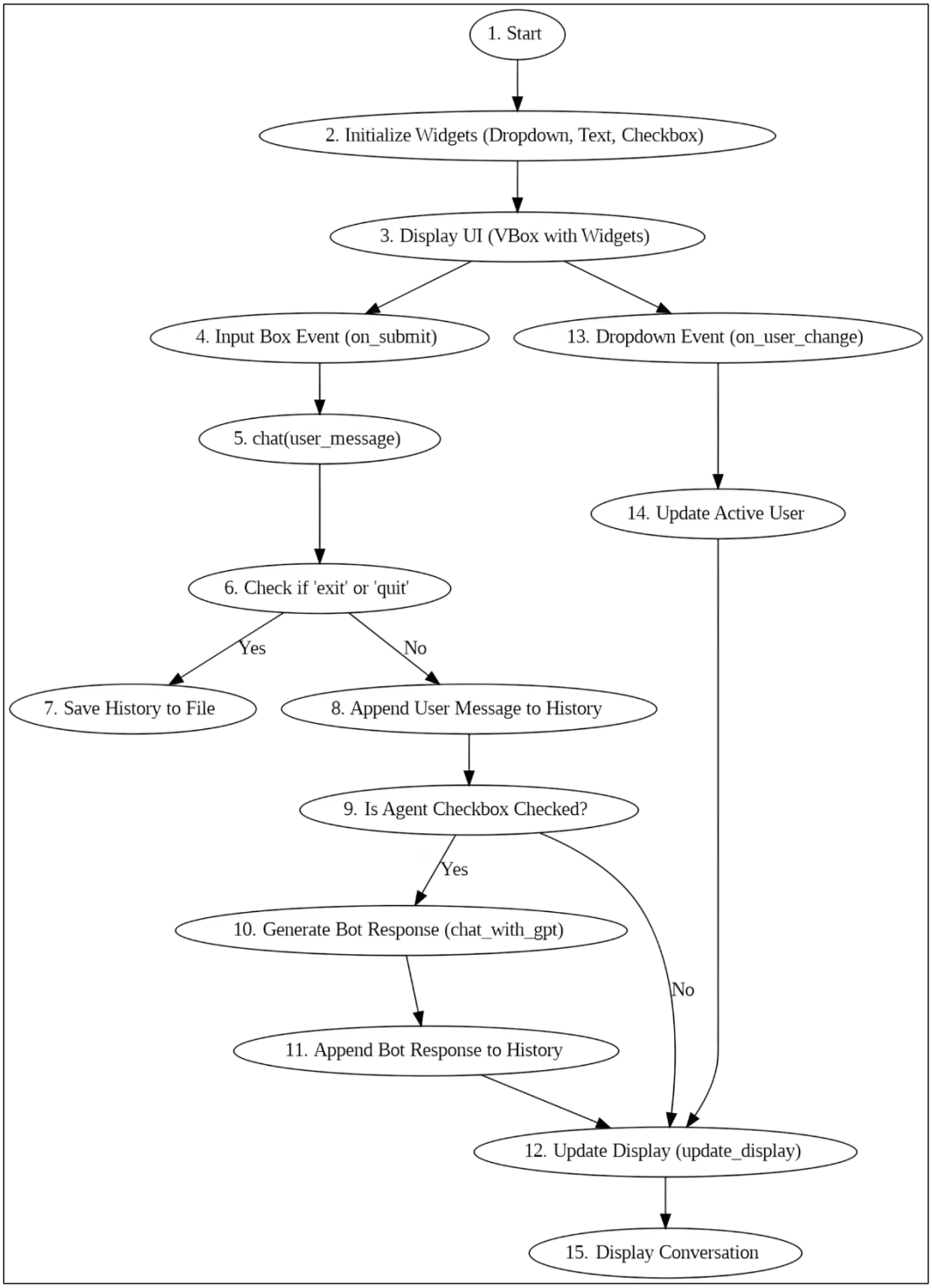

Prompt 2- User02, task orchestration for sentiment analysis.
1. User selection
2. Input
2.1. Exit
2.2. Display
2.3. Summarize
3. Agent checked
3.1. RAG instruction
3.2. RAG data
3.2. User history
4. GenAI agent
5. Output


Prompt 3- User03, task orchestration for episodic memory.
1. User selection
2. Input
2.1. Exit
2.2. Display
2.3. Summarize
3. Agent checked
3.1. RAG instruction
3.2. RAG data
3.2. User history
4. GenAI agent
5. Output


Prompt 4- User03, task orchestration for ideation.
1. User selection
2. Input
2.1. Exit
2.2. Display
2.3. Summarize
3. Agent checked
3.1. RAG instruction
3.2. RAG data
3.2. User history
4. GenAI agent
5. Output


Prompts 5 and 6- User01, Dialog without AI agent.
1. User selection
2. Input
3. Agent checked
3.1. RAG instruction
2.1. Exit
3.2. RAG data
4. GenAI agent
5. Output
2.2. Display
2.3. Summarize
3.2. User history


User:
User01
exit
Agent

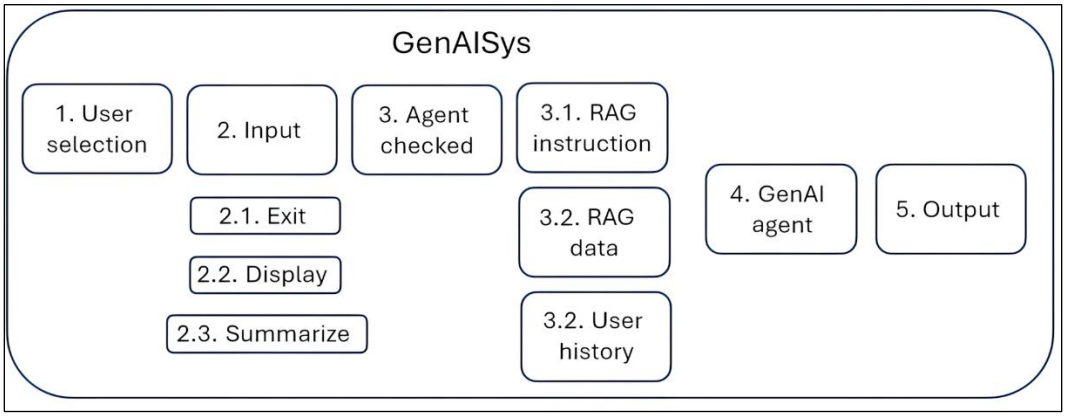

1. Start
2. Initialize Widgets (Dropdown, Text, Checkbox)
3. Display UI (VBox with Widgets)
4. Input Box Event (on_submit)
13. Dropdown Event (on_user_change)
5. chat(user_message)
14. Update Active User
6. Check if 'exit' or 'quit'
Yes
No
7. Save History to File
8. Append User Message to History
9. Is Agent Checkbox Checked?
Yes
No
10. Generate Bot Response (chat_with_gpt)
11. Append Bot Response to History
12. Update Display (update_display)
15. Display Conversation

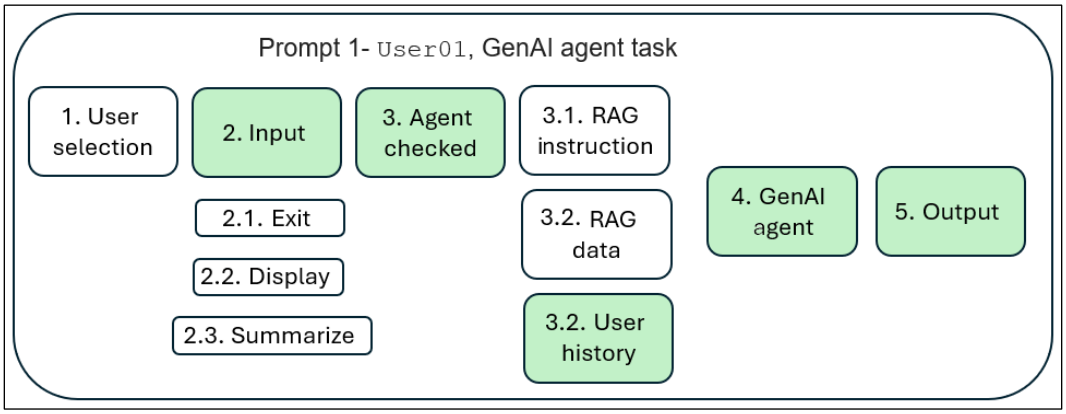
```
Conversation ended. Saving history...
Conversation history saved to conversation_history.json.
History saved. Proceed to the next cell.
```

User: User01

Type your message here or type 'exit' or 'quit' to end the conversation.

Agent

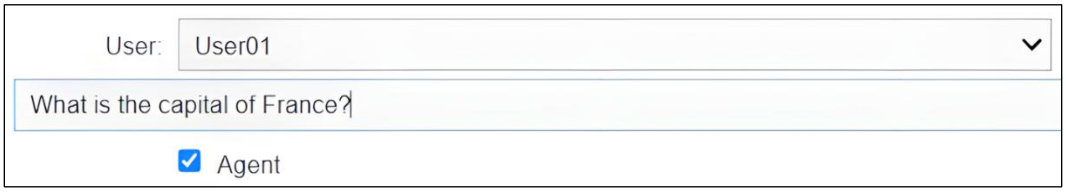


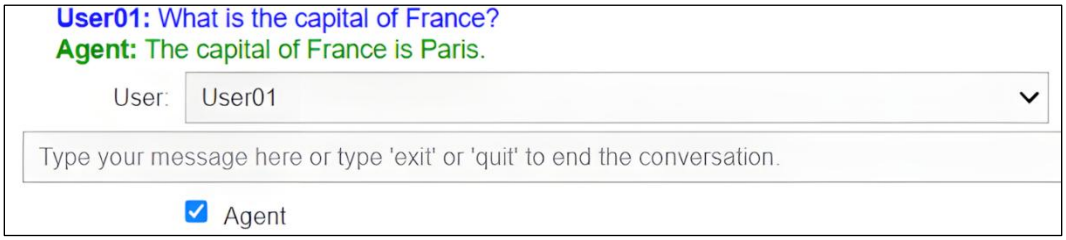


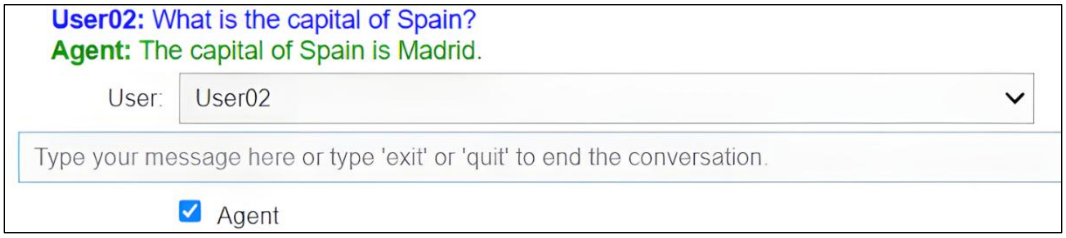
User: User01

Type your message here or type 'exit' or 'quit' to end the conversation.

Agent

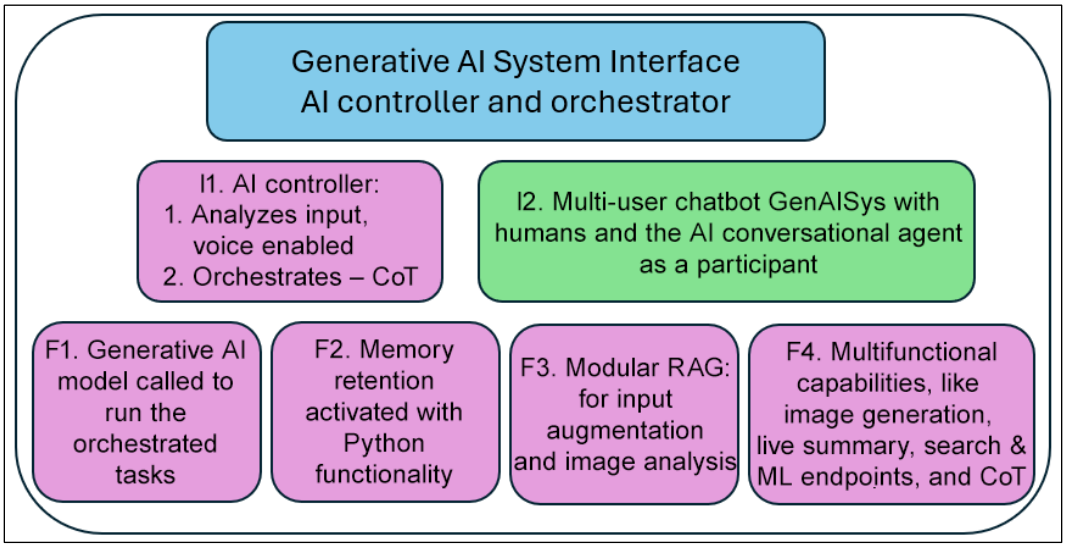

User:
User01
What is the capital of France?
Agent


User01: What is the capital of France?
Agent: The capital of France is Paris.
User:
User01
Type your message here or type 'exit' or 'quit' to end the conversation.
Agent


User02: What is the capital of Spain?
Agent: The capital of Spain is Madrid.
User:
User02
Type your message here or type 'exit' or 'quit' to end the conversation.
Agent

$$S = \{a, b, c, d, e\}$$

Chapter 5: Adding Multimodal, Multifunctional Reasoning with Chain of Thought




```
Process: Performing machine learning analysis of the customer database.
```

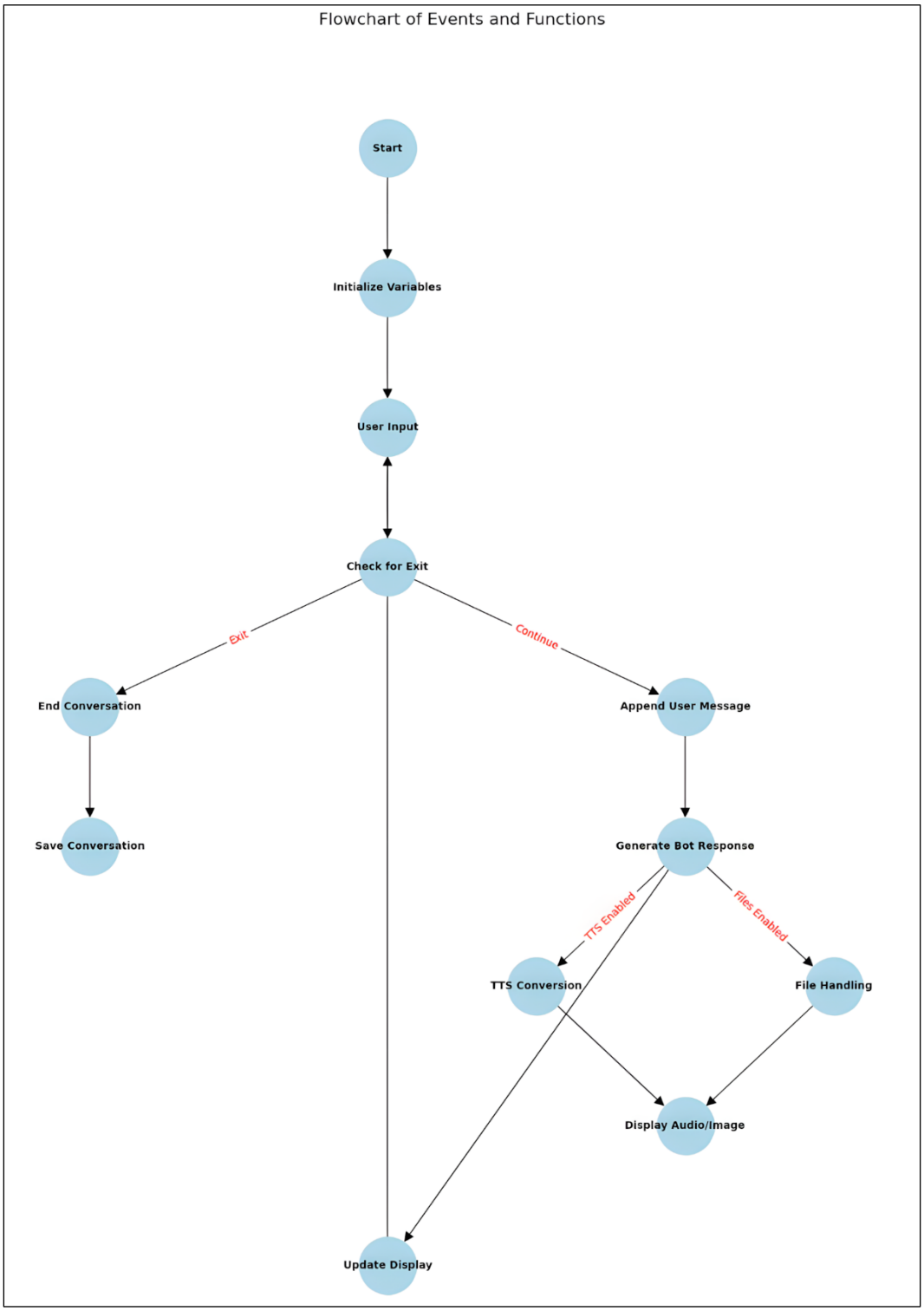
```
Process: Searching for activities that fit the customer needs.
```

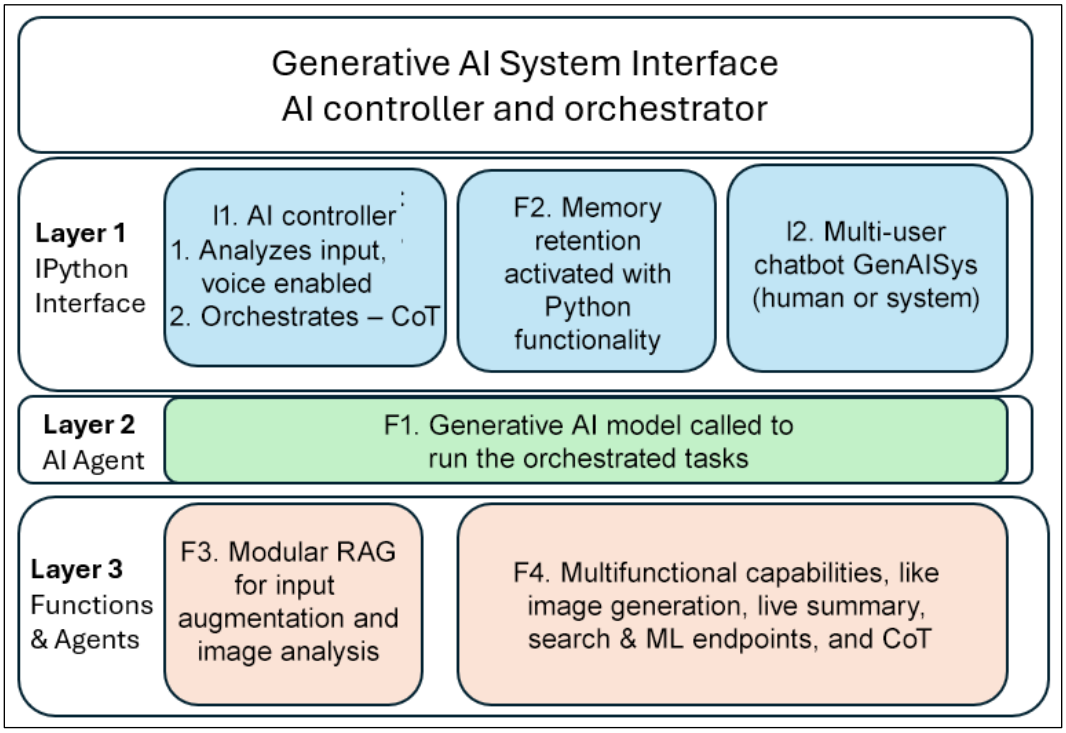
```
Process: Generating an image based on the ideation.
```

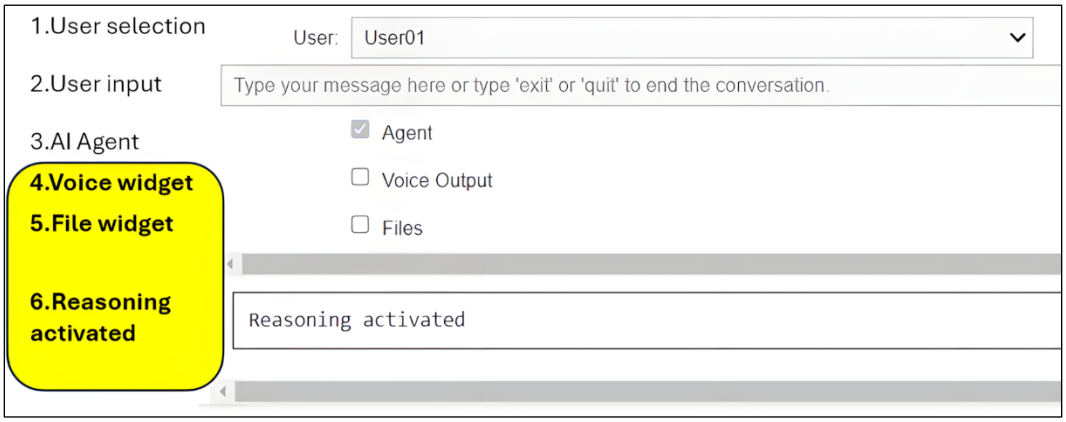

EDUCATIONAL & HISTORICAL

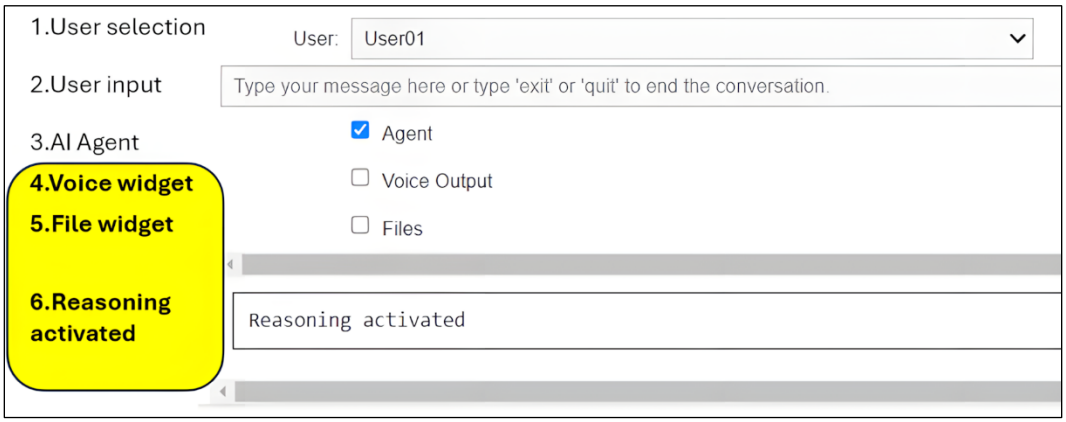
```
Process: Providing an engaging story based on the generated image.
```

```
Reasoning activated
```

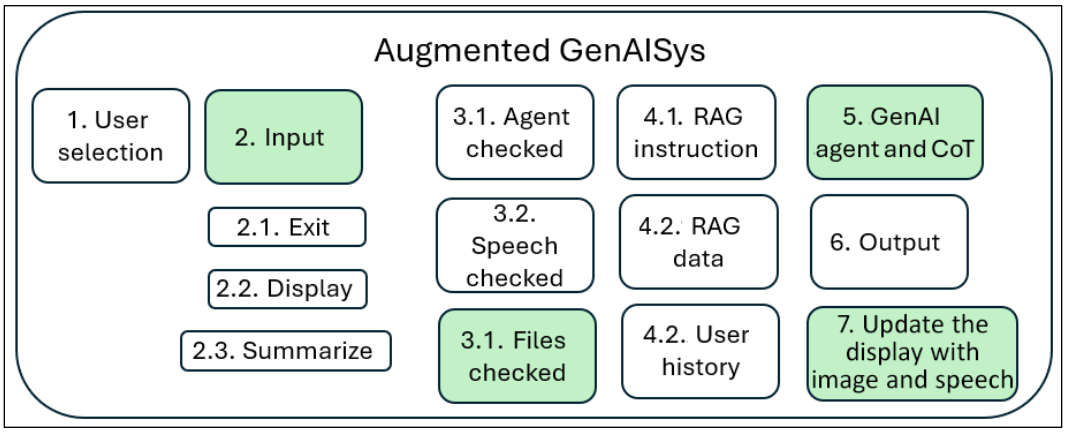

Flowchart of Events and Functions
Start
Initialize Variables
User Input
Check for Exit
Exit
Continue
End Conversation
Save Conversation
Append User Message
Generate Bot Response
TTS Enabled
Files Enabled
TTS Conversion
File Handling
Display Audio/Image
Update Display


Generative AI System Interface
AI controller and orchestrator
Layer 1
IPython
Interface
I1. AI controller
1. Analyzes input, voice enabled
2. Orchestrates – CoT
F2. Memory retention activated with Python functionality
I2. Multi-user chatbot GenAISys (human or system)
Layer 2
AI Agent
F1. Generative AI model called to run the orchestrated tasks
Layer 3
Functions
& Agents
F3. Modular RAG for input augmentation and image analysis
F4. Multifunctional capabilities, like image generation, live summary, search & ML endpoints, and CoT


1.User selection
User:
User01
2.User input
Type your message here or type 'exit' or 'quit' to end the conversation.
3.AI Agent
Agent
4.Voice widget
Voice Output
5.File widget
Files
6.Reasoning activated
Reasoning activated



CUSTOMER_ID	LOCATION	ACTIVITY	RATING
729	Paris	Montmartre	7
970	Athens	Acropole	5
356	Athens	Acropole	7
340	Paris	Eiffel Tower	3
394	Paris	Parliament	5
980	Washington	Senate	3
633	Athens	restaurants	4
101	Rome	walking	6
954	Washington	statues	3
350	Athens	restaurants	2



User: User01

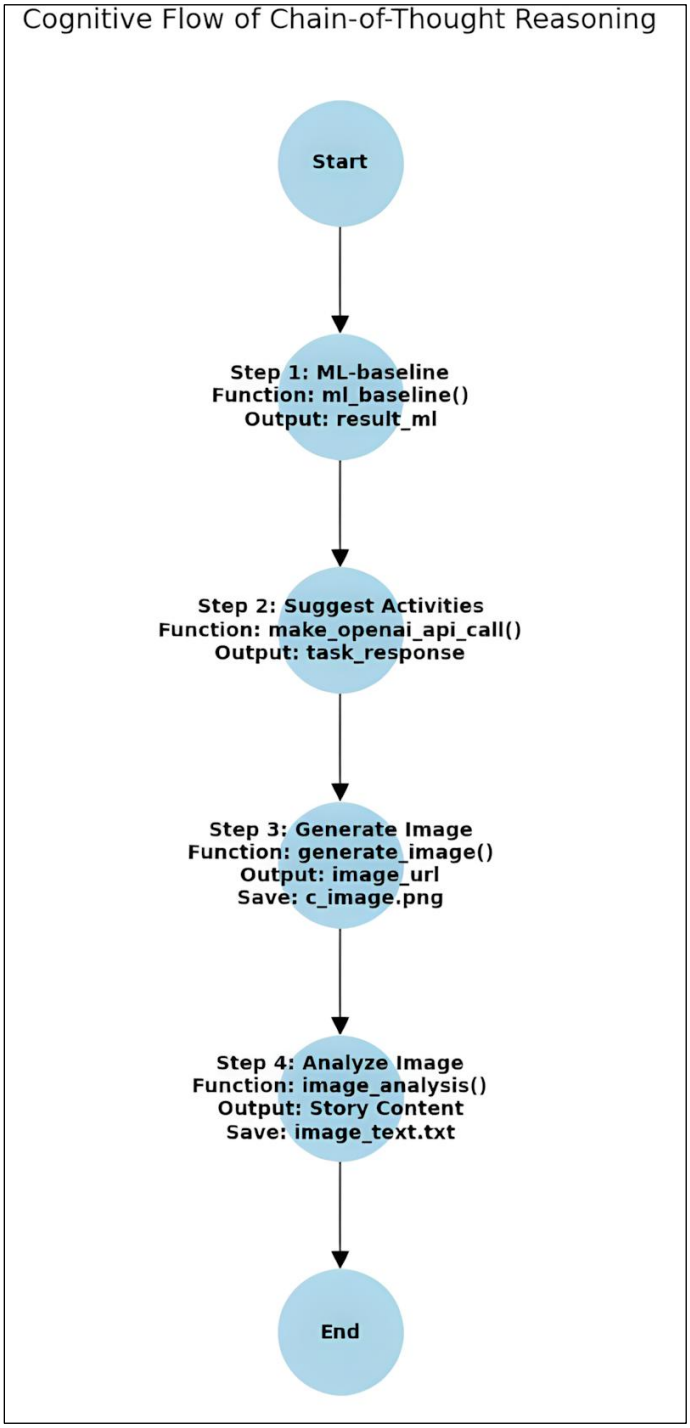
Create an image of a cruise ship around Antartica

- [x] Agent
- [ ] Voice Output
- [x] Files

User01: Create an image of a cruise ship near Antartica

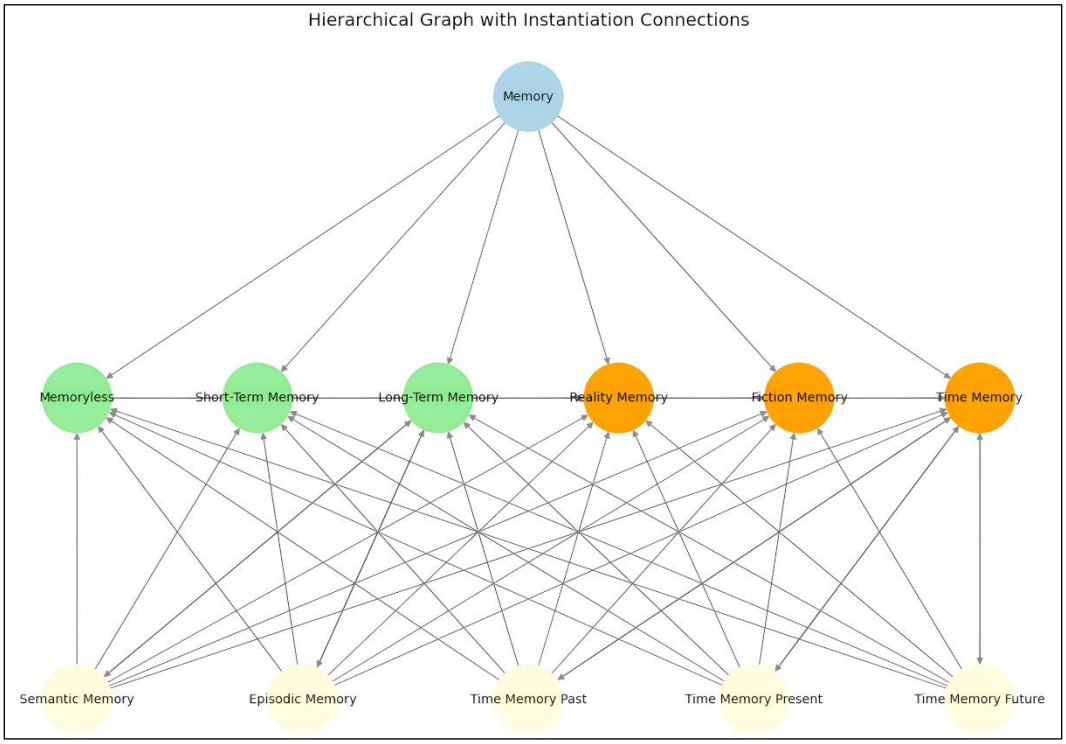
Agent: Image created

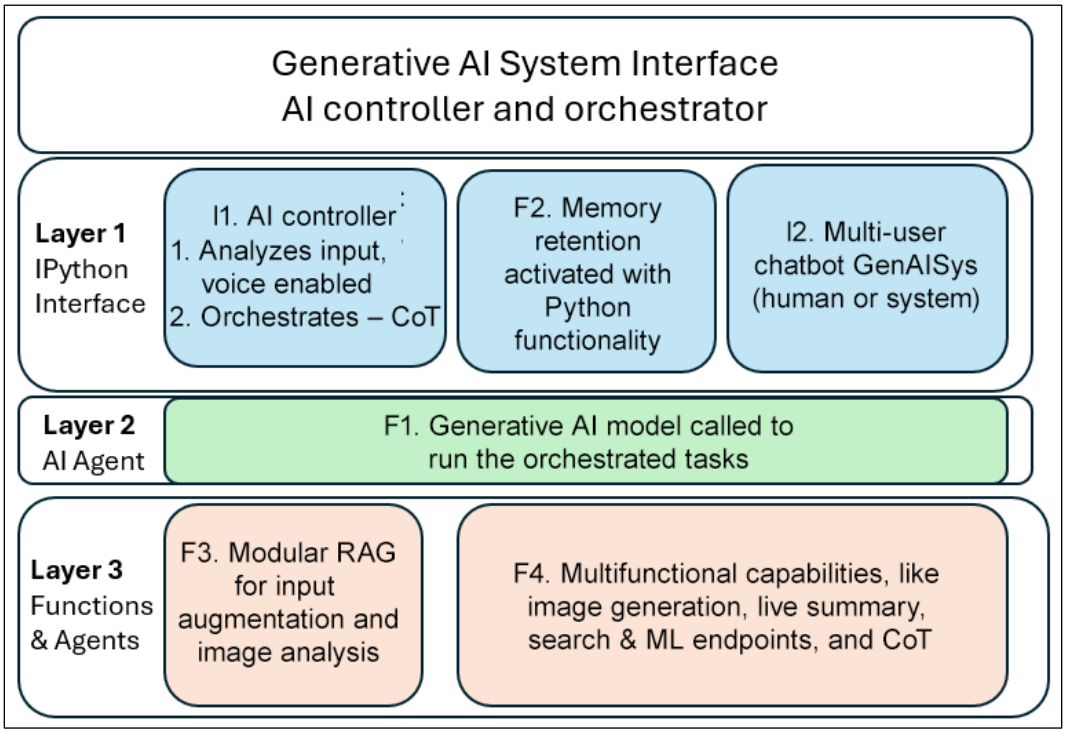


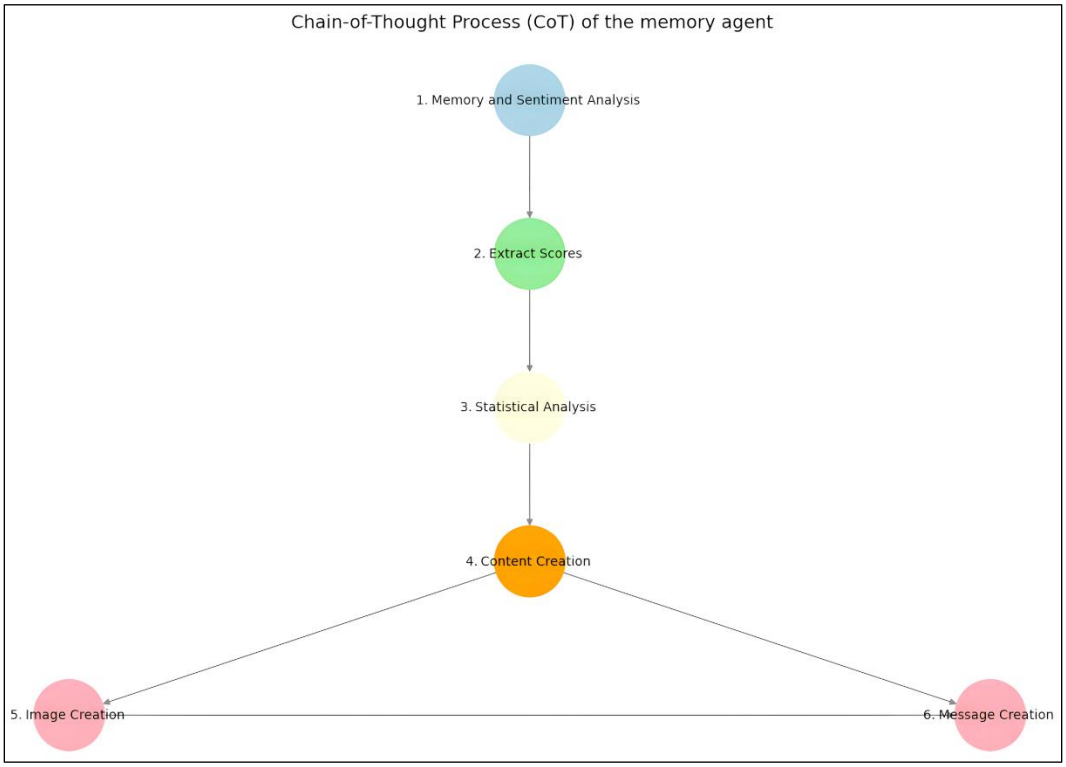

Cognitive Flow of Chain-of-Thought Reasoning
Start
Step 1: ML-baseline
Function: ml_baseline()
Output: result_ml
Step 2: Suggest Activities
Function: make_openai_api_call()
Output: task_response
Step 3: Generate Image
Function: generate_image()
Output: image_url
Save: c_image.png
Step 4: Analyze Image
Function: image_analysis()
Output: Story Content
Save: image_text.txt
End

Chapter 6: Reasoning E-Marketing AI Agents




Hierarchical Graph with Instantiation Connections
Memory
Memoryless
Short-Term Memory
Long-Term Memory
Reality Memory
Fiction Memory
Time Memory
Semantic Memory
Episodic Memory
Time Memory Past
Time Memory Present
Time Memory Future


Generative AI System Interface
AI controller and orchestrator
Layer 1
IPython
Interface
I1. AI controller
1. Analyzes input, voice enabled
2. Orchestrates – CoT
F2. Memory retention activated with Python functionality
I2. Multi-user chatbot GenAISys (human or system)
Layer 2
AI Agent
F1. Generative AI model called to run the orchestrated tasks
Layer 3
Functions
& Agents
F3. Modular RAG for input augmentation and image analysis
F4. Multifunctional capabilities, like image generation, live summary, search & ML endpoints, and CoT



index	Review	Rating
0	We got a nice hotel. The parking was rather expensive. However we got a good deal for my birthday. We arrived in during the night because of a late flight but the parking was open 24hours The check in was rapid. The room was a large size. Cool. But we didn't get the view expected. The bed was super comfortable and got a nice sleep for a few hours but then there was a raucous of a noisy crowd going to bed late. Anyway the price was acceptable and we made it to the shopping mall next to the hotel which reminded me of the one we had at home.	3
1	An absolutely flawless experience from start to finish. The valet was courteous check-in was a breeze and they even upgraded us to a suite with a breathtaking city view. The room was spotless the bed felt like sleeping on a cloud and the complimentary breakfast was restaurant-quality. It's expensive but you get what you pay for. We will definitely be back.	5



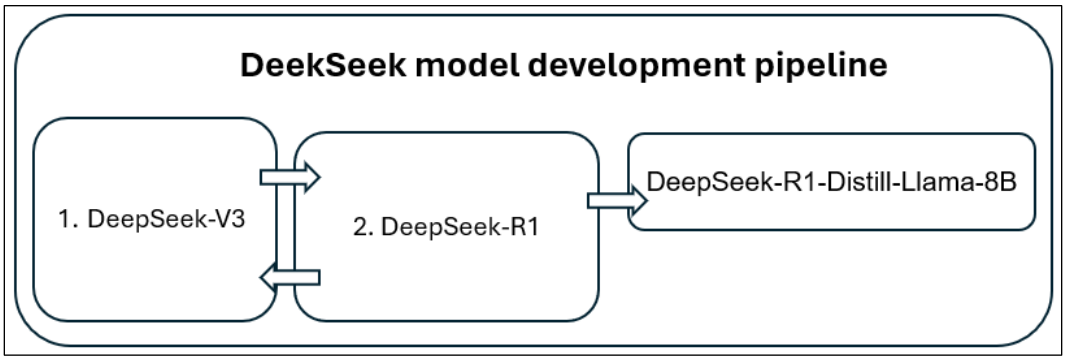
- [x] Agent
- [ ] Voice Output
- [ ] Files

None

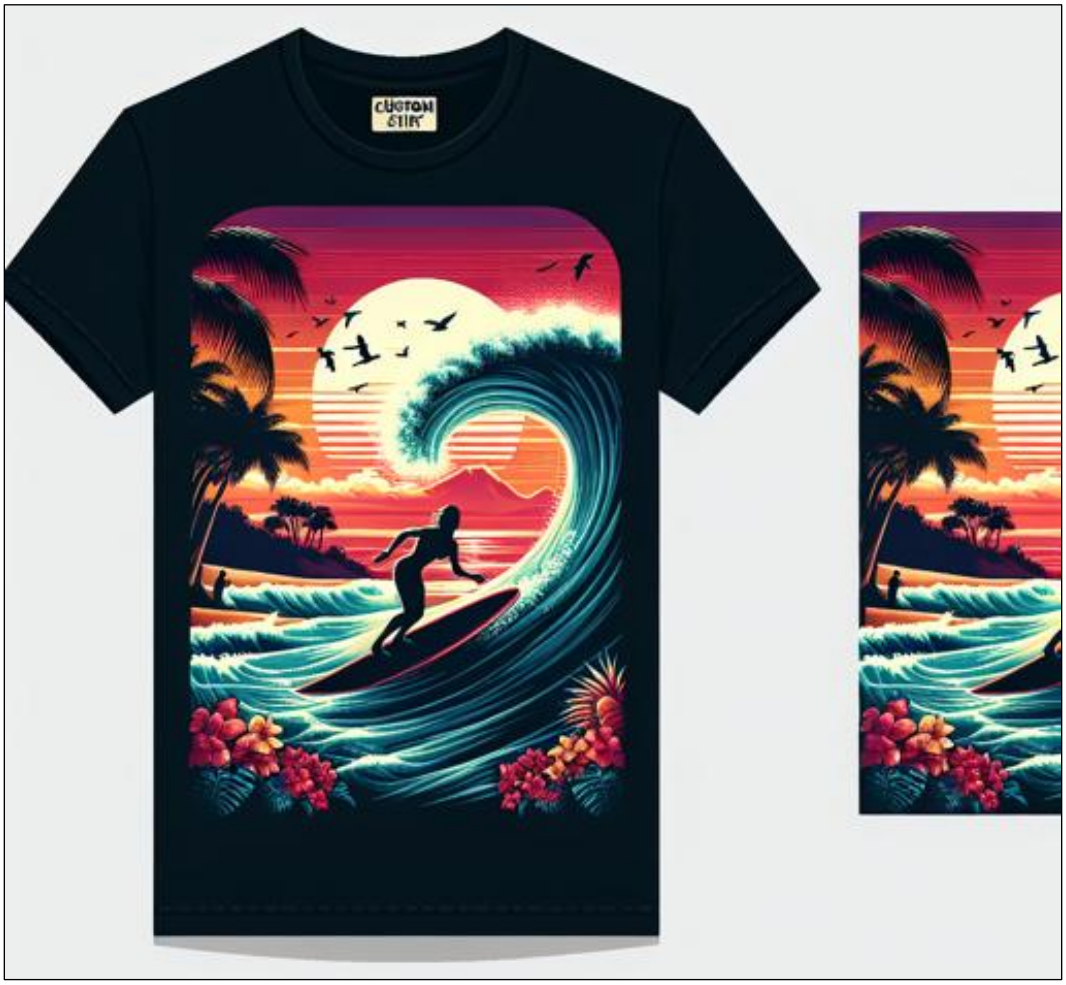
None
Analysis
Generation

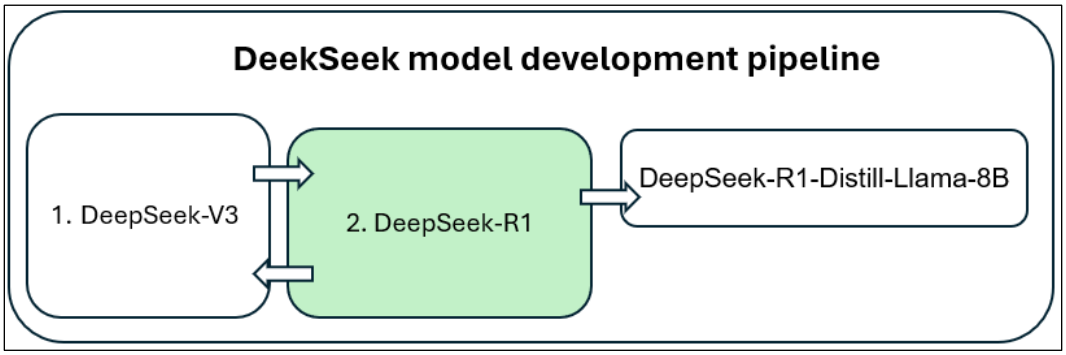


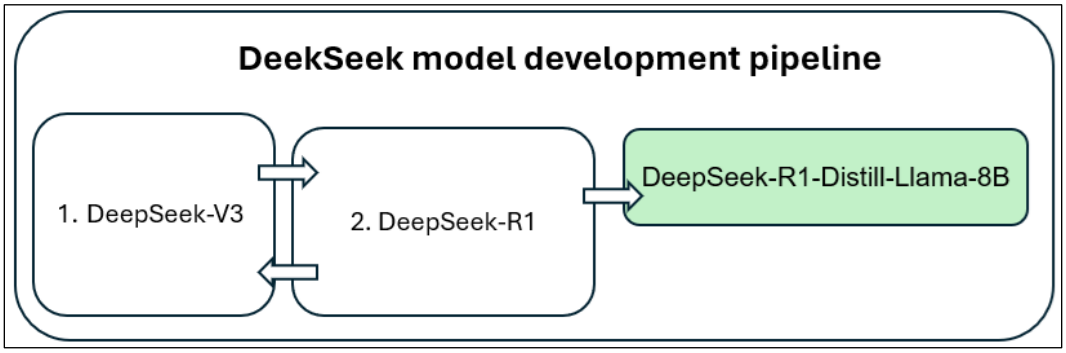
Chapter 7: Enhancing the GenAISys with DeepSeek

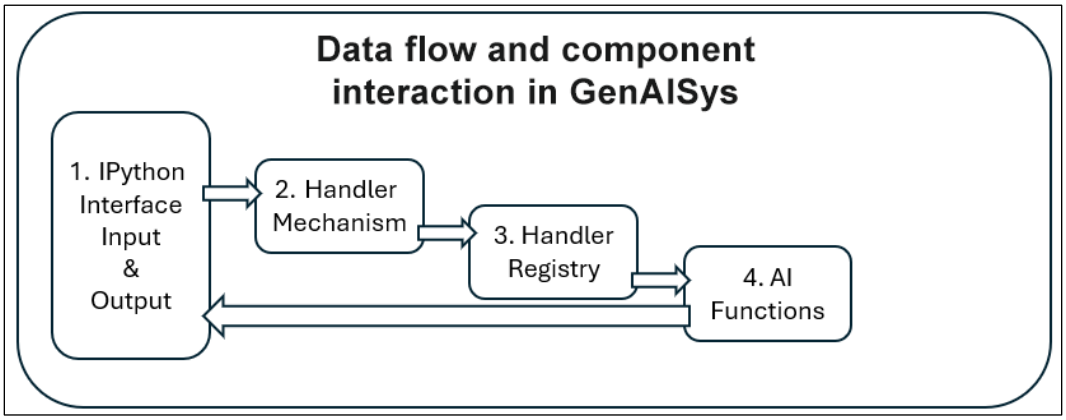


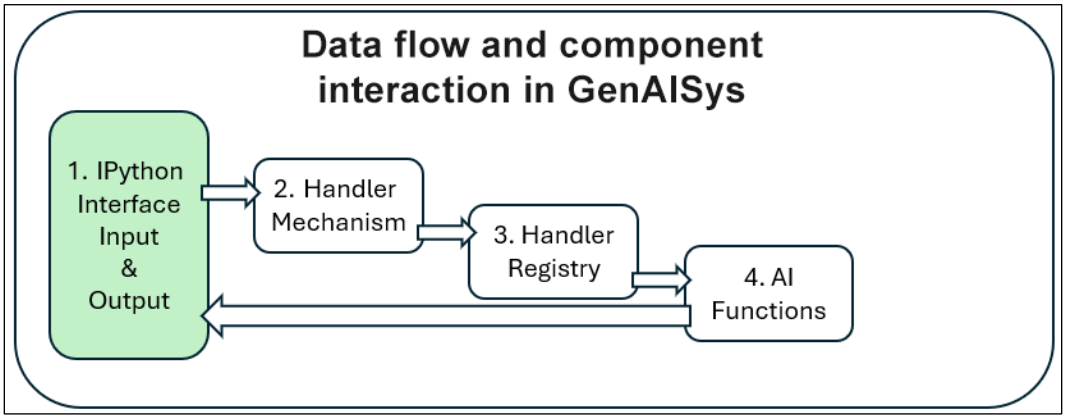


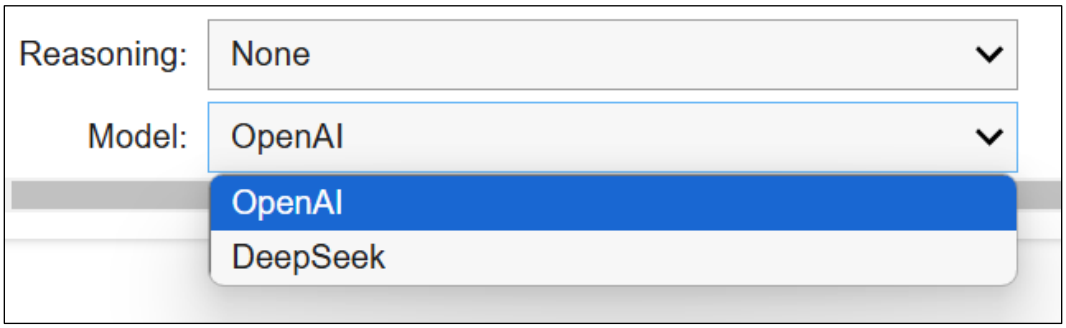


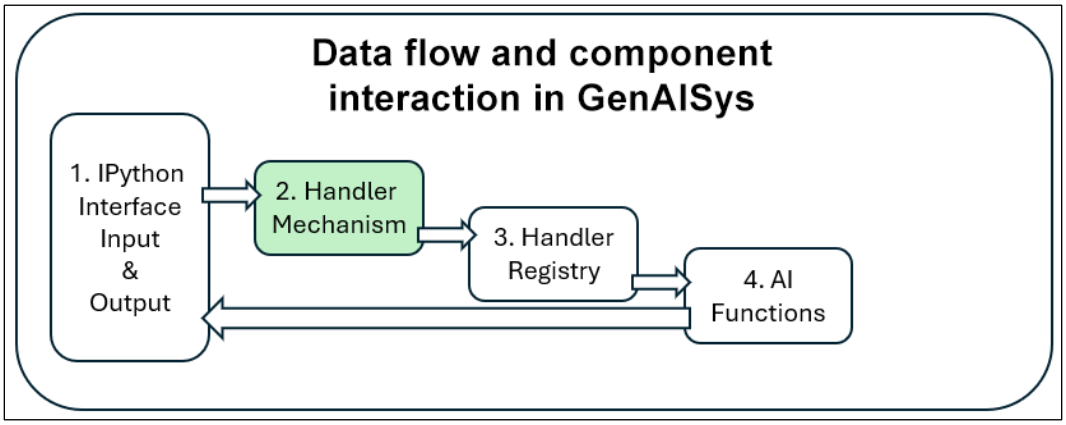

DeekSeek model development pipeline
1. DeepSeek-V3
2. DeepSeek-R1
DeepSeek-R1-Distill-Llama-8B

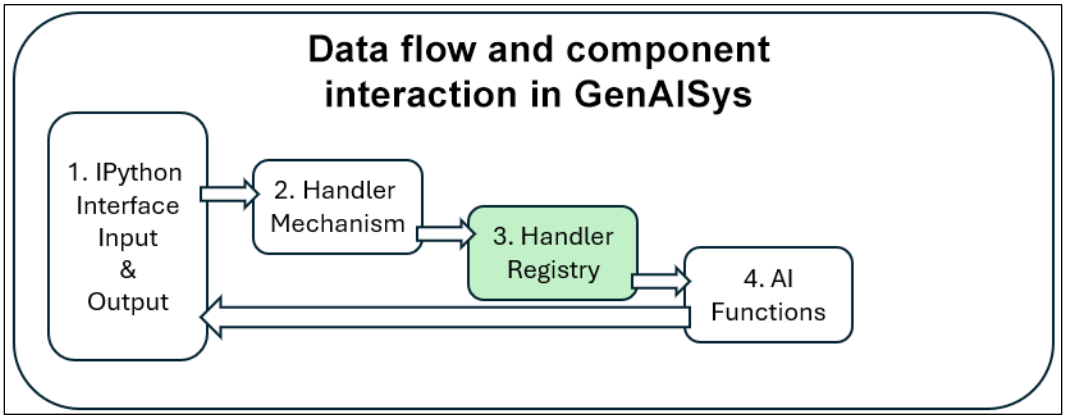

DeekSeek model development pipeline
1. DeepSeek-V3
2. DeepSeek-R1
DeepSeek-R1-Distill-Llama-8B

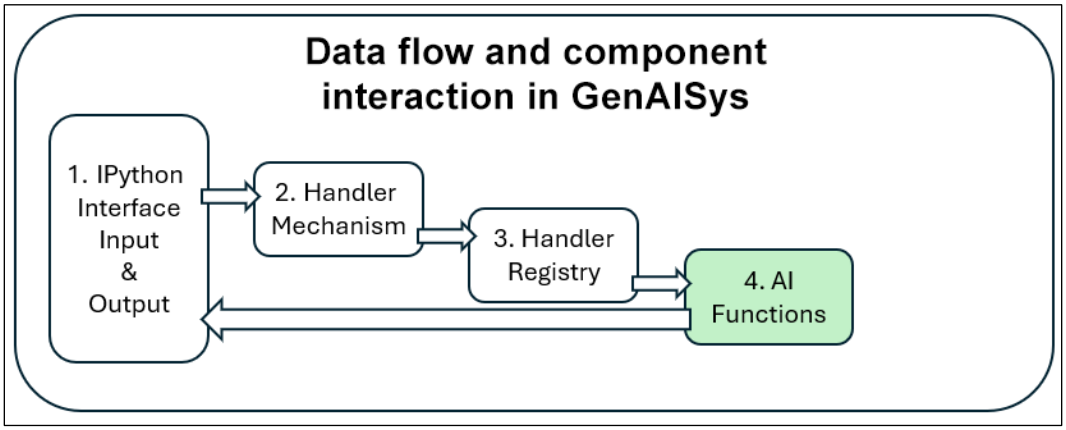

Data flow and component interaction in GenAISys
1. IPython Interface Input & Output
2. Handler Mechanism
3. Handler Registry
4. AI Functions


Data flow and component interaction in GenAISys
1. IPython Interface Input & Output
2. Handler Mechanism
3. Handler Registry
4. AI Functions

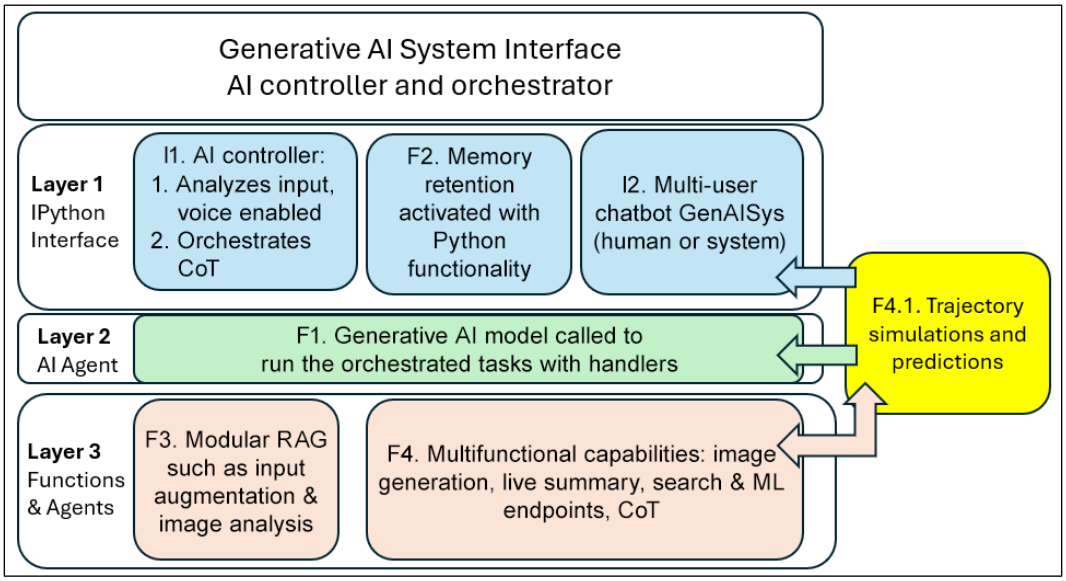

Reasoning:
None
Model:
OpenAI
OpenAI
DeepSeek

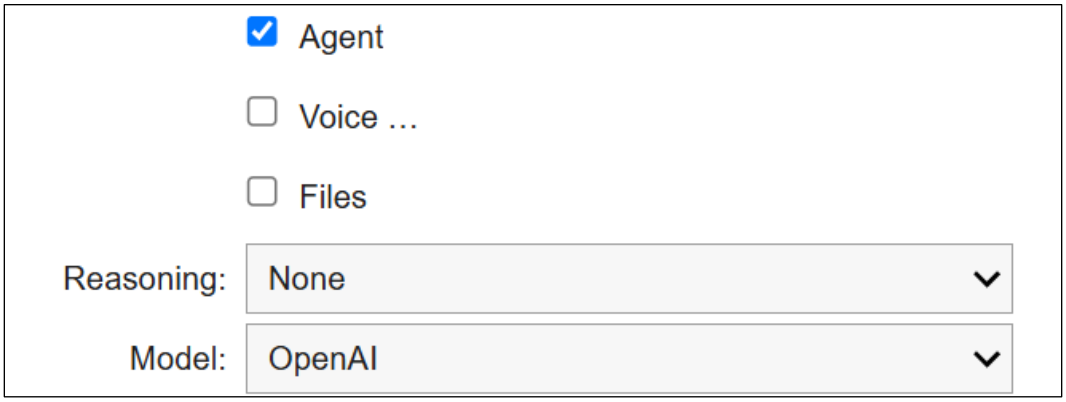

Data flow and component interaction in GenAISys
1. IPython Interface Input & Output
2. Handler Mechanism
3. Handler Registry
4. AI Functions

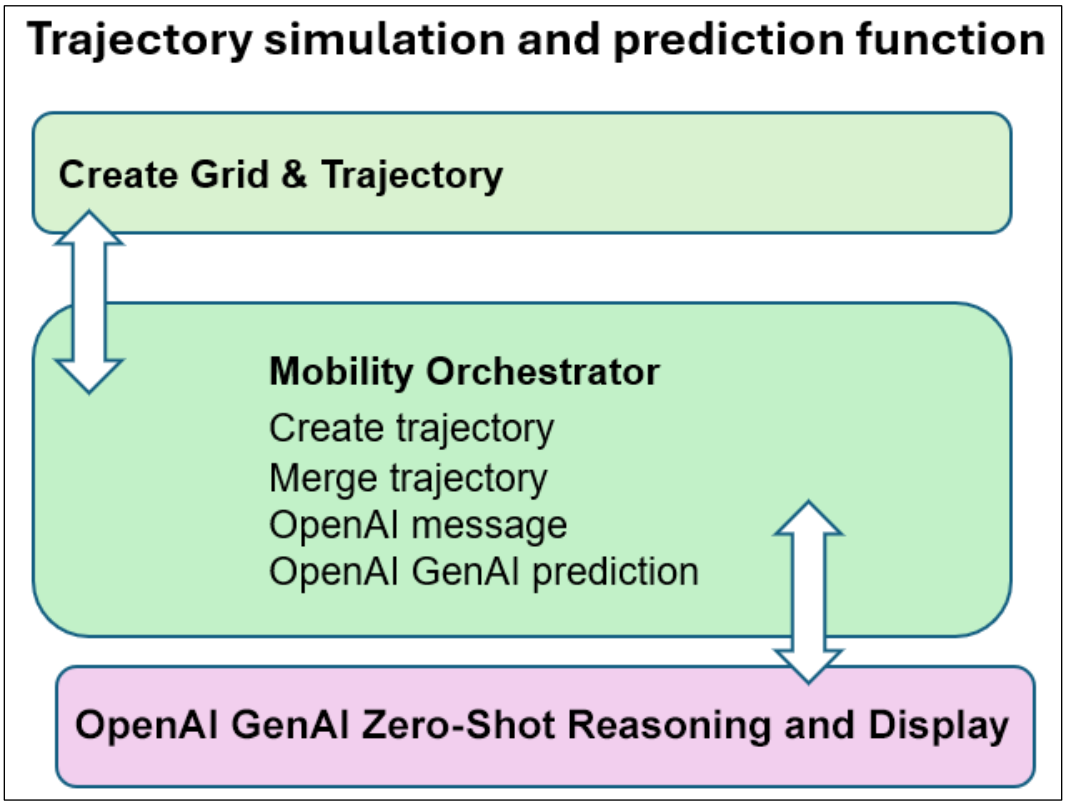

Data flow and component interaction in GenAISys
1. IPython Interface Input & Output
2. Handler Mechanism
3. Handler Registry
4. AI Functions

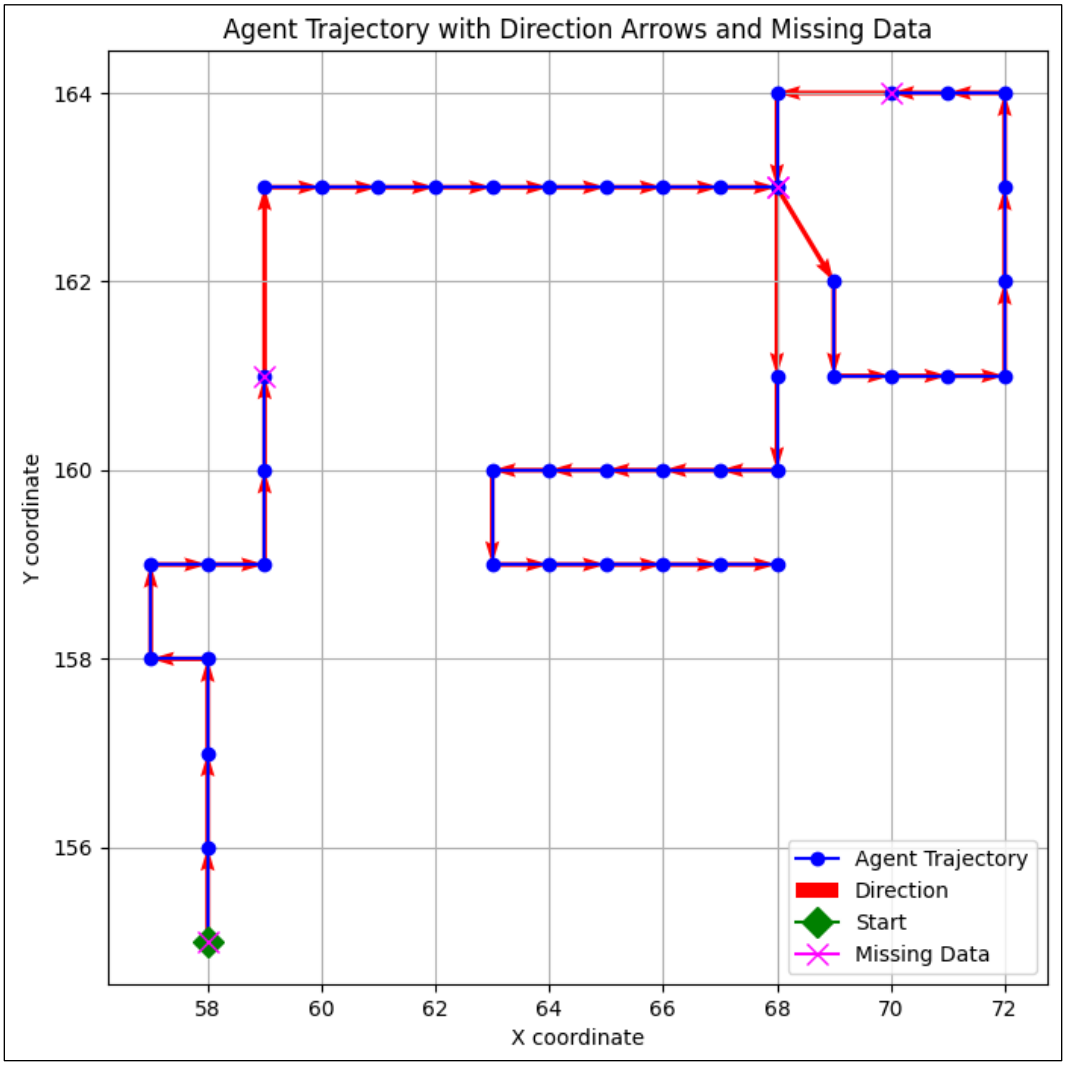

Data flow and component interaction in GenAISys
1. IPython Interface Input & Output
2. Handler Mechanism
3. Handler Registry
4. AI Functions

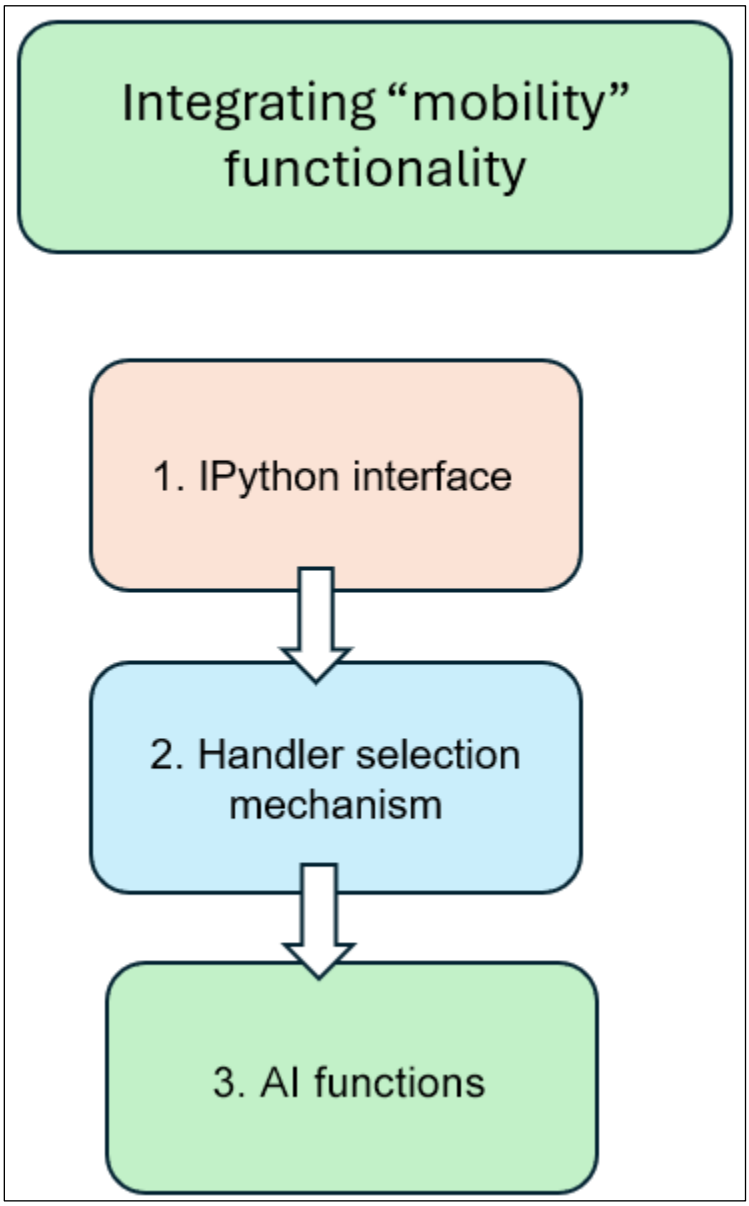
Chapter 8: GenAISys for Trajectory Simulation and Prediction

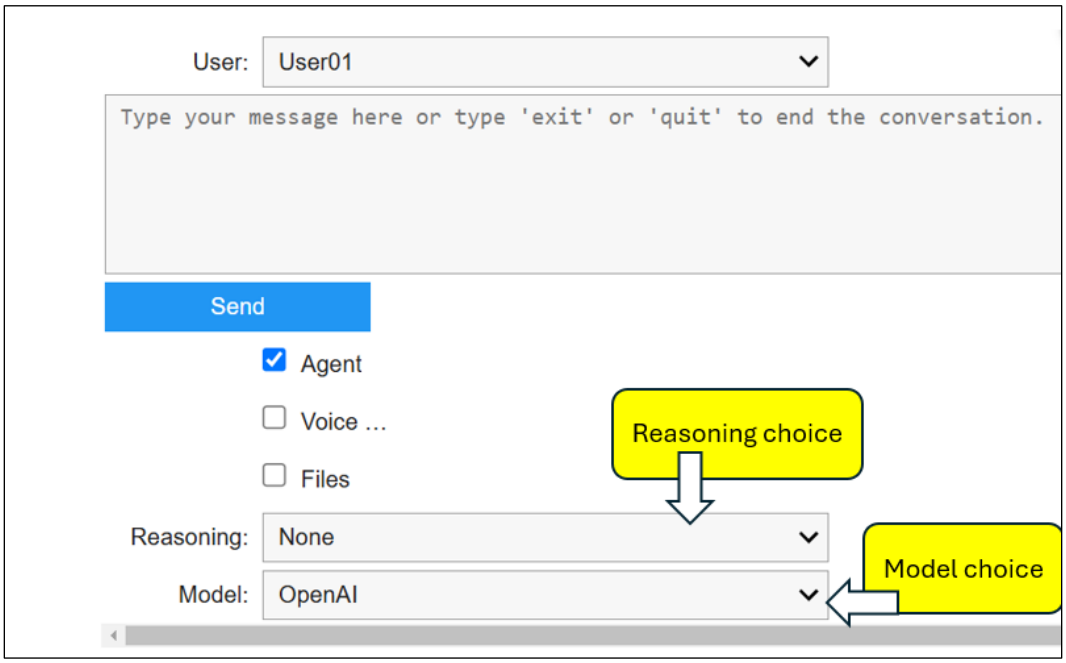


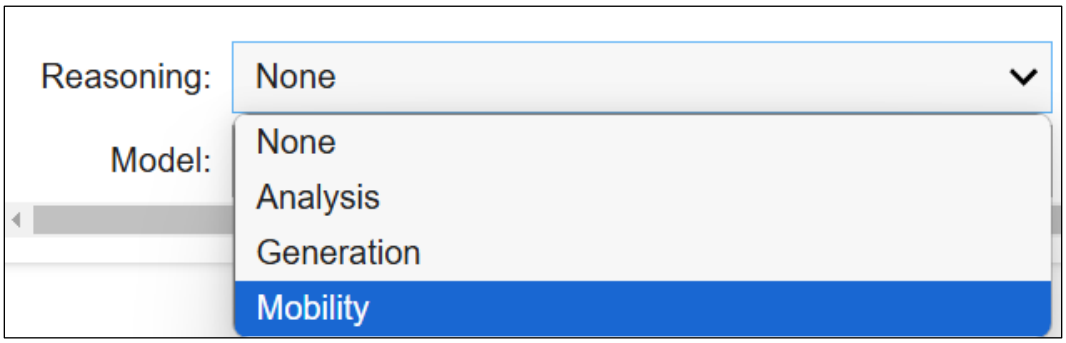


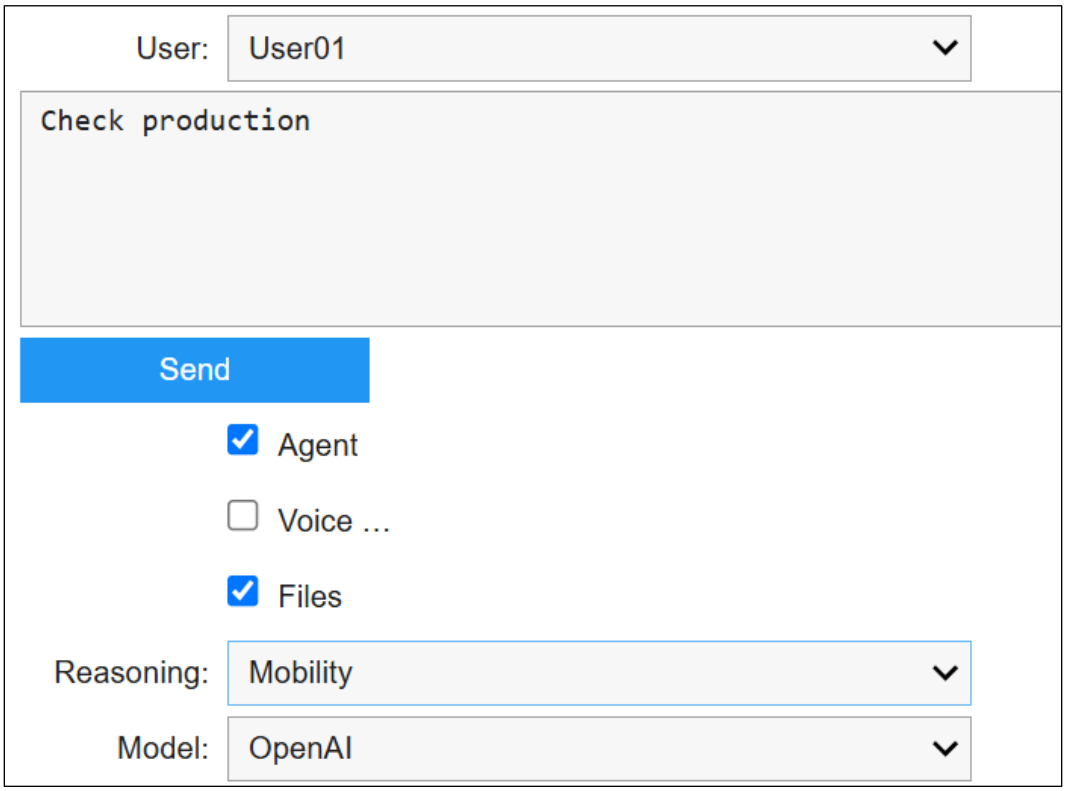

Trajectory simulation and prediction function
Create Grid & Trajectory
Mobility Orchestrator
Create trajectory
Merge trajectory
OpenAI message
OpenAI GenAI prediction
OpenAI GenAI Zero-Shot Reasoning and Display

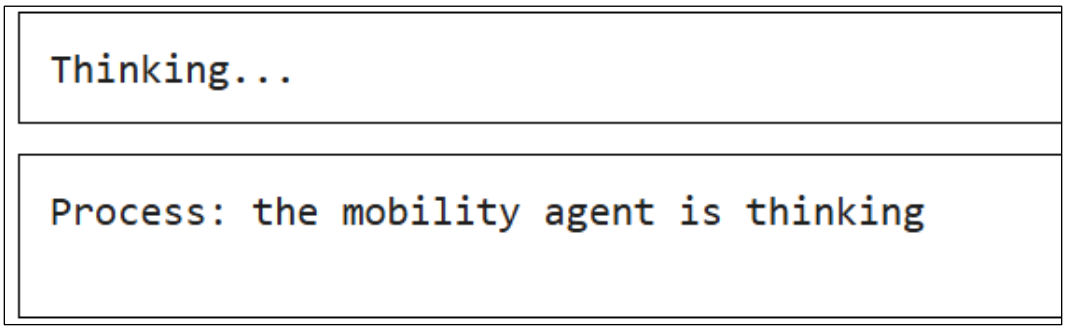

Agent Trajectory with Direction Arrows and Missing Data
Y coordinate
X coordinate
164
162
160
158
156
58
60
62
64
66
68
70
72
Agent Trajectory
Direction
Start
Missing Data

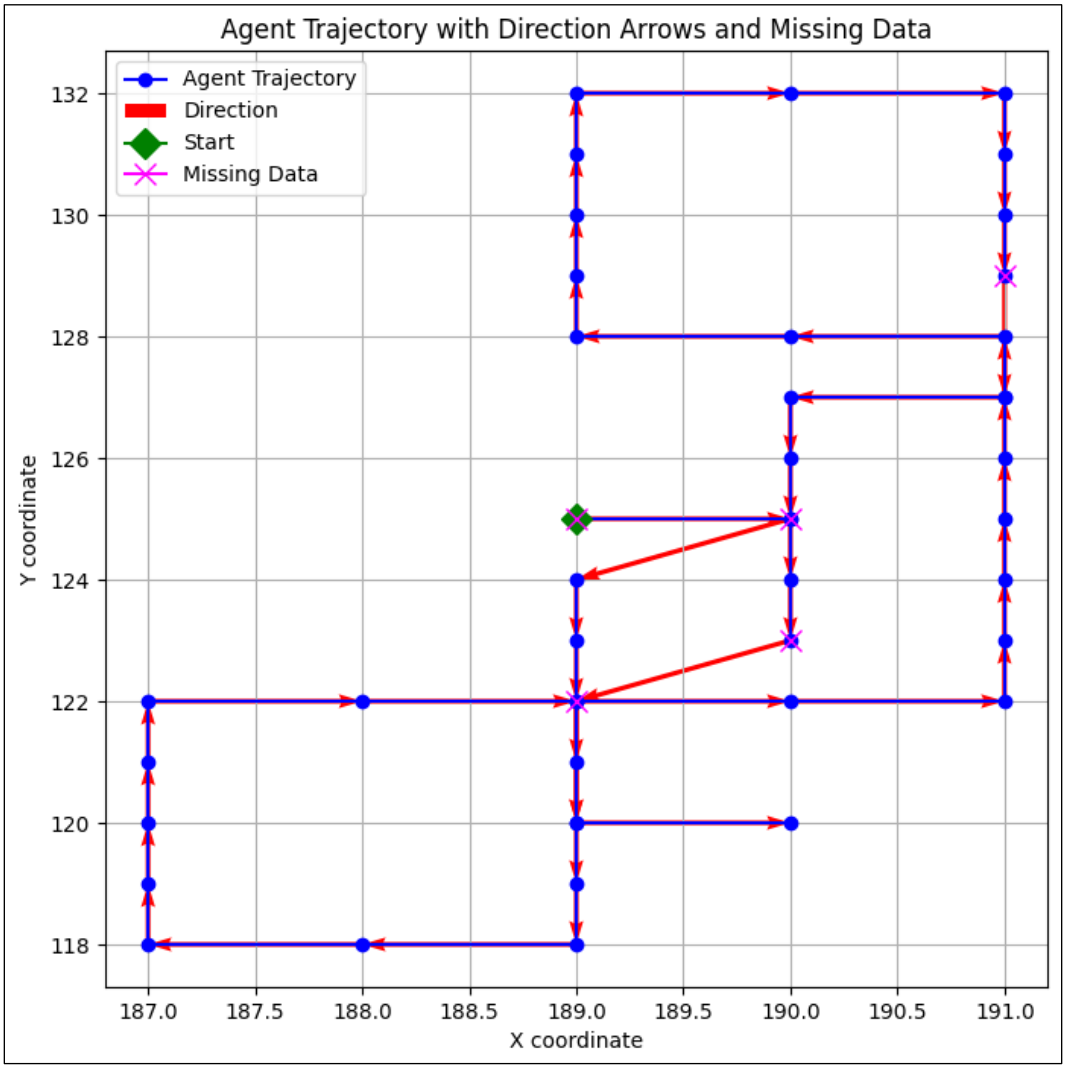

Integrating “mobility” functionality
1. IPython interface
2. Handler selection mechanism
3. AI functions


User:
User01
Type your message here or type 'exit' or 'quit' to end the conversation.
Send
Agent
Voice ...
Files
Reasoning choice
Reasoning:
None
Model:
OpenAI
Model choice

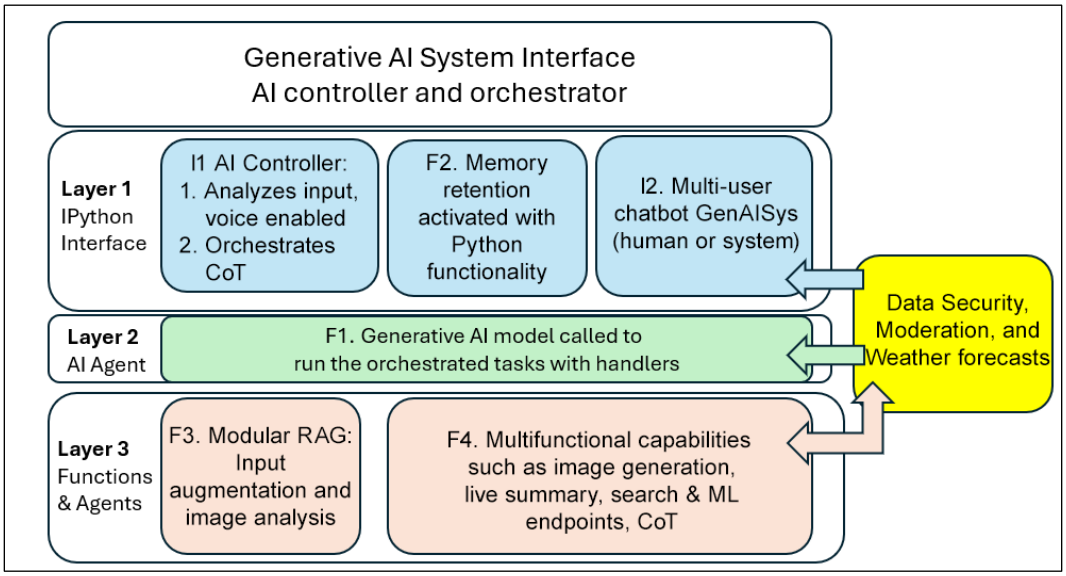

Reasoning:
None
Model:
None
Analysis
Generation
Mobility

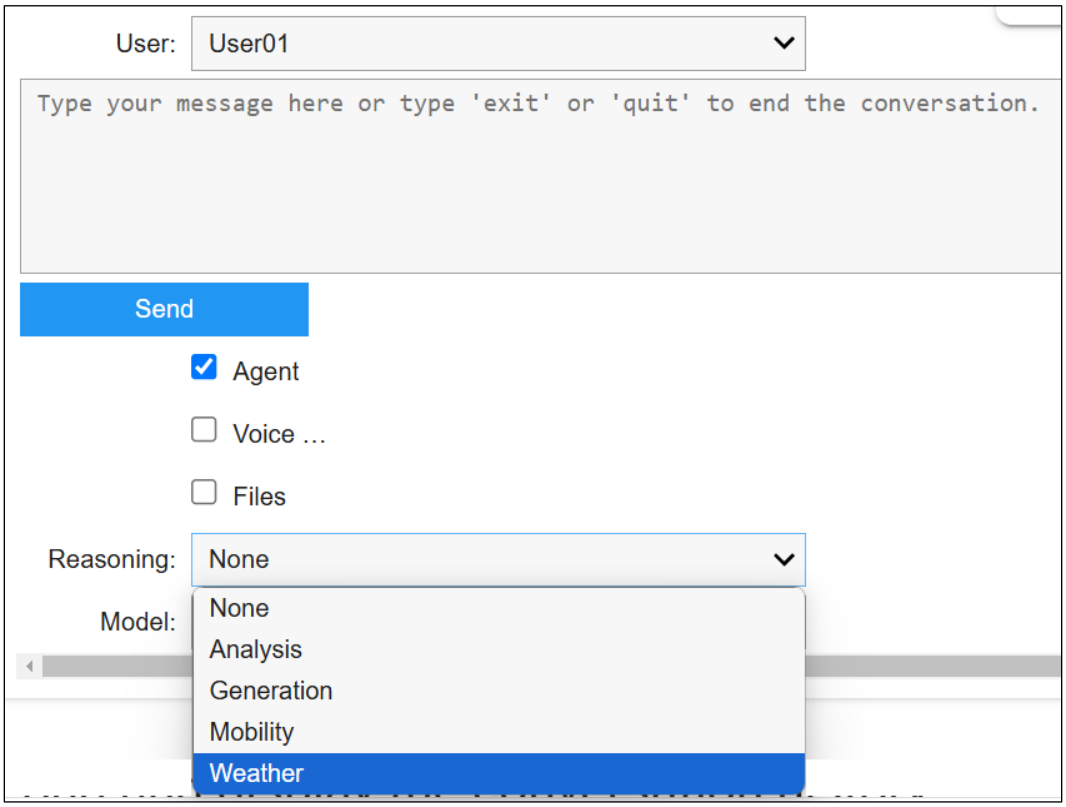

User:
User01
Check production
Send
Agent
Voice …
Files
Reasoning:
Mobility
Model:
OpenAI

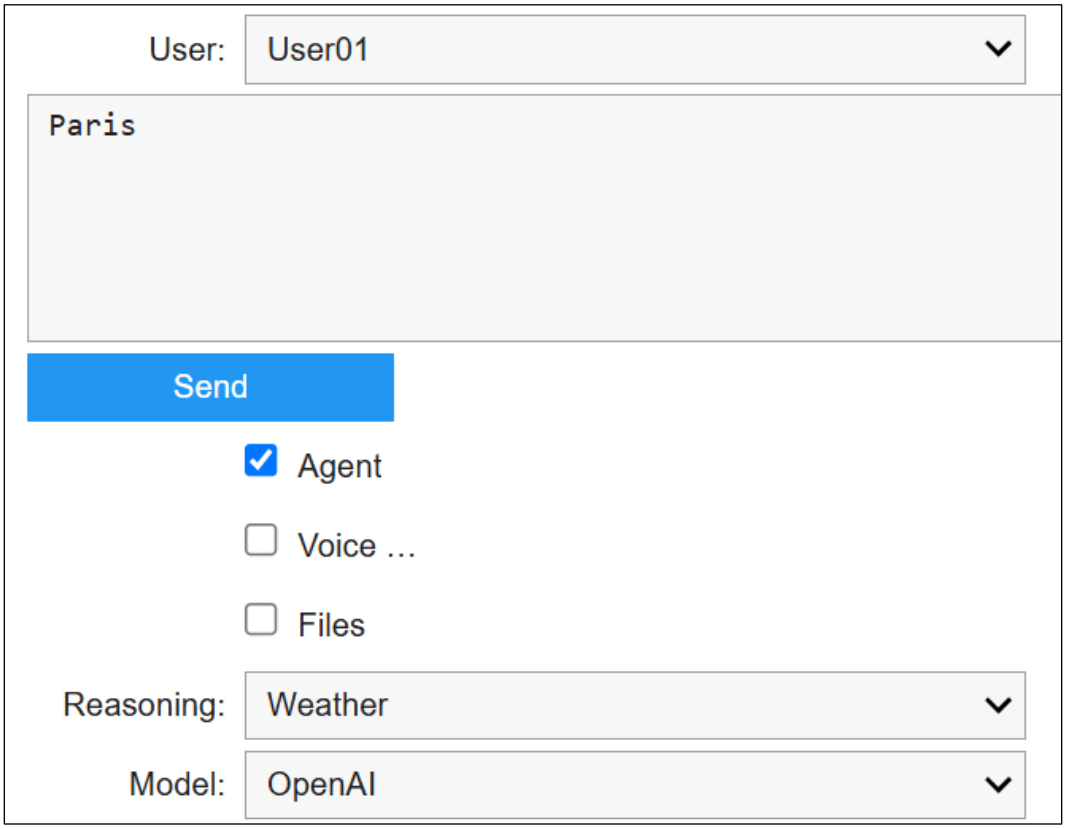

Thinking...
Process: the mobility agent is thinking

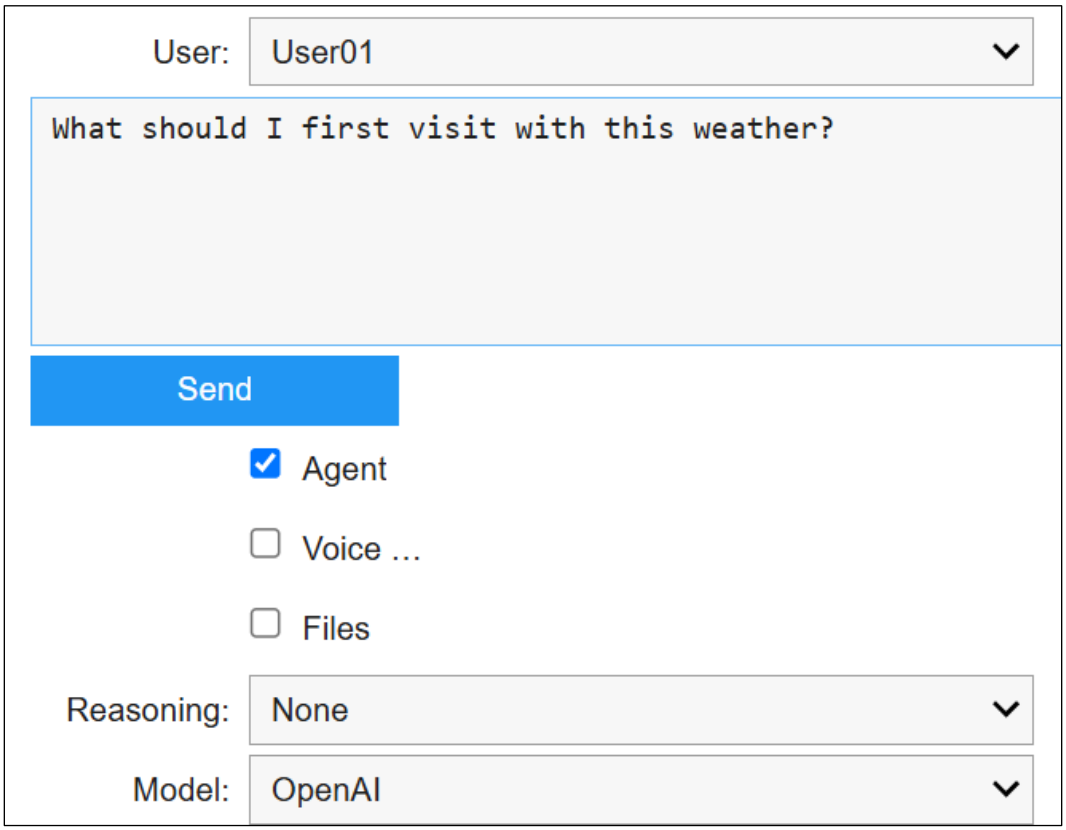

Agent Trajectory with Direction Arrows and Missing Data
Agent Trajectory
Direction
Start
Missing Data
Y coordinate
X coordinate
132
130
128
126
124
122
120
118
187.0
187.5
188.0
188.5
189.0
189.5
190.0
190.5
191.0

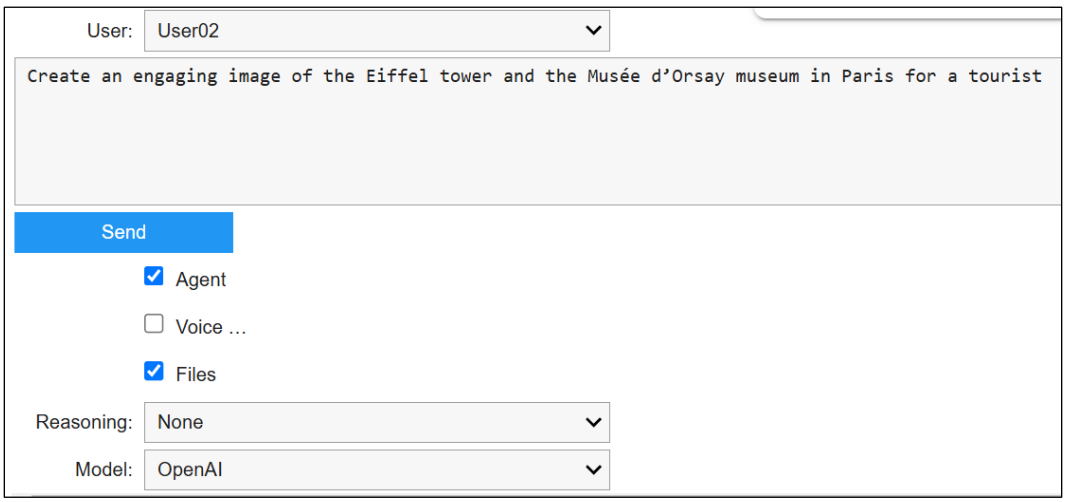
Chapter 9: Upgrading the GenAISys with Data Security and Moderation for Customer Service



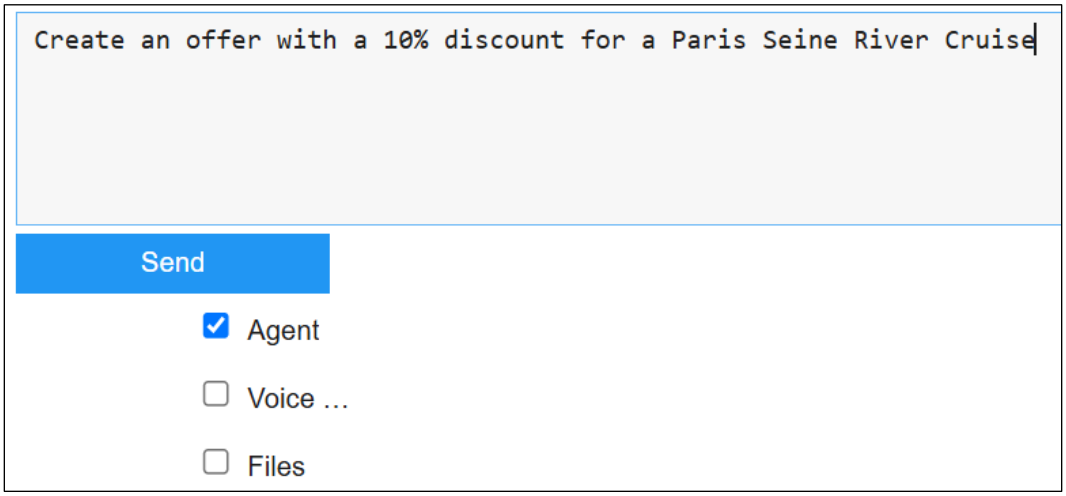

User:
User01
Type your message here or type 'exit' or 'quit' to end the conversation.
Send
Agent
Voice ...
Files
Reasoning:
None
Model:
None
Analysis
Generation
Mobility
Weather

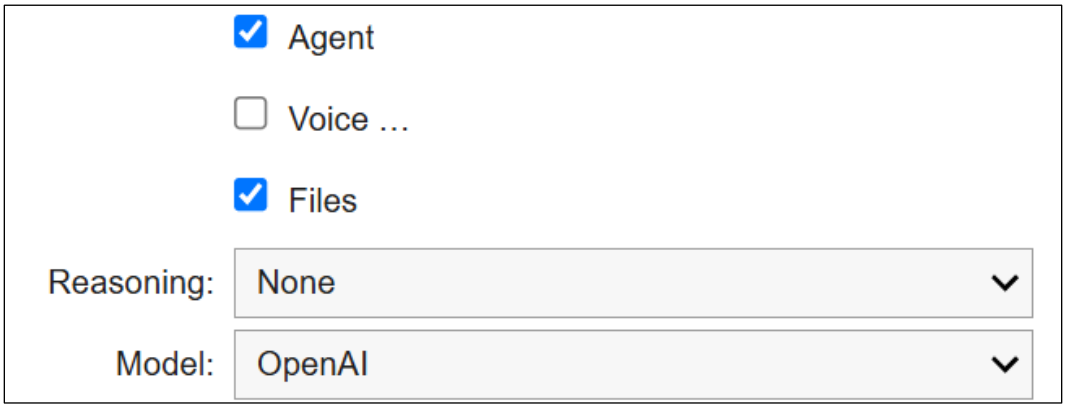

User:
User01
Paris
Send
Agent
Voice …
Files
Reasoning:
Weather
Model:
OpenAI


User:
User01
What should I first visit with this weather?
Send
Agent
Voice ...
Files
Reasoning:
None
Model:
OpenAI


User:
User02
Create an engaging image of the Eiffel tower and the Musée d'Orsay museum in Paris for a tourist
Send
Agent
Voice ...
Files
Reasoning:
None
Model:
OpenAI




Create an offer with a 10% discount for a Paris Seine River Cruise
Send
Agent
Voice …
Files


Agent
Voice …
Files
Reasoning:
None
Model:
OpenAI



Send

- [x] Agent
- [ ] Voice …
- [ ] Files

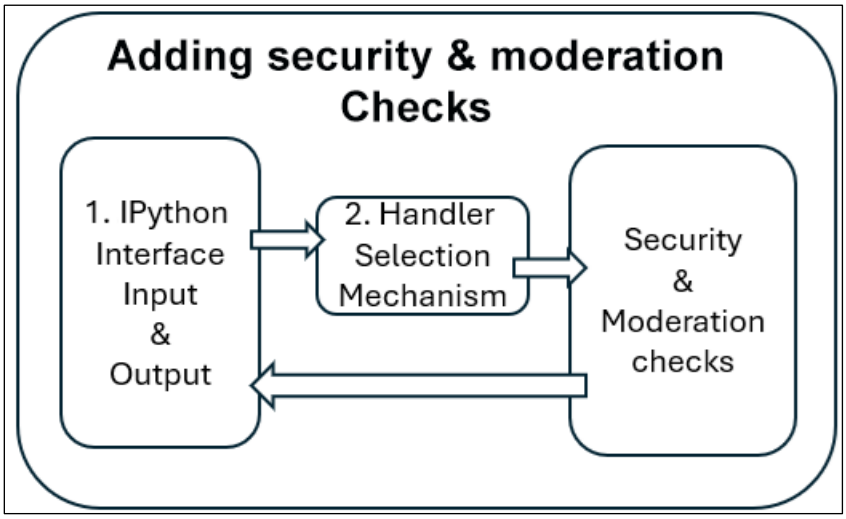
Agent

Voice …

Files

Reasoning: Weather

Model: OpenAI



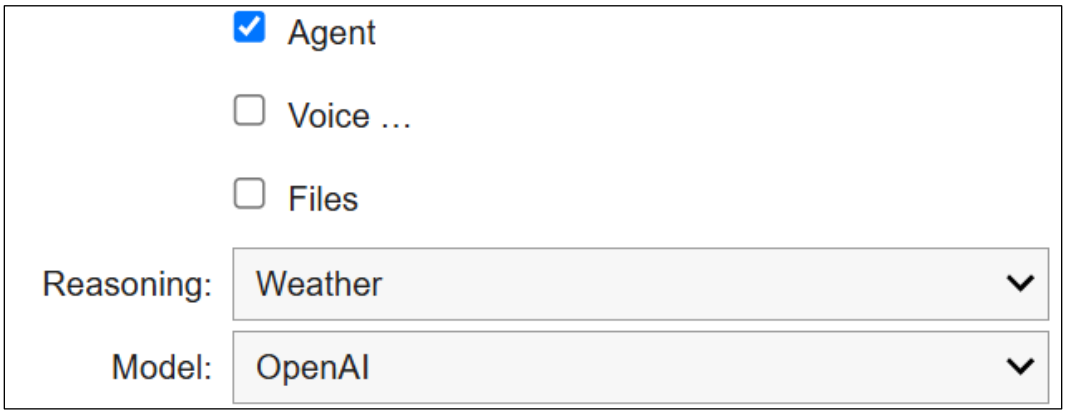
Agent

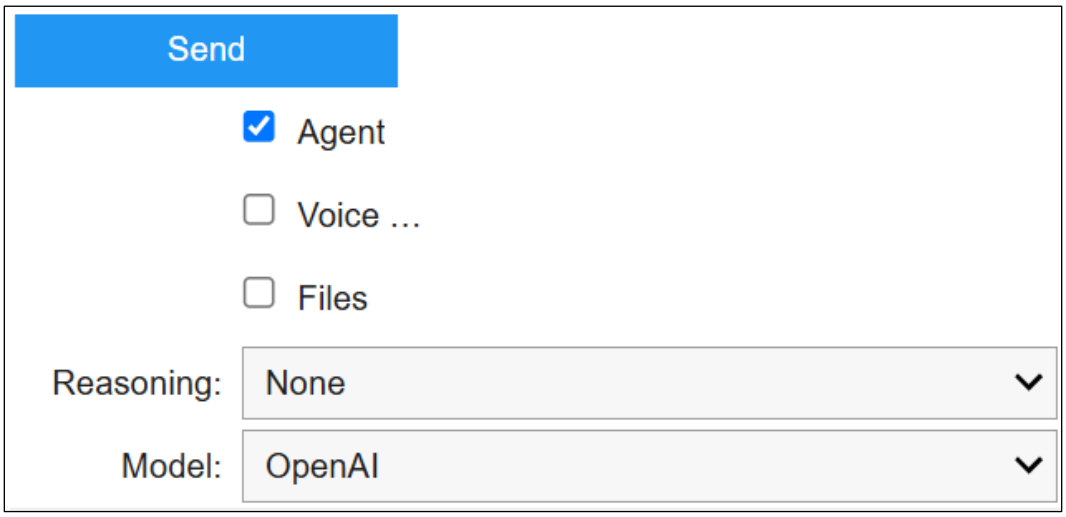
Voice …

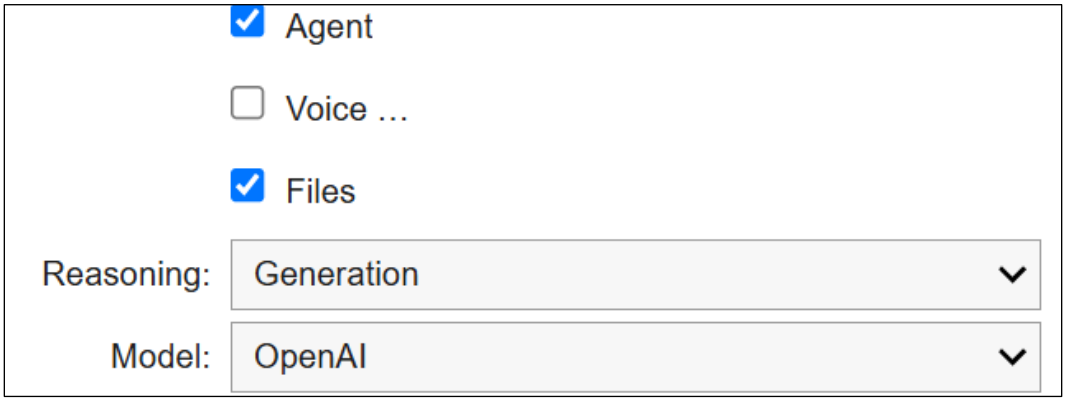
Files

Reasoning: None

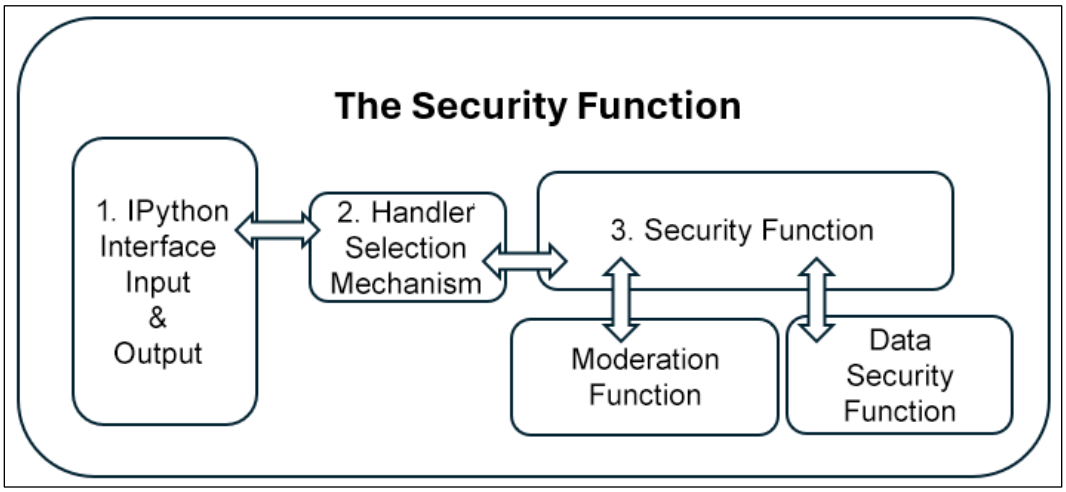
Model: OpenAI

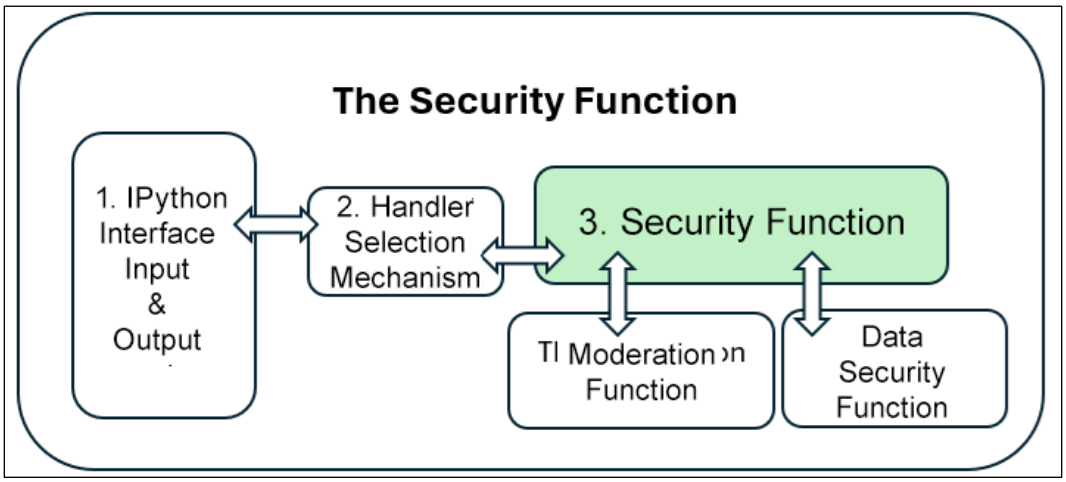

Agent
Voice …
Files
Reasoning:
Weather
Model:
OpenAI

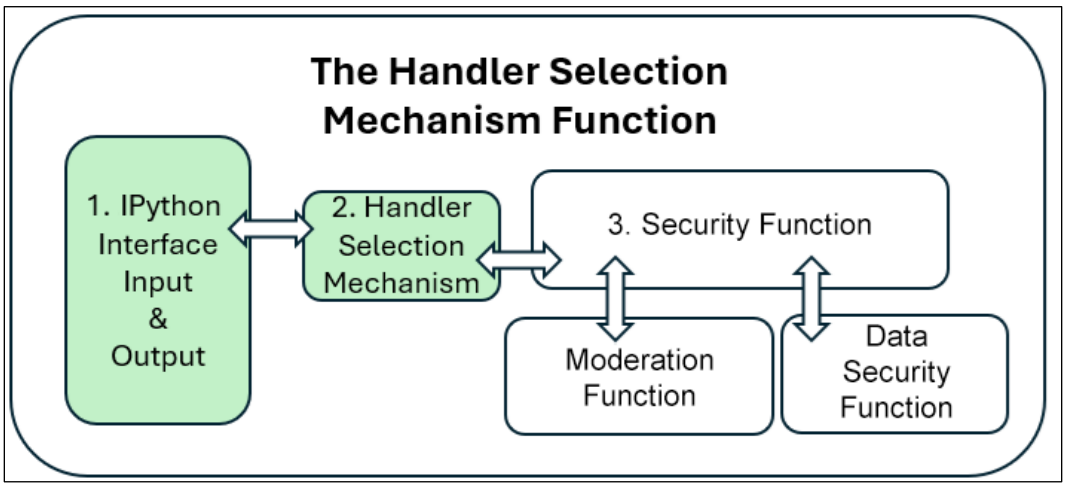

Send
Agent
Voice …
Files
Reasoning:
None
Model:
OpenAI

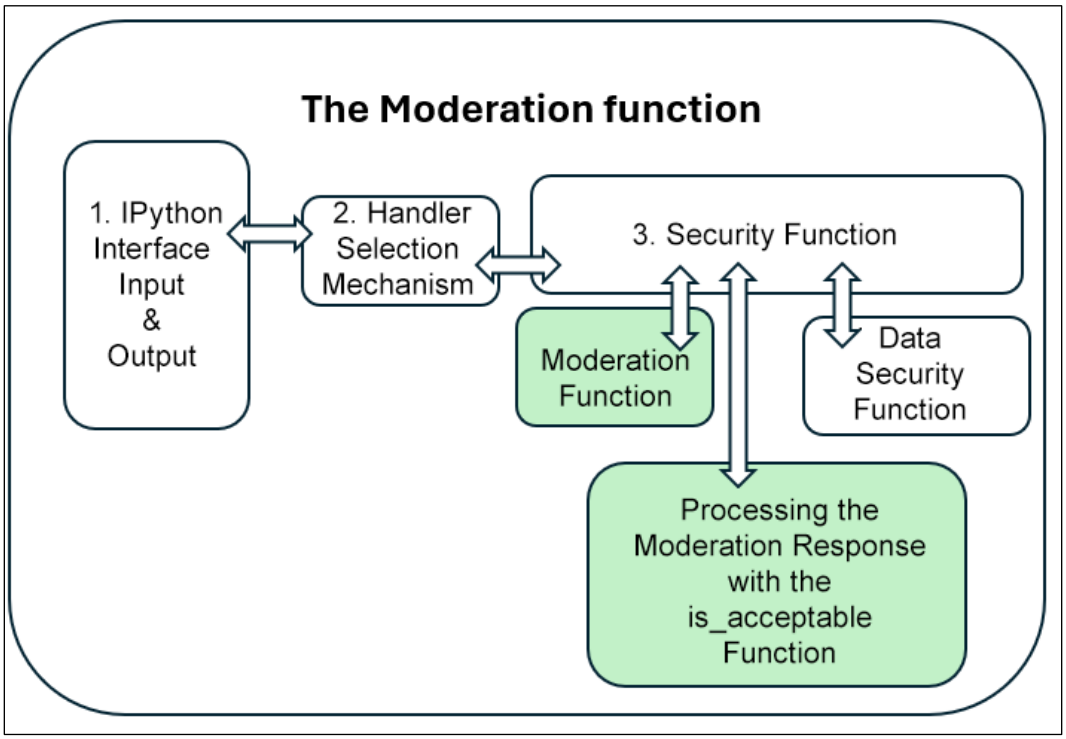

Agent
Voice …
Files
Reasoning:
Generation
Model:
OpenAI

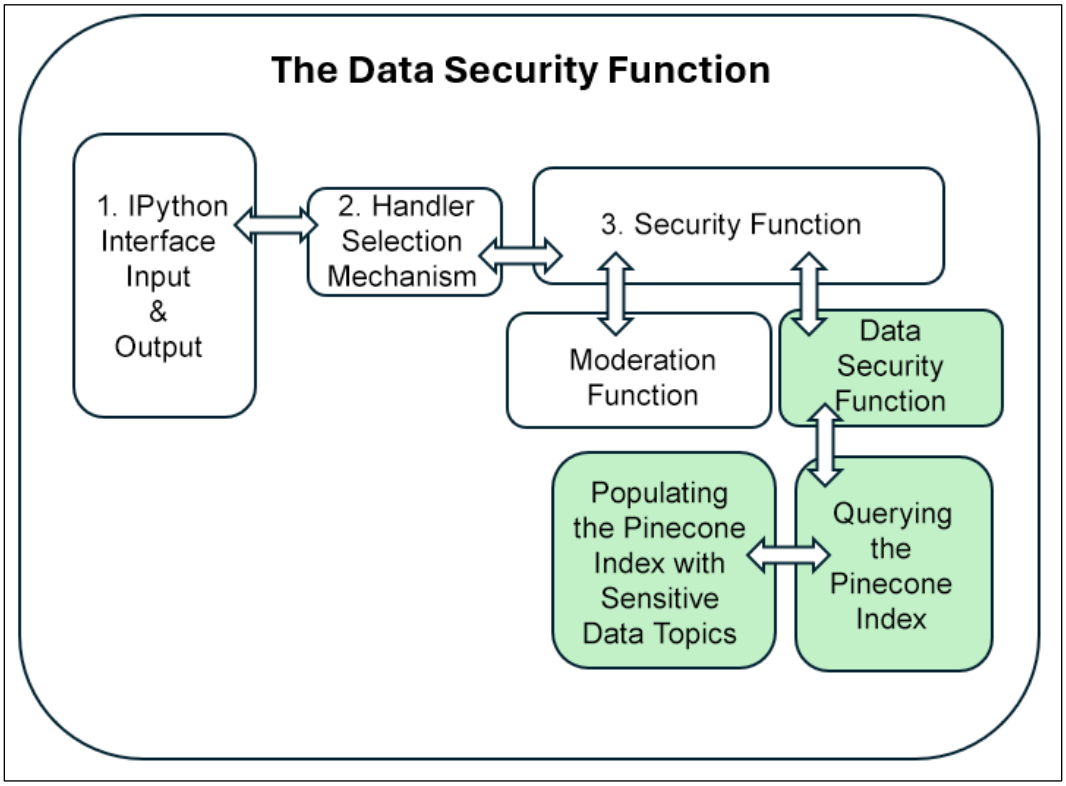

A IAMAGINE A TIMST
NEW YORK


The Security Function
1. IPython Interface Input & Output
2. Handler Selection Mechanism
3. Security Function
Moderation Function
Data Security Function


The Security Function
1. IPython Interface Input & Output
2. Handler Selection Mechanism
3. Security Function
Moderation Function
Data Security Function


The Handler Selection Mechanism Function
1. IPython Interface Input & Output
2. Handler Selection Mechanism
3. Security Function
Moderation Function
Data Security Function

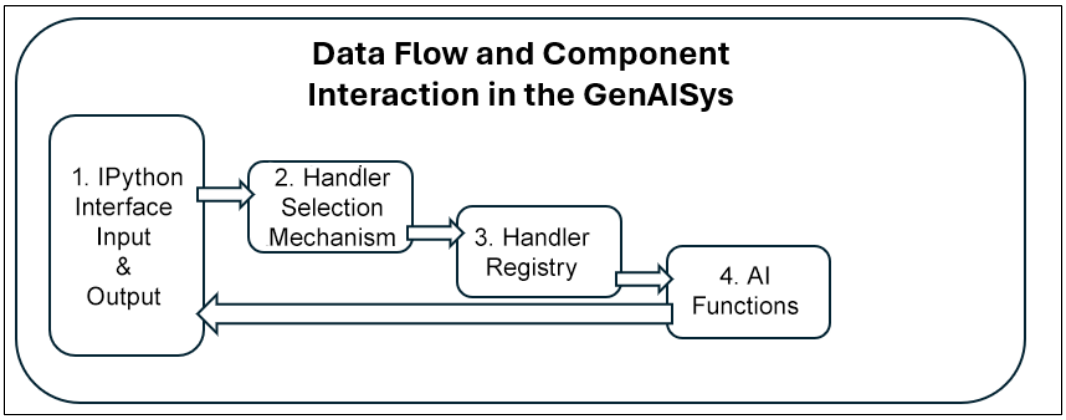

The Moderation function
1. IPython Interface Input & Output
2. Handler Selection Mechanism
3. Security Function
Moderation Function
Data Security Function
Processing the Moderation Response with the is_acceptable Function



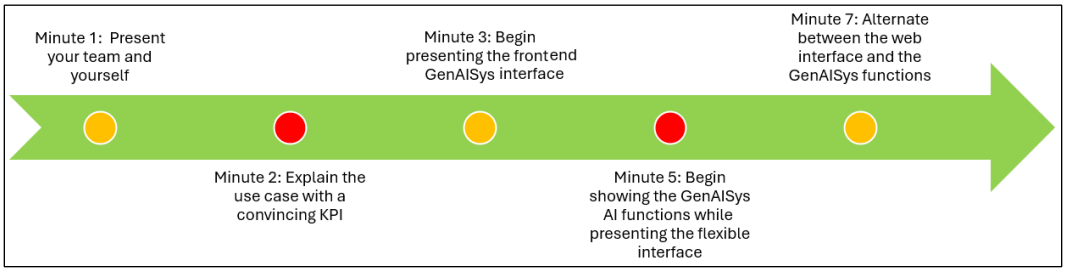
User: User01

Your ideas are always foolish and contribute nothing to our discussions.

Send


Data Flow and Component Interaction in the GenAISys
1. IPython Interface Input & Output
2. Handler Selection Mechanism
3. Handler Registry
4. AI Functions

Chapter 10: Presenting Your Business-Ready Generative AI System





Generative AI Chat Interface

Connected to Pinecone Index

Gross Margin

$540.63

User:

User01

Your Message:

Type your message here or type 'exit' or 'quit' to end the conversation.

Send

Agent Voice Output Files

Reasoning:

None

Model:

OpenAI

Conversation output will appear here...

Context Retrieved from Pinecone

No results yet.

Reasoning:

None

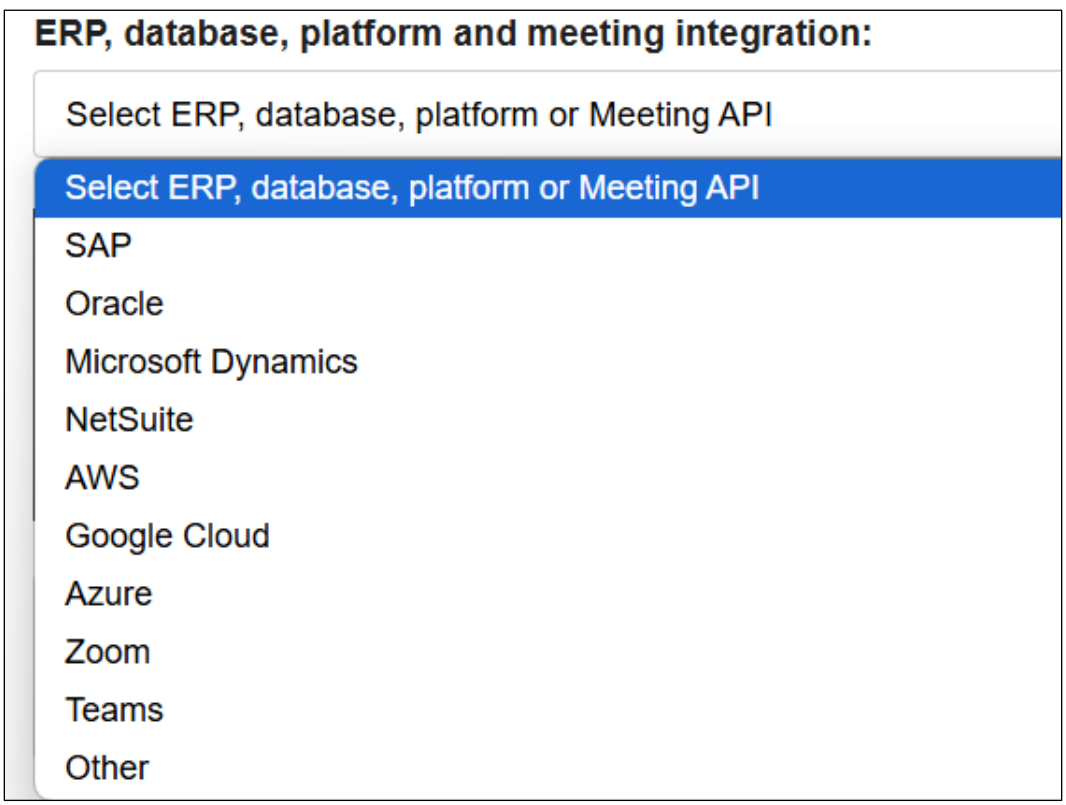
Model:

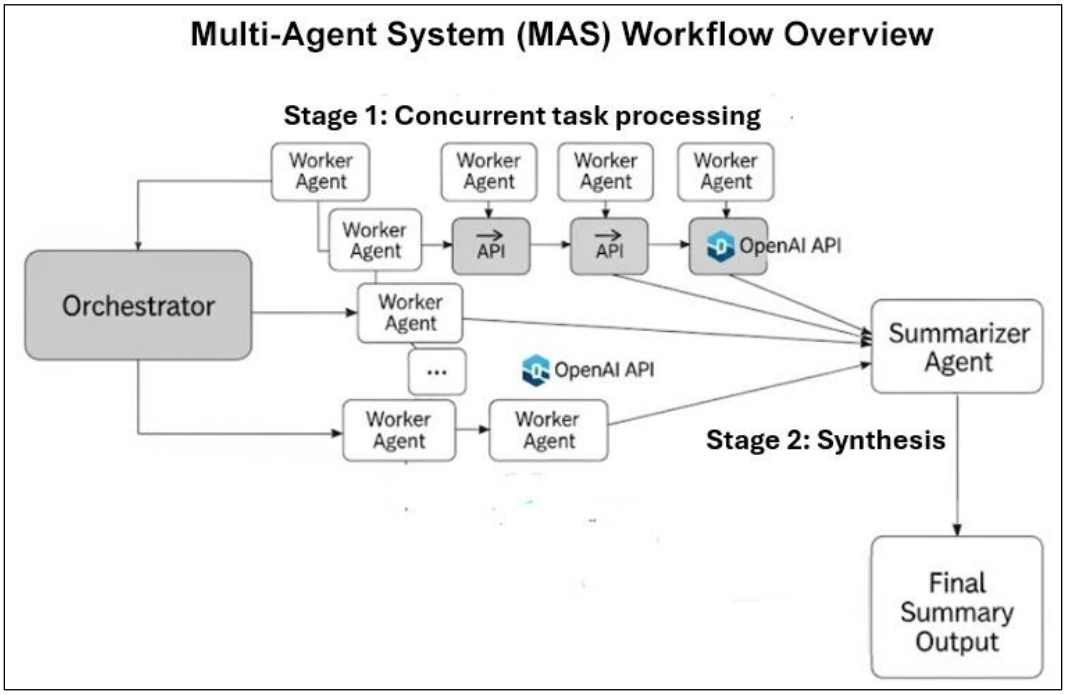
OpenAI

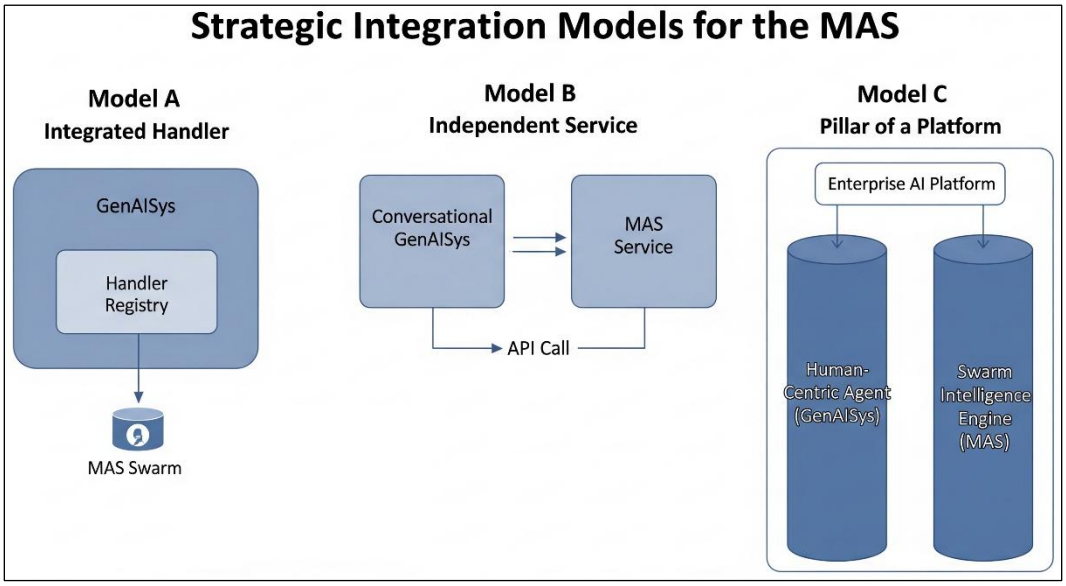
ERP, database, platform and meeting integration:

Select ERP, database, platform or Meeting API

Conversation output will appear here...




Multi-Agent System (MAS) Workflow Overview
Stage 1: Concurrent task processing
Worker Agent
Worker Agent
Worker Agent
Worker Agent
Worker Agent
API
API
OpenAI API
Orchestrator
Worker Agent
Summarizer Agent
...
OpenAI API
Worker Agent
Worker Agent
Stage 2: Synthesis
Final Summary Output


Strategic Integration Models for the MAS
Model A
Integrated Handler
GenAISys
Handler Registry
MAS Swarm
Model B
Independent Service
Conversational GenAISys
MAS Service
API Call
Model C
Pillar of a Platform
Enterprise AI Platform
Human-Centric Agent (GenAISys)
Swarm Intelligence Engine (MAS)

Generative AI Chat Interface

Security Alert: Suspicious activity detected.

Three-Phase Customization Process

- Phase 1 involves brainstorming workshops for frontend HTML . customization.
- Phase 2 focuses on developing an isolated sandbox for feature additions.
- Phase 3 integrates sandbox evolutions into the HTML page progressively.
- The process is iterative, allowing for continuous improvement.
- All phases work in cycles to enhance collaboration and outcomes.



User:

User01

Your Message:

Type your message here or type 'exit' or 'quit' to end the conversation.

Send

☑ Agent ☐ Voice Output ☐ Files

Reasoning:

On-site or remote parallel processing with three persons (or teams)

- **Phase 1**: Customizing the frontend HTML page through brainstorming workshops with as many groups as necessary.



- **Phase 2**: Creating an isolated sandbox to add features to the GenAISys you built progressively.
- **Phase 3**: Integrating the sandbox evolutions step by step in the HTML page (documentation and stabilization).
- **Leverage everything you need from this book during the customization process. Also, add your ideas and found technologies.**

Generative AI Chat Interface

● Connected to Pinecone Index

Gross Margin

$577.78

7.RACI Matrix for GenAISys Implementation

Task
1. Requirements & Planning Define project scope for integrating DeepSeek, OpenAI, Pinecone, and the interactive interface.
2. Environment Setup & File Downloading Implement the initial file download script (e.g., fetching `grequests.py` via curl) and set up the basic Python environment.
3. DeepSeek Activation & Setup Configure DeepSeek as shown in the “DeepSeek Activation Guide” and set the appropriate flags (`deepseek=True` , `hf=True` , etc.).
4. GPU & Cache Configuration Mount Google Drive, check GPU status (via `nvidia-smi`), and set cache paths for Hugging Face as detailed in the notebook.
5. Hugging Face Environment Installation & Model Loading Install transformers, check version, and load the DeepSeek model using `AutoTokenizer` and `AutoModelForCausalLM` .
6. OpenAI Integration Run OpenAI installation scripts, initialize the API key (via `openai_setup.py`), and import the API call functions from `reason.py` .

Generative AI Chat Interface

User:

User01

Your Message:

Type your message here or type 'exit' or 'quit' to end the conversation.

Send

Agent Voice Output Files

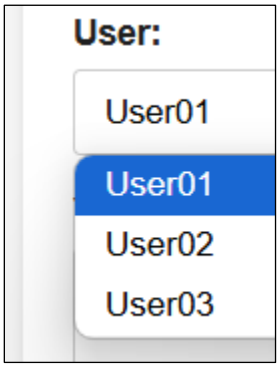
Reasoning:

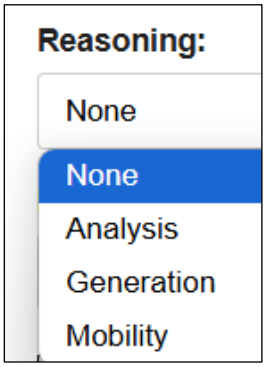
None

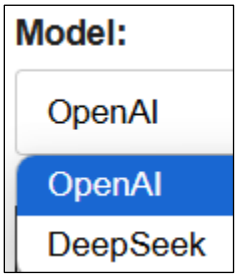
Model:

OpenAI

Conversation output will appear here...




Reasoning:
None
None
Analysis
Generation
Mobility


Model:
OpenAI
OpenAI
DeepSeek

Generative AI Chat Interface

● Connected to Pinecone Index

User:

User03

Your Message:

Type your message here or type 'exit' or 'quit' to end the conversation.

Send

☑ Agent ☐ Voice Output ☐ Files

Reasoning:

None

Model:

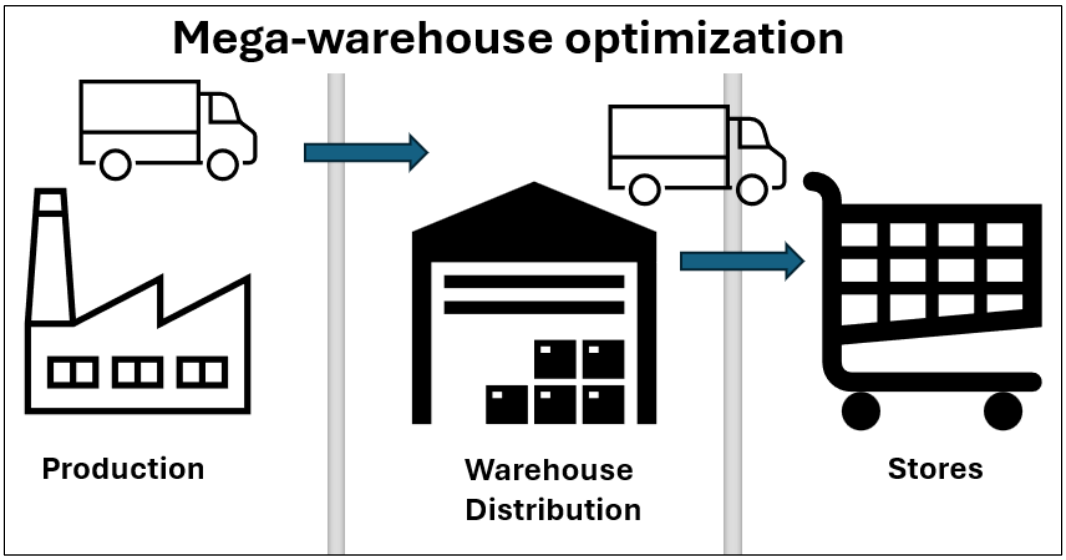
OpenAI

Conversation output will appear here...

Context Retrieved from Pinecone

No results yet.


SALES
GROWTH
FAST DELIVERY
TRAVEL
TRAVEL
TRAVEL
RANTH
FAST
DELIVERY



$$\text{ROI Scenario1} = \frac{\text{Layoff savings}}{\text{GenAISys project cost}}$$

$$\text{ROI Scenario 2: Growth} > \text{GenAISys project cost}$$