WORKSHOP

GenAI Bootcamp

Agenda

Day 1 Feb 04th

- 09:30 **Intro**
  GenAI & Bedrock
- 11:30 **Bedrock API**
  + Multimodality
- 13:30 **RAG**
  Knowledge Bases
- 15:00 **Bedrock Data Automation + QuickSuite**
- 18:00 **Afterwork**

Day 2 Feb 05th

- 09:30 **Agents**
  Tools, MCP
- 11:30 **Strands Agents**
  + Multi agents
- 13:30 **AI Security**
  Secure chatbots
- 15:00 **AgentCore**
  Introduction
- 16:30 **Workshop sessions**
  Detailed scoping & Architecture

Day 3 Feb 06th

- 09:30 **AgentCore + Eval**
  Deep dive
- 11:30 **AI coding**
  Kiro, Spec Driven Dev
- 13:30 **Closing session**
  Tech FR lead keynote
- 13:30 **Workshop sessions**
  + Next steps

Build Phase

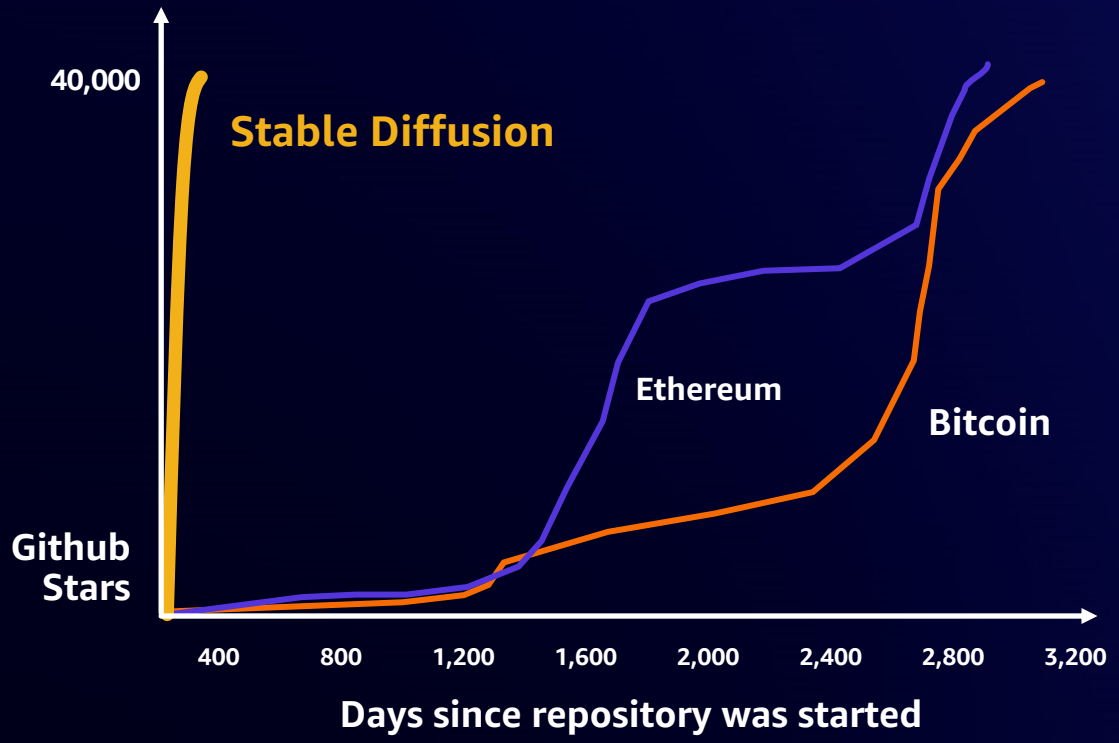
Feb 09th – Feb 20th

Intro to Generative AI (GenAI) and Amazon Bedrock

Generative AI is the fastest growing trend in AI

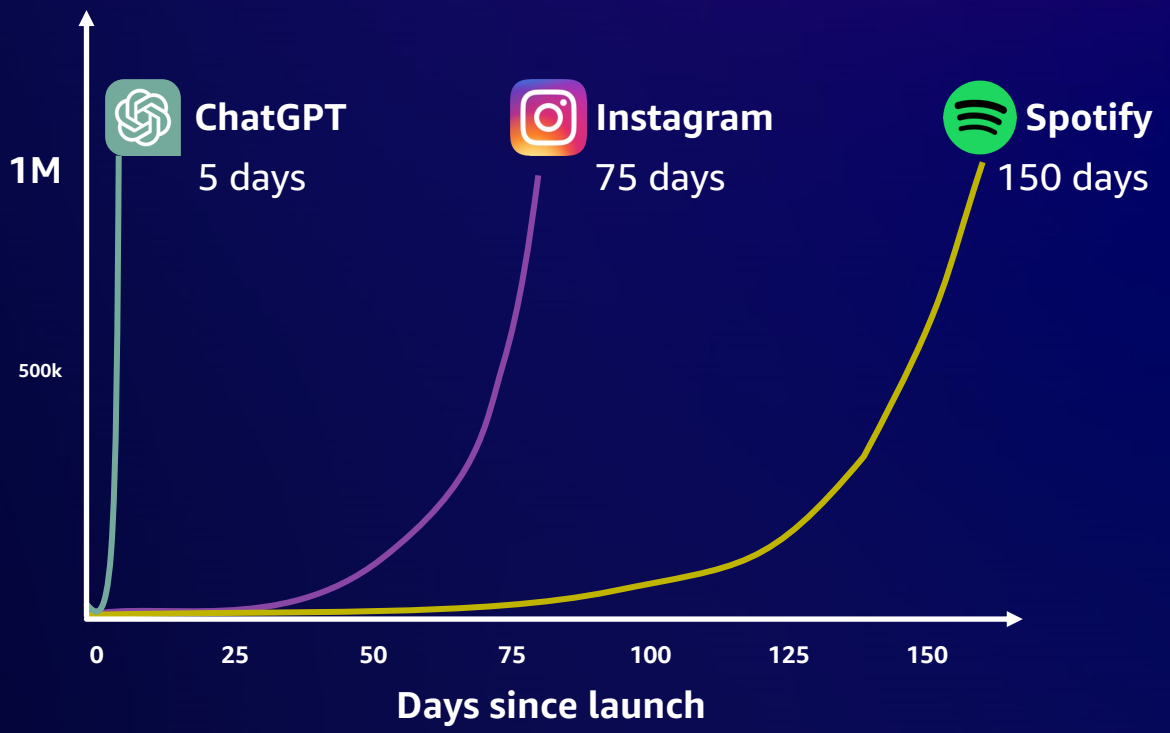
Developer adoption

Stable Diffusion accumulated 40k stars on GitHub in its first 90 days

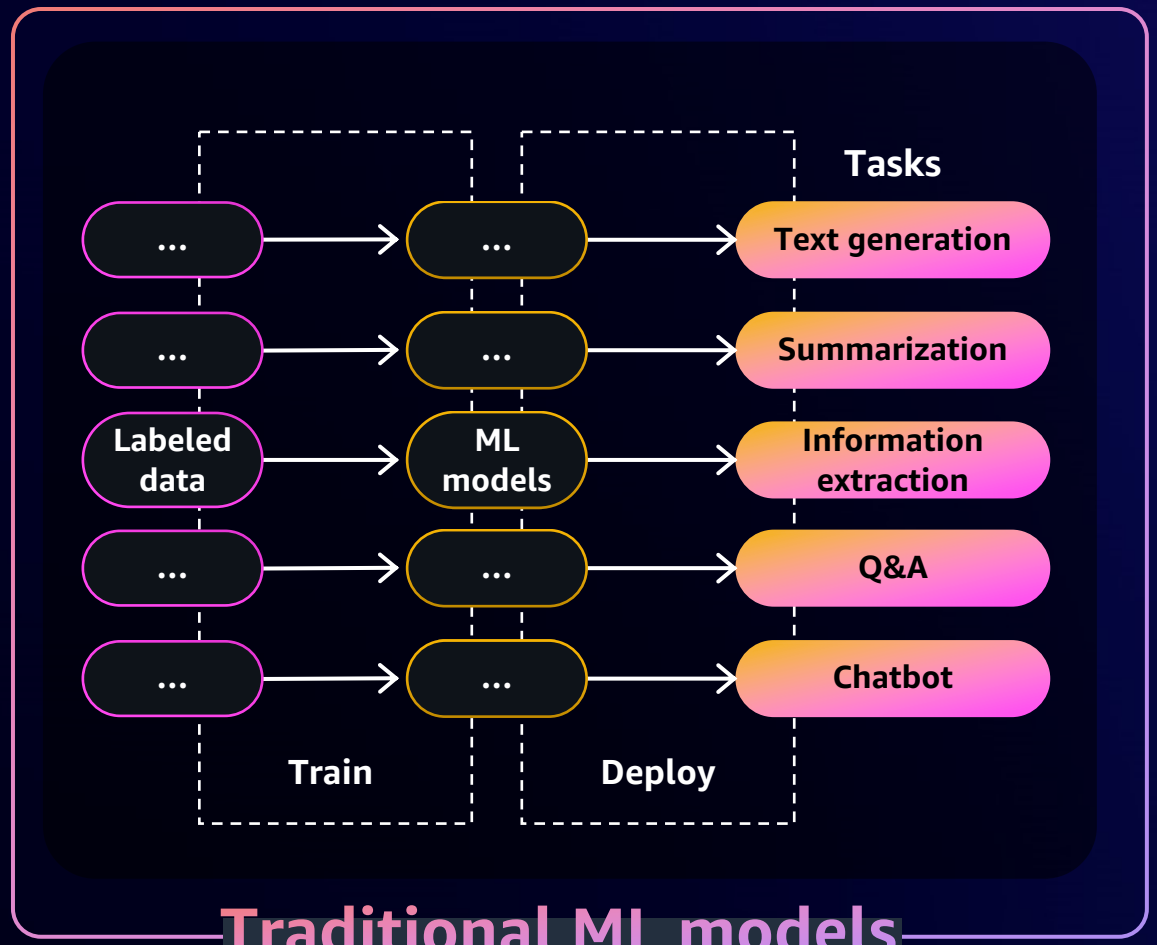


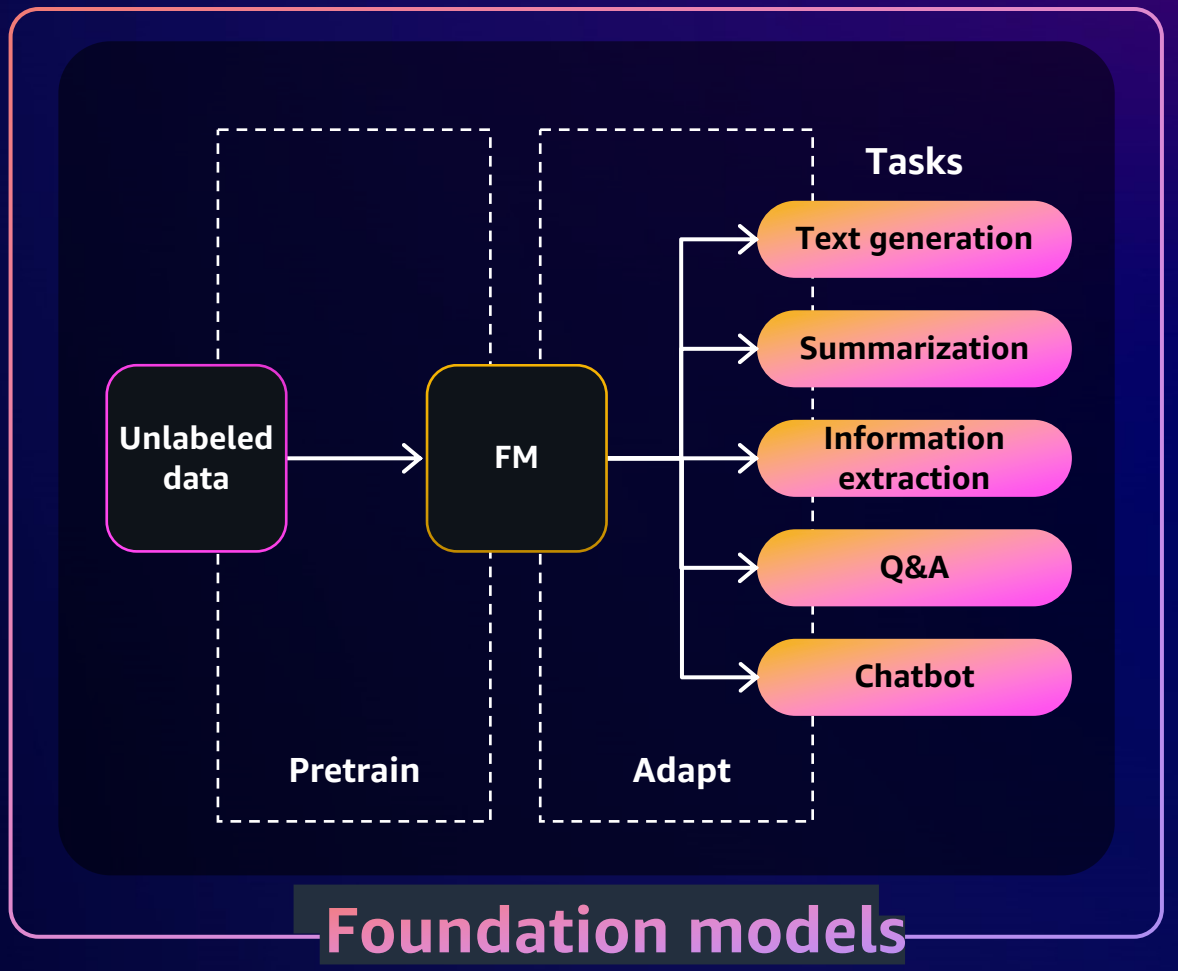
Consumer adoption

ChatGPT reached the 1 million users mark in just 5 days



Why foundation models?

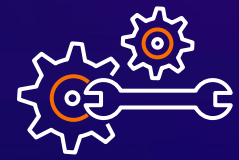




What is Generative AI?

AI that can produce original content close enough to human generated content for real-world tasks

Powered by foundation models pre-trained on large sets of data



Can complete wide range of tasks without additional training

Applicable to many use cases like text summarization, question answering, digital art creation, code generation, etc.

Reduces time and cost to develop ML systems and innovate faster

Generative AI helps across a range of domains …

Enhance customer experience

CHATBOTS

VIRTUAL ASSISTANTS

AI-POWERED CONTACT CENTER

PERSONALIZATION

Boost employee productivity

CONVERSATIONAL SEARCH

SUMMARIZATION

ENTITY EXTRACTION

CODE GENERATION

DATA TO INSIGHTS

Creativity and content creation

WRITING

MEDIA

DESIGN

MODELING

Improve business operations

DOCUMENT PROCESSING

PROCESS OPTIMIZATION

CYBERSECURITY

DATA AUGMENTATION

What is an LLM?

A Large Language Model is a massive, deep learning AI model pretrained on vast amounts of text.

It understands text; extracts meanings; and learns grammar, languages, and knowledge.

Building generative AI applications can be challenging

Accessing multiple FMs and newer versions

Customizing FMs is not easy

Data privacy and security

Getting FMs to execute tasks

Connecting to data sources

Difficult to manage infrastructure

NEW AWS AI Portfolio

INTERFACES & PROTOCOLS (MCP/A2A)

SECURITY AND POLICIES

APPLICATIONS AND AGENTS

- Kiro
- Amazon Quick Suite
- AWS Transform
- Amazon Connect
- AWS Marketplace

FRONTIER AGENTS

- Kiro autonomous agent
- AWS DevOps Agent
- AWS Security Agent

AI & AGENT DEVELOPMENT SOFTWARE & SERVICES

AGENT BUILDERS

Vertically Integrated
- Amazon Nova Act

Flexible/OSS
- Strands Agents

AMAZON BEDROCK

Models
- Amazon Nova
- 3P Models

Capabilities
- Optimization
- Guardrails
- Customization
- Knowledge Bases

AgentCore
- Runtime
- Gateway
- Memory
- 1P Tools
- Identity
- Observability
- Policy
- Evaluations

INFRASTRUCTURE

AMAZON SAGEMAKER

Custom Model Building
- Model building
- Model training
- Deployment
- MLOps
- Hyperpod

Data Foundation for AI
- Data Processing
- Governance
- Storage and Databases
- Vector Search

AI COMPUTE

- Trainium
- Inferentia
- GPUs

Amazon Bedrock

Open source

Protocols:
MCP, A2A

Frameworks:
CrewAI, LangGraph, Strands Agents

Amazon Bedrock

AgentCore

- Runtime
- Code Interpreter
- Gateway
- Browser
- Memory
- Policy
- Observability
- Evaluations
- Identity

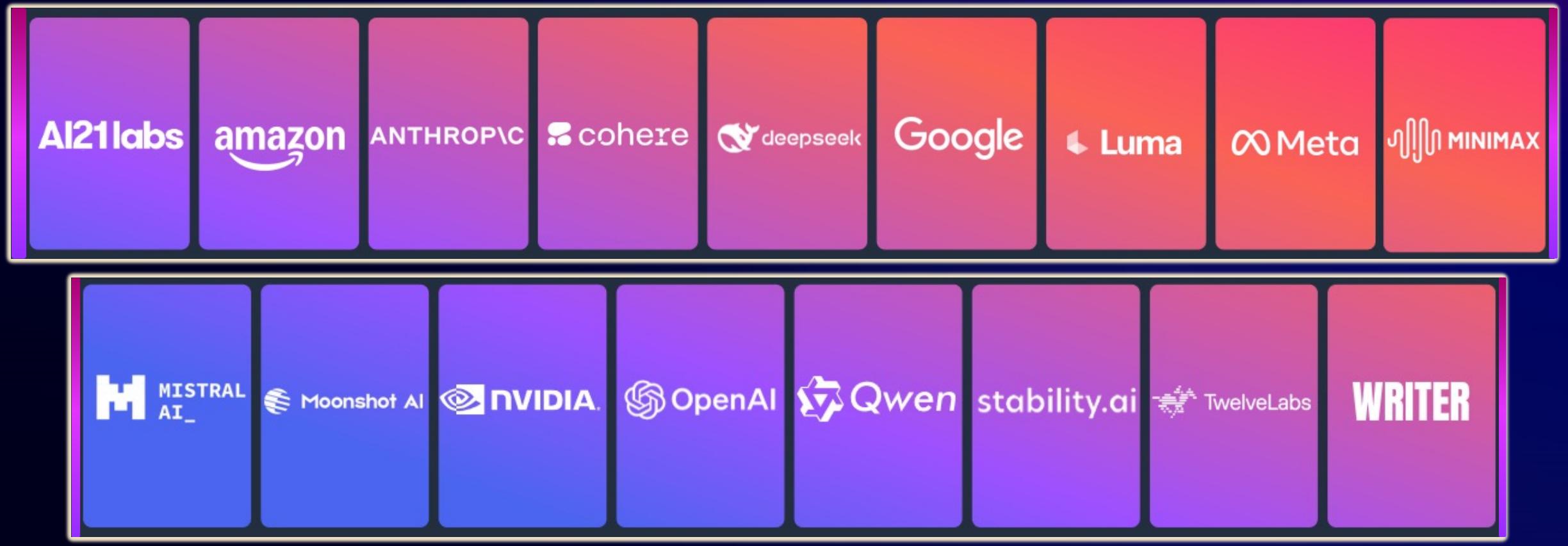
Model Capabilities

- Model access
- Cost and performance optimization
- Customization
- Guardrails

Knowledge Bases

Amazon Bedrock

Broad selection of fully managed models from leading AI companies



Amazon Bedrock Marketplace

100+ PUBLICLY AVAILABLE AND PROPRIETARY EMERGING, POPULAR, AND SPECIALIZED MODELS THROUGH A SERVERFUL OFFERING

IBM
NVIDIA
HUGGING FACE
MISTRAL AI
STABILITY AI
LG AI RESEARCH
SNOWFLAKE

UPSTAGE
PREFERRED NETWORKS
GRETEL
WIDN
EVOLUTIONARY SCALE
ARCEE AI
WRITER
CAMB.AI

DATABRICKS
STOCKMARK
KARAKURI
LIQUID
JOHN SNOW LABS
CYBERAGENT
NCSOFT

Prompt engineering

Prompt Engineering

- What is a **Prompt**?
  - Text input provided to an AI system to elicit a response
- What is **Prompt Engineering**?
  - Using techniques to craft prompts that steer FMs/LLMs towards desired responses
- Why is this important?
  - Enables control over models' behavior
  - Targets desired capabilities

Zero/Few-Shot Prompting

Zero Shot

Input

Tell me sentiment of this statement:
I loved the pizza at that Italian pizzeria

Output

The statement expresses a positive sentiment towards a pizza restaurant.

Few Shot

Input

Best Pakistani restaurant in Zurich: Positive
New York stinks, don't go there: Negative
The talk was on Generative AI: Neutral
This is a rip-off, store not recommended:

Output

Negative

Prompt Engineering - Methodology

1. Collect a set of test user queries to use as test cases
2. Write a prompt [template]
3. Evaluate against test cases
4. Refine prompt
5. Repeat steps 3 & 4 until results are satisfactory
6. Start using the prompt
7. Repeat steps 1 & 5 as requirements change

Prompt Engineering - Tips

Be clear and concise; use simple language

- Include context if needed
- Describe task, specify rules and directions about output formatting
- Provide diverse examples including edge cases (few-shot prompting)
- Instruct the model to say “I don’t know” to prevent hallucinations
- Follow the guidelines for each model (e.g. Reasoning models)
- Metaprompts – use an LLM to generate your prompt
- Leverage automatic prompt tuning/optimization frameworks (e.g. DSPy)

Prompt Engineering – Model adaptation

Different models expect prompt in different formats, depending on how they were trained or fine-tuned

- Model creators provide prompt guides for their model

- e.g. Amazon Nova:
  - Have clear sectional delimitation using markdown, XML, or other structure.
  - To define the name of the section, use ##Section Name##, then refer to that section in your prompt with ##Section Name##.
- e. g. Claude:
  - Any input and output elements should use XML tags, e.g. put context inside <context></context> tags
- e.g. Mistral
  - Add instructions like "You will only respond with a JSON object"
  - Use delimiters like ###, <<< >>>

Amazon Bedrock offers several features to balance cost, latency and accuracy

Amazon Bedrock support for prompt caching

Cache repetitive context in prompts across multiple API calls

Securely cache entire prompts

Enhance accuracy through longer, detailed prompts

Reduce costs by **up to 90**% and latency **by up to 85%** for supported models

Amazon Bedrock Intelligent Prompt Routing

Automatically route prompts to different foundation models to optimize response quality and lower costs

Provides a single endpoint to efficiently route prompts

Uses advanced prompt matching to meet cost and latency thresholds

Reduce costs by **up to 30%** without compromising on accuracy

Amazon Bedrock Model Distillation

Utilize smaller, more cost-effective models

Maximize distilled model performance with proprietary data synthesis

Distilled models are up to 500% faster and up to **75% less expensive** than original models, with **less than 2% accuracy loss** for use cases like RAG

Model Parameters

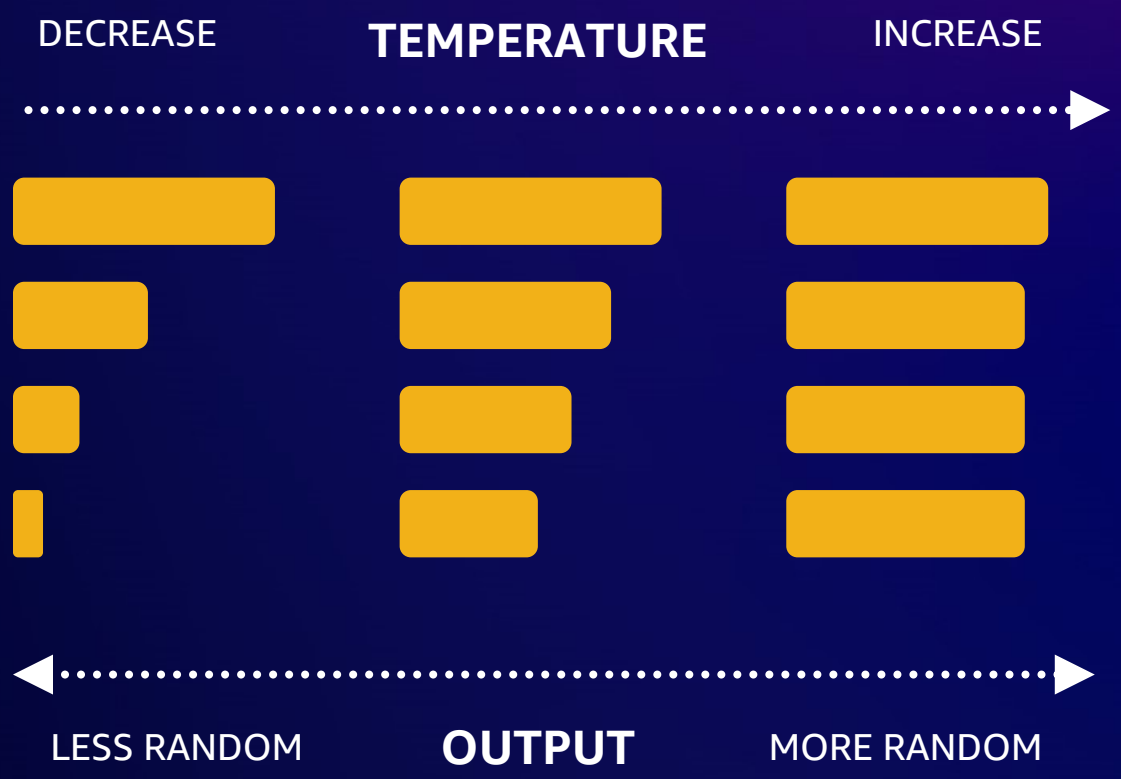
Model parameters - Variety

Temperature

Adjusts randomness of the output

Lower values make output more predictable

Higher values make output more creative



Model parameters - Variety

Temperature

Adjusts randomness of the output

Lower values make output more predictable

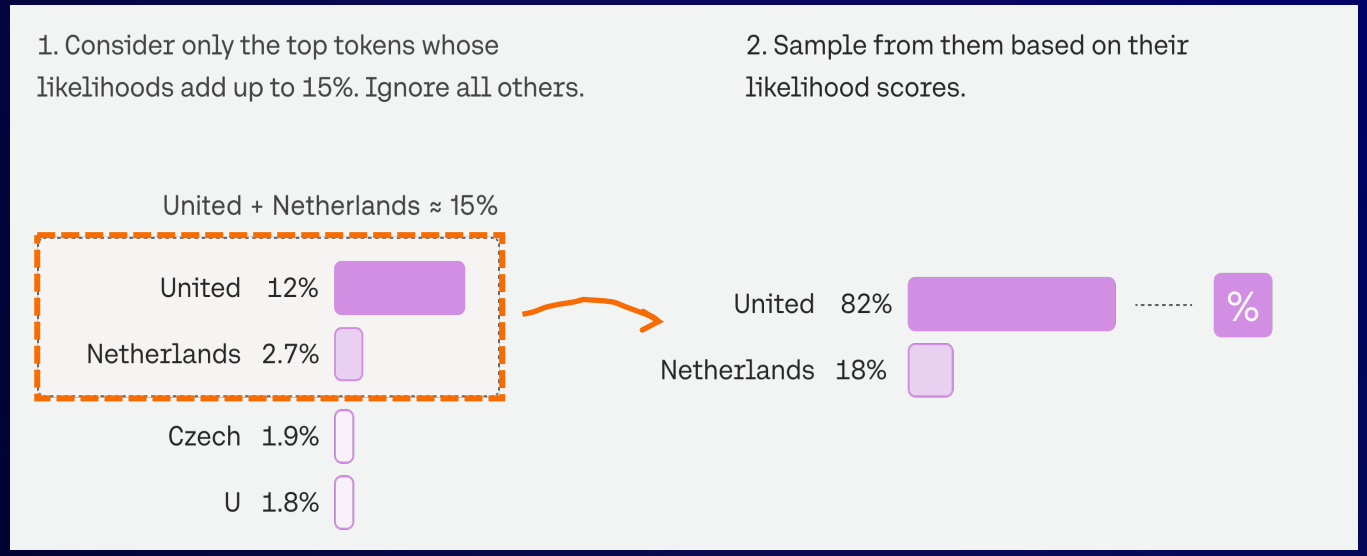
Higher values make output more creative

Top P/Top K

Only include the most likely tokens when choosing the next value

Limits the choices the model can pick from

Prevents choosing highly unlikely options



Model parameters - Length

Stop Words

Stop generation if one of the words is encountered.

Limit model output based on context

Allow model to generate different length of sequences based on input

Model parameters - Length

Stop Words

Stop generation if one of the words is encountered.

Limit model output based on context

Allow model to generate different length of sequences based on input

Max Token Count / Response Length

Stop generation once certain number of tokens is produced

Prevent high usage for unusual prompts

Limit cost of model errors

But wait, how can I use code?

THROUGH AMAZON BEDROCK 2 MAIN APIS

InvokeModel

Control model-specific parameters

Bedrock model identifier
e.g anthropic.claude-sonnet-4-20250514-v1:0

Inference configuration
temperature, top_p , max_tokens etc.

Messages (actual prompt)
The prompt to sent to the model

```python
client = boto3.client("bedrock-runtime", region_name="us-east-1")
client.invoke_model(
    modelId=model_id,
    body=json.dumps({
        "anthropic_version": "bedrock-2023-05-31",
        "max_tokens": 512,
        "temperature": 0.5,
        "messages": [{
            "role": "user",
            "content": [{"type": "text", "text": prompt}],
        }],
    }),
)
```

The *body* argument structure will change based on the model used

But wait, how can I use code?

THROUGH AMAZON BEDROCK 2 MAIN APIS

Converse

Unified Interface Across Models

Bedrock model identifier
e.g anthropic.claude-sonnet-4-20250514-v1:0

Inference configuration
temperature, top_p , max_tokens etc.

Messages (actual prompt)
The prompt to sent to the model

```python
client = boto3.client("bedrock-runtime", region_name="us-east-1")
client.converse(
    modelId=model_id,
    inferenceConfig={"maxTokens": 512, "temperature": 0.5},
    messages=[{
        "role": "user",
        "content": [{"text": prompt}],
    }],
)
```

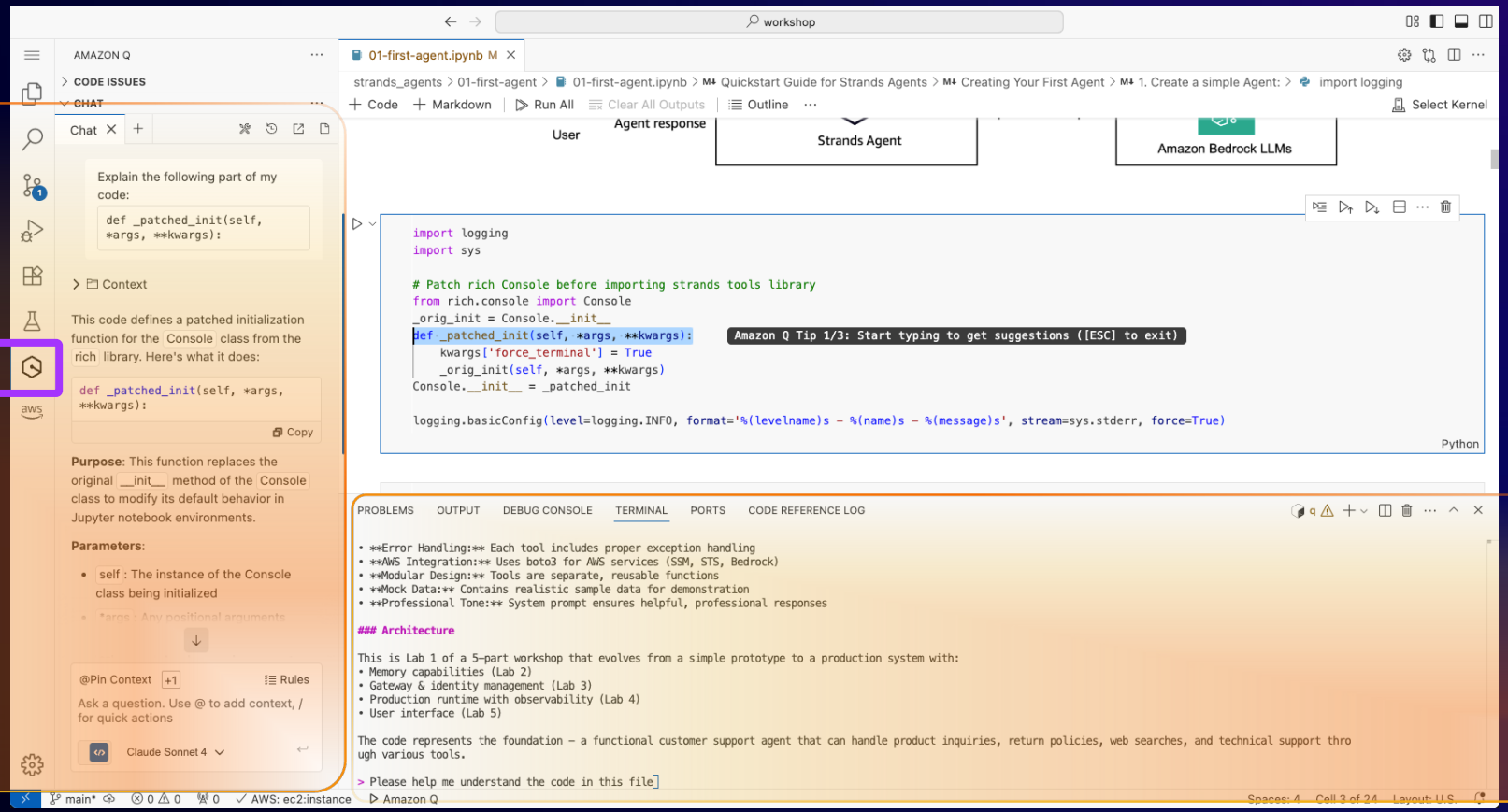
Recommended API
provides a consistent interface across all Amazon Bedrock models

Workshop time!

Ask us,...
...or ask Amazon Q & Kiro

Amazon Q Developer Extension

Click the Q extension icon to begin

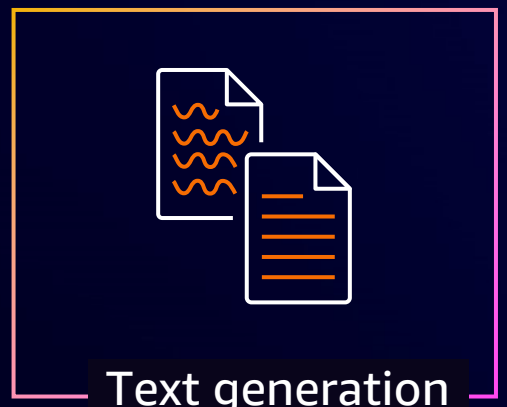


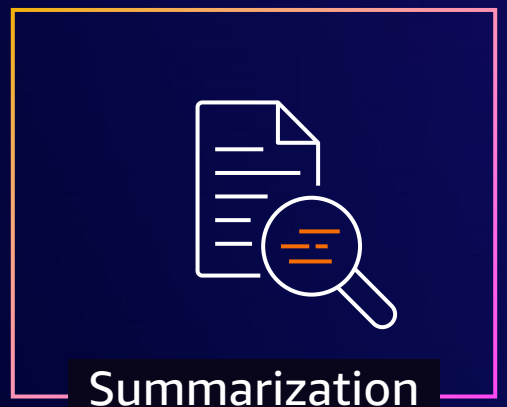
Kiro CLI

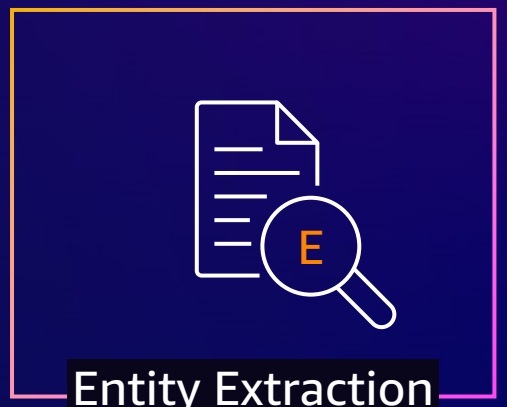
Type “kiro-cli” to begin

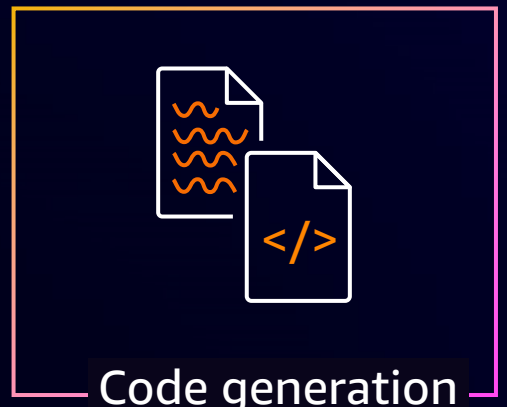
We’ve installed the Amazon Q Extension and Kiro CLI for you.
Make use of it during the workshop!

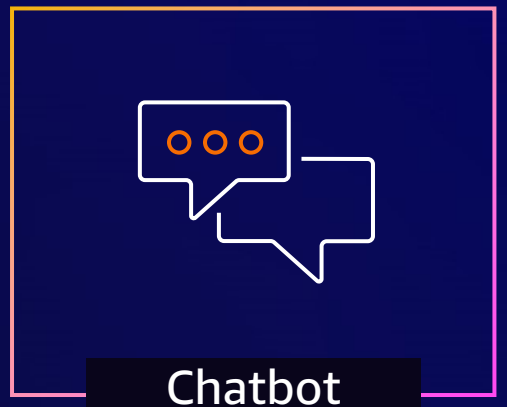
Let’s get hands on - Architecture patterns











Multimodality with Amazon Nova

Amazon Nova Foundation Models

State-of-the-art foundation models that deliver frontier intelligence and industry-leading price performance

UNDERSTANDING MODELS

Amazon Nova **Micro**

Amazon Nova **Lite**

Amazon Nova **Pro**

Amazon Nova **Premier**

CREATIVE CONTENT GENERATION MODELS

Amazon Nova **Canvas**

Amazon Nova **Reel**

SPEECH-TO-SPEECH MODEL

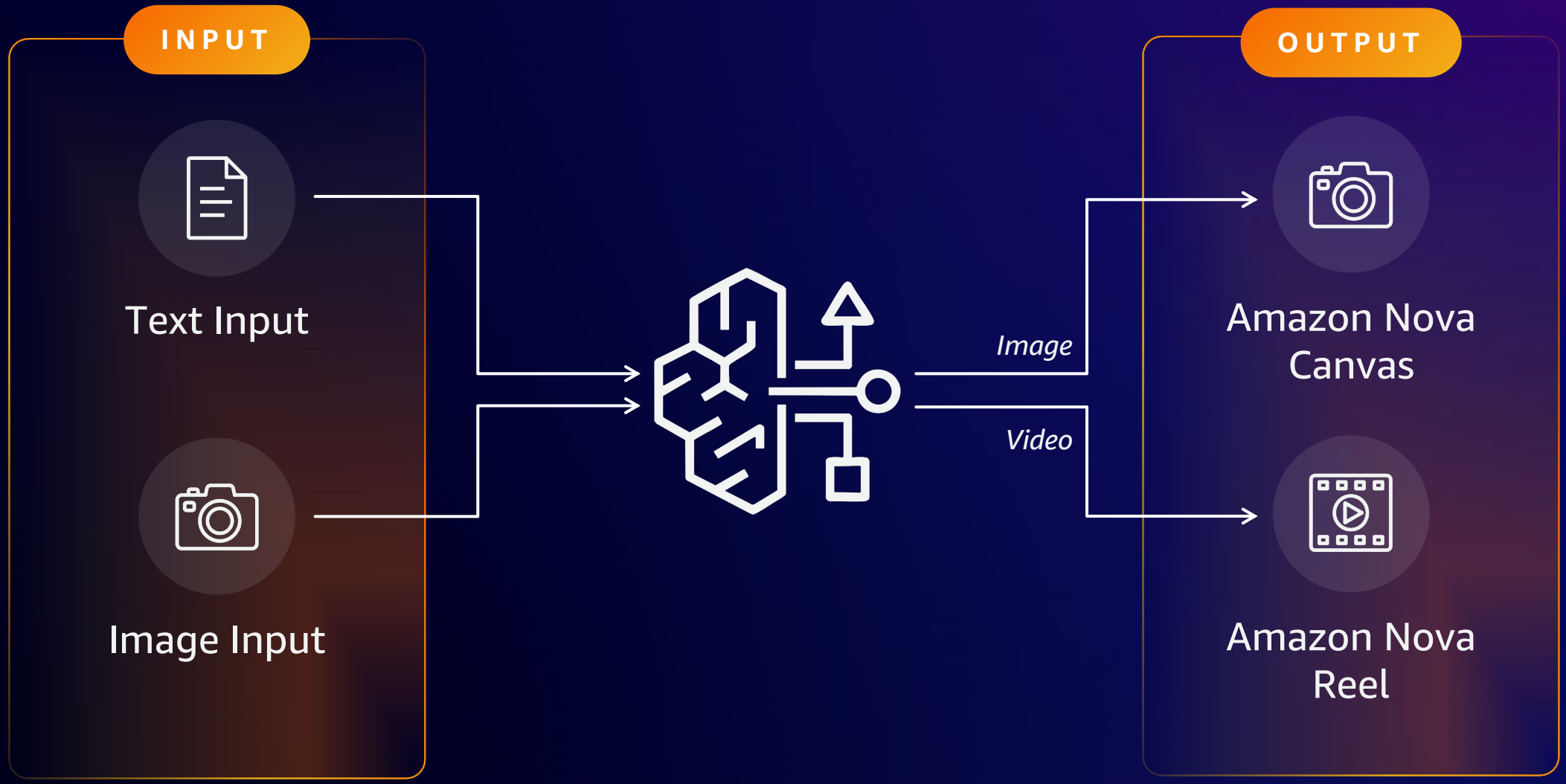
Amazon Nova **Sonic**

aws

Amazon Nova Understanding Models

	Amazon Nova Micro	Amazon Nova Lite	Amazon Nova Pro	Amazon Nova Premier
Availability	GA	GA	GA	GA
Context window	128K	300K	300K (5M coming soon)	1M
Languages	200+ languages	200+ languages	200+ languages	200 + languages
Modalities supported	Text input; text output	Text, image, video input; text output	Text, image, video input; text output	Text, image, video input; text output
Fine-tuning	Yes	Yes	Yes	Yes

Amazon Nova Creative-Content Models



NEW

Amazon Nova 2 Models

Nova 2 foundation models

GENERALLY AVAILABLE

Amazon
Nova 2 Lite

Fast, cost-effective reasoning model for everyday workloads

PREVIEW

Amazon
Nova 2 Pro

Our most intelligent reasoning model for highly complex tasks

PREVIEW

Amazon
Nova 2 Omni

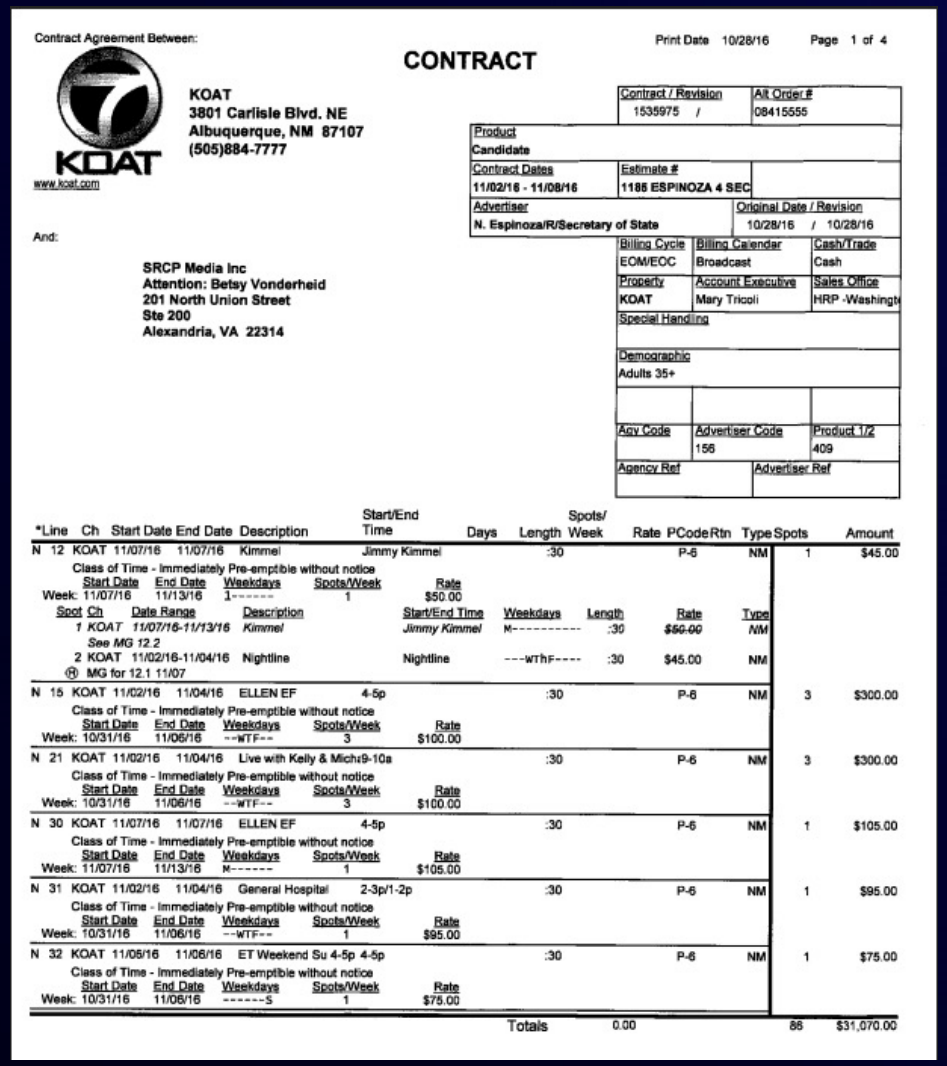
Unified model for multimodal reasoning and image generation

GENERALLY AVAILABLE

Amazon
Nova 2 Sonic

Speech-to-speech foundation model for real-time, human-like conversational AI

Nova 2 Omni extracts key information from documents



Contract Agreement Between:

KOAT
3801 Carlisle Blvd. NE
Albuquerque, NM 87107
(505)884-7777
www.koat.com

And:

SRCP Media Inc
Attention: Betsy Vonderheid
201 North Union Street
Ste 200
Alexandria, VA 22314

CONTRACT

Print Date 10/28/16 Page 1 of 4

Contract / Revision	Alt Order #
1535975 /	08415555

Product: Candidate

Contract Dates	Estimate #
11/02/16 - 11/08/16	1185 ESPINOZA 4 SEC

Advertiser	Original Date / Revision
N. Espinoza/R/Secretary of State	10/28/16 / 10/28/16

Billing Cycle	Billing Calendar	Cash/Trade
EOM/EOC	Broadcast	Cash

Property	Account Executive	Sales Office
KOAT	Mary Tricoli	HRP -Washingt

Special Handling

Demographic: Adults 35+

Agy Code	Advertiser Code	Product 1/2
	156	409

Agency Ref	Advertiser Ref

*Line	Ch	Start Date	End Date	Description	Start/End Time	Days	Length	Spots/ Week	Rate	PCode	Rtn	Type	Spots	Amount
N 12	KOAT	11/07/16	11/07/16	Kimmel	Jimmy Kimmel		:30			P-6		NM	1	$45.00
Class of Time - Immediately Pre-emptible without notice														
Week:	Start Date 11/07/16	End Date 11/13/16	Weekdays 1------	Spots/Week 1	Rate $50.00									
Spot	Ch	Date Range	Description	Start/End Time	Weekdays	Length	Rate	Type						
1	KOAT	11/07/16-11/13/16	Kimmel	Jimmy Kimmel	M----------	:30	$50.00	NM						
See MG 12.2														
2	KOAT	11/02/16-11/04/16	Nightline	Nightline	---WThF----	:30	$45.00	NM						
Ⓗ MG for 12.1 11/07														
N 15	KOAT	11/02/16	11/04/16	ELLEN EF	4-5p		:30			P-6		NM	3	$300.00
Class of Time - Immediately Pre-emptible without notice														
Week:	Start Date 10/31/16	End Date 11/06/16	Weekdays --WTF--	Spots/Week 3	Rate $100.00									
N 21	KOAT	11/02/16	11/04/16	Live with Kelly & Micha	9-10a		:30			P-6		NM	3	$300.00
Class of Time - Immediately Pre-emptible without notice														
Week:	Start Date 10/31/16	End Date 11/06/16	Weekdays --WTF--	Spots/Week 3	Rate $100.00									
N 30	KOAT	11/07/16	11/07/16	ELLEN EF	4-5p		:30			P-6		NM	1	$105.00
Class of Time - Immediately Pre-emptible without notice														
Week:	Start Date 11/07/16	End Date 11/13/16	Weekdays M------	Spots/Week 1	Rate $105.00									
N 31	KOAT	11/02/16	11/04/16	General Hospital	2-3p/1-2p		:30			P-6		NM	1	$95.00
Class of Time - Immediately Pre-emptible without notice														
Week:	Start Date 10/31/16	End Date 11/06/16	Weekdays --WTF--	Spots/Week 1	Rate $95.00									
N 32	KOAT	11/05/16	11/06/16	ET Weekend Su 4-5p	4-5p		:30			P-6		NM	1	$75.00
Class of Time - Immediately Pre-emptible without notice														
Week:	Start Date 10/31/16	End Date 11/06/16	Weekdays ------S	Spots/Week 1	Rate $75.00									
					Totals		0.00						86	$31,070.00

Between KOAT & SRCP Media Inc
Date: 10/28/16

Spot Schedule

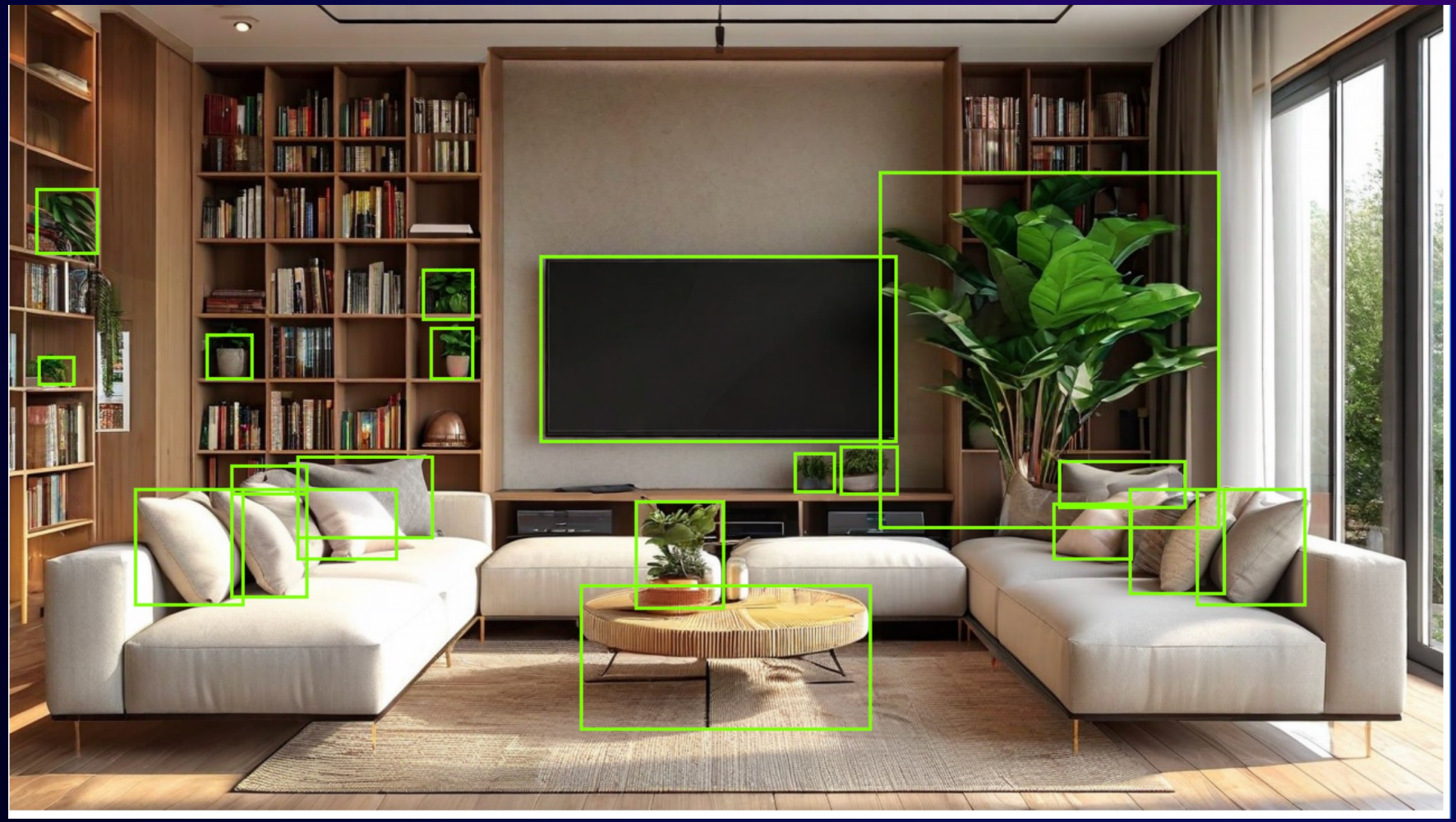
Line	Program	Date Range	Time	Length	Rate	Spots
12	Jimmy Kimmel	11/07/16	--	:30	$50.00	1
15	ELLEN	11/02-04/16	4-5p	:30	$100.00	3
21	Live with Kelly	11/02-04/16	9-10a	:30	$100.00	3
30	ELLEN	11/07/16	4-5p	:30	$105.00	1
31	General Hospital	11/02-04/16	2-3p	:30	$95.00	1
32	ET Weekend	11/06/16	4-5p	:30	$75.00	1

Financial Terms

- Gross: $31,070.00
- Agency Comm: ($4,660.50)
- Net: $26,409.50
- Tax (7.313%): $1,931.33
- **Total: $28,340.83**

And understands what objects are in your images

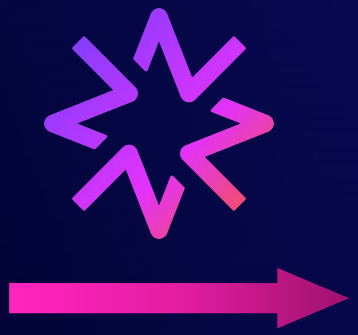
Prompt:
Detect plants, cushions, tables and the TV



Nova 2 Omni can also generate hyper-realistic imagery

Prompt:
Solo female traveler with a backpack, standing on mountain summit with arms raised in triumph

And edit your existing images



Prompt: Change the snowy environment surrounding the house to a beach setting with sand and palm trees.

Nova 2 Lite A fast cost-effective reasoning model	**Input:** Text, image, video **Output:** Text	**Use cases:** Everyday AI tasks such as chatbots, code generation	**Context window:** 1M	**Developer controls for thinking effort:** Yes
Nova 2 Pro (preview) Our most intelligent reasoning model	**Input:** Text, image, video, speech **Output:** Text	**Use cases:** Complex AI tasks such as agentic coding	**Context window:** 1M	**Developer controls for thinking effort:** Yes
Nova 2 Omni (preview) All-in-one model for multimodal reasoning and image generation	**Input:** Text, image, video, speech **Output:** Text, image	**Use cases:** Image generation, multimodal understanding	**Context window:** 1M	**Developer controls for thinking effort:** Yes
Nova 2 Sonic A speech-to-speech model for conversational AI	**Input:** Text, speech **Output:** Text, speech	**Use cases:** Customer support, voice-enabled personal assistant	**Context window:** 1M	
Nova Multimodal Embeddings A state-of-the-art multimodal embedding model	**Input:** Text, image, video, audio **Output:** Embedding	**Use cases:** Semantic search, agentic RAG	**Context window:** 8K	

GENERALLY AVAILABLE

Amazon Nova Forge

Build your own frontier models using Amazon Nova

1. Access checkpoints across all phases of model development, and leverage new Nova models before they are widely available
2. Blend your proprietary data with Amazon Nova-curated training data
3. Perform reinforcement learning with reward functions in your environment
4. Use push-button recipes that are optimized to build with Nova through visual workflows or a command line interface
5. Use the built-in responsible AI toolkit to implement custom safety guardrails

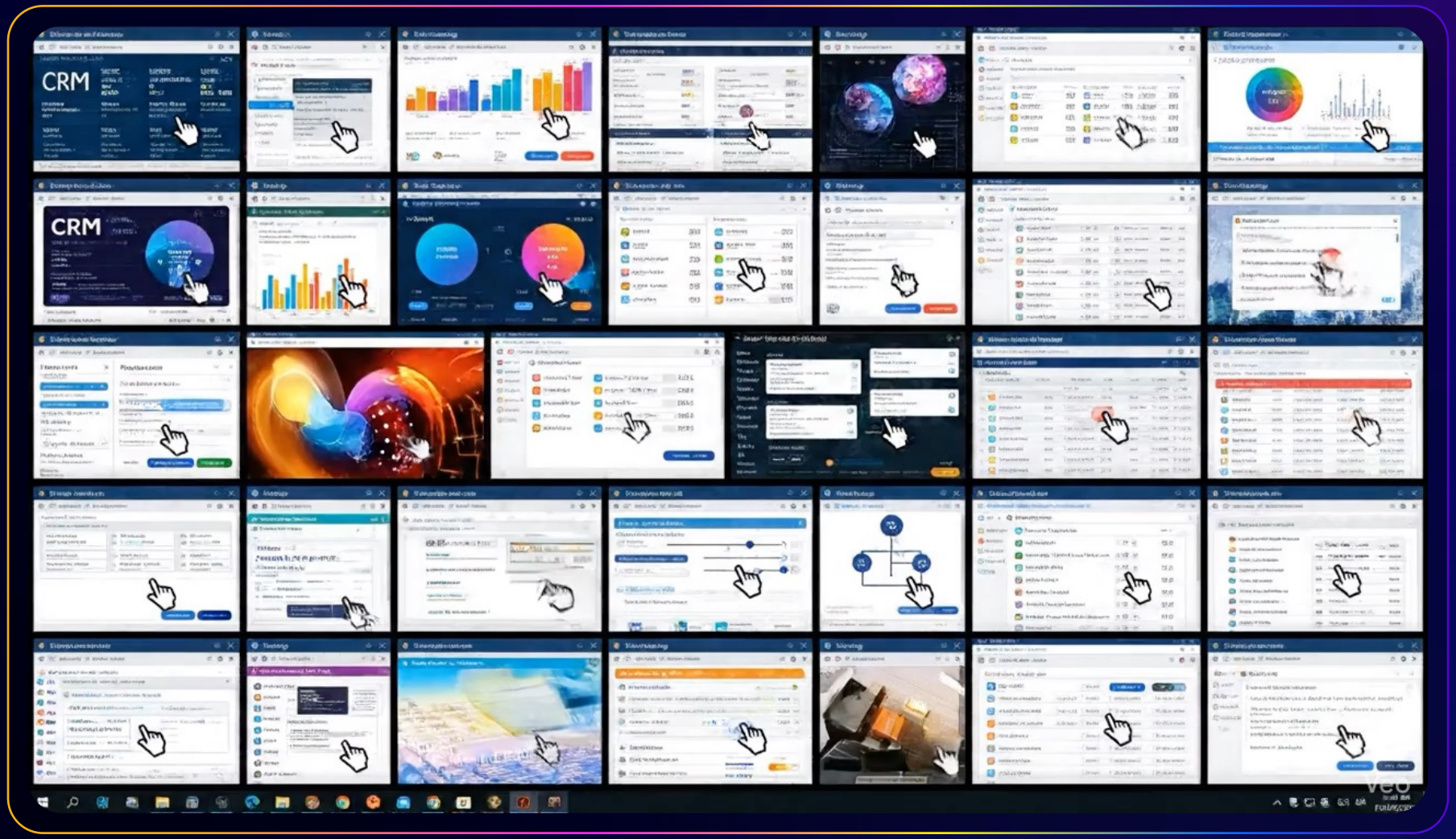
GENERALLY AVAILABLE

Amazon Nova Act

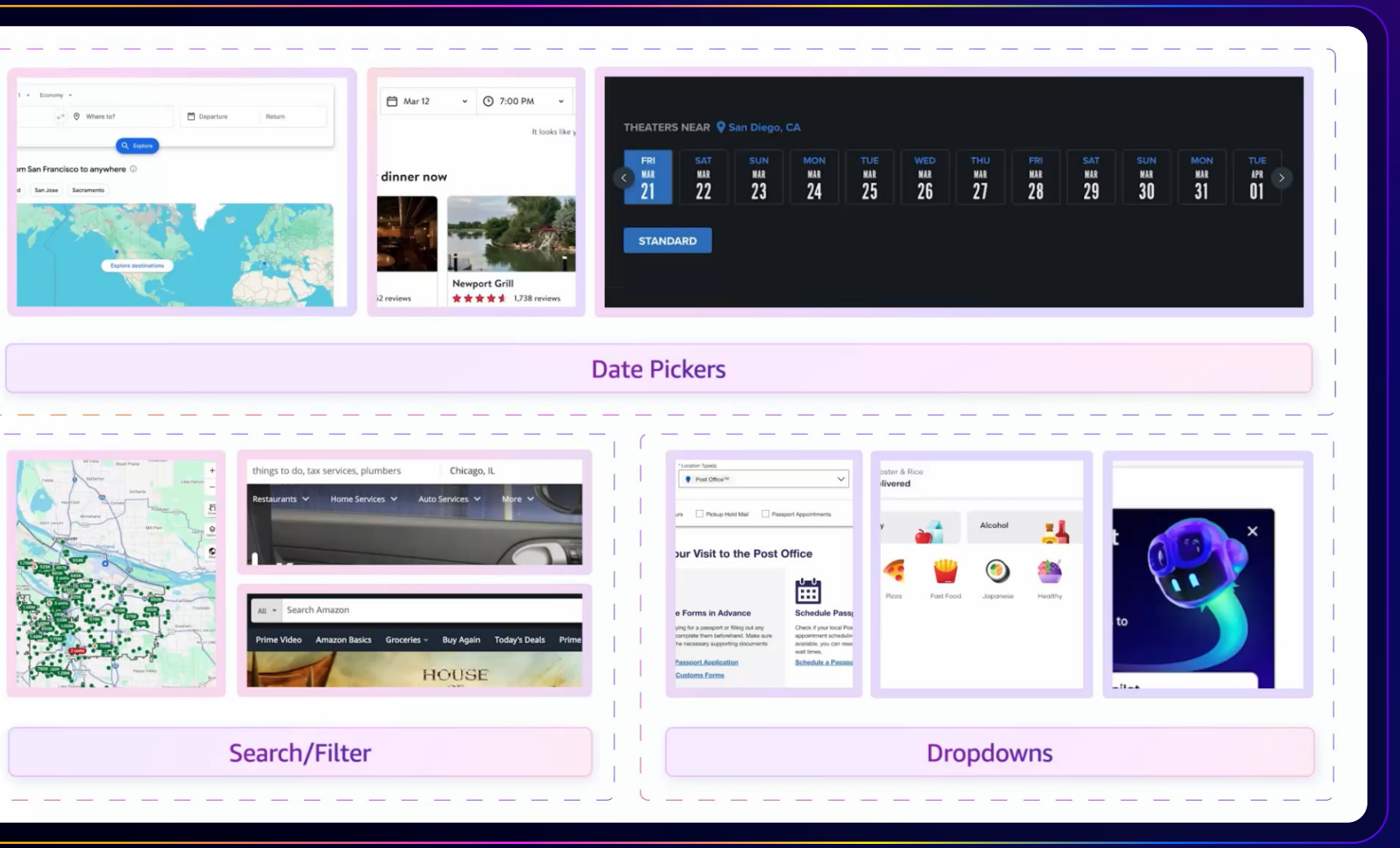
Our AWS service for building, deploying and running reliable AI agents

- Browser-user model
- New AWS Service & Console
- Nova Act Playground
- SDK
- IDE Extension
- CLI to deploy to AWS
- Human-in-the-loop
- Tool Use

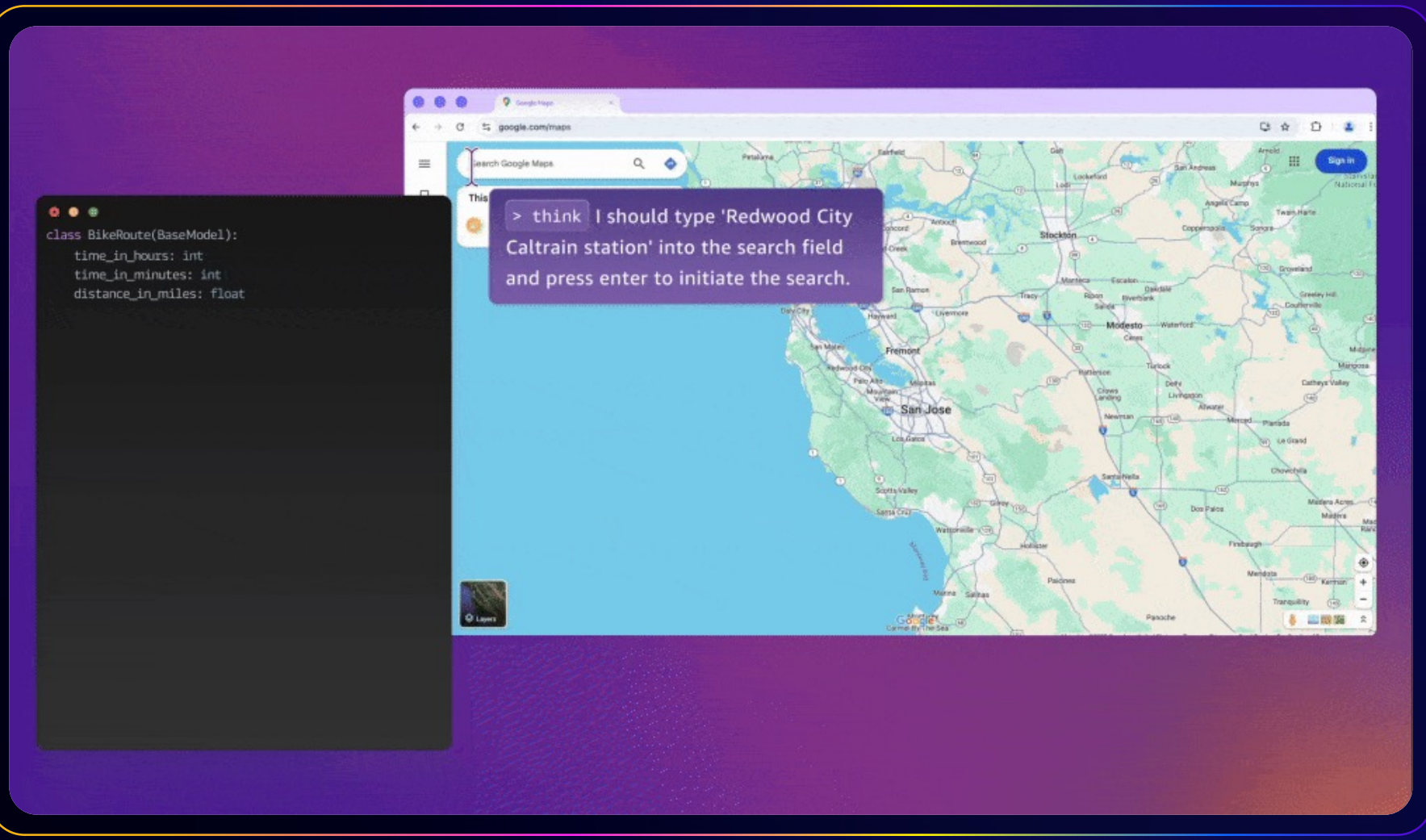
Trained using real web interactions



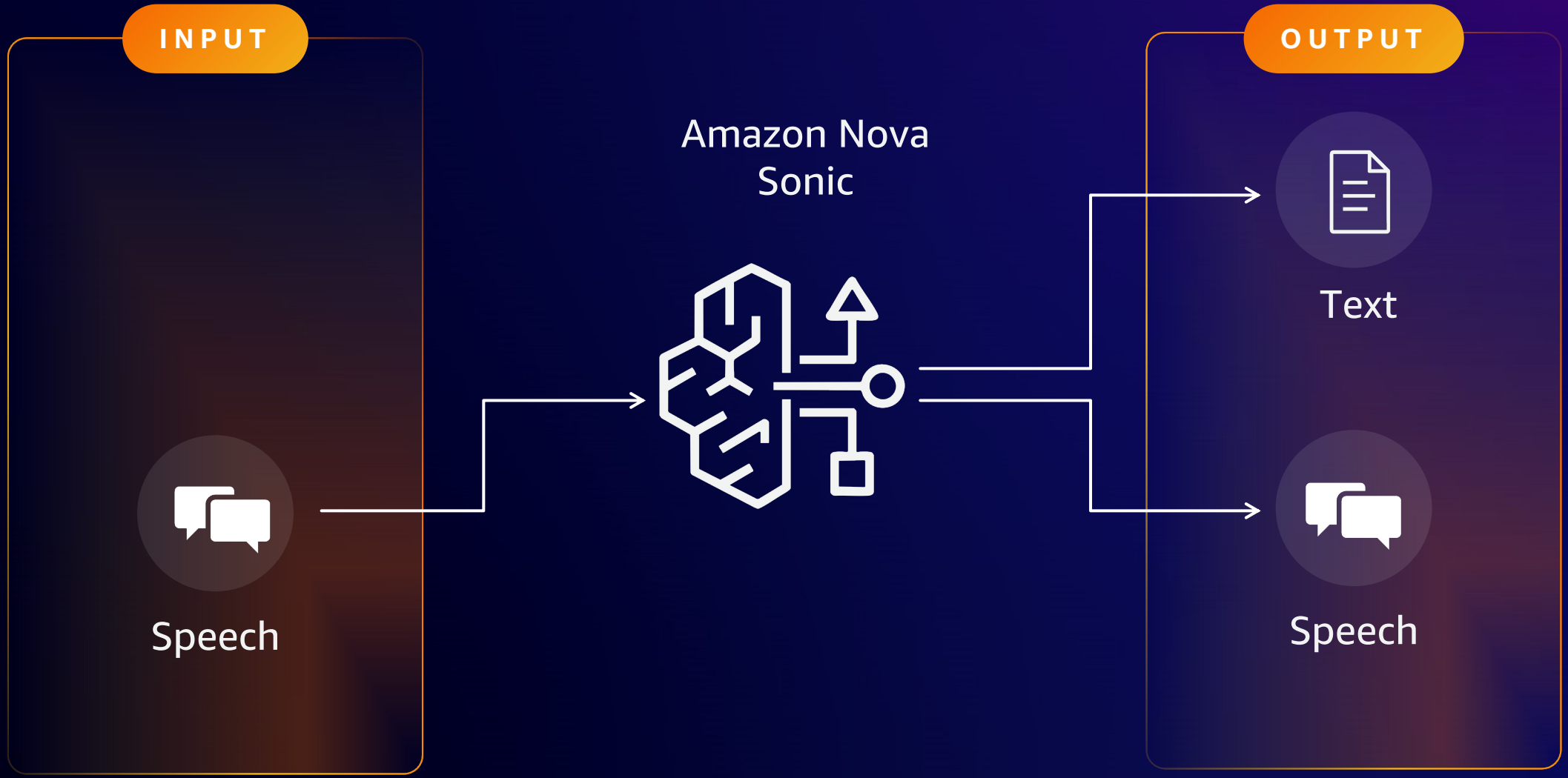
Act can interact with key browser elements



Mimicking real user actions

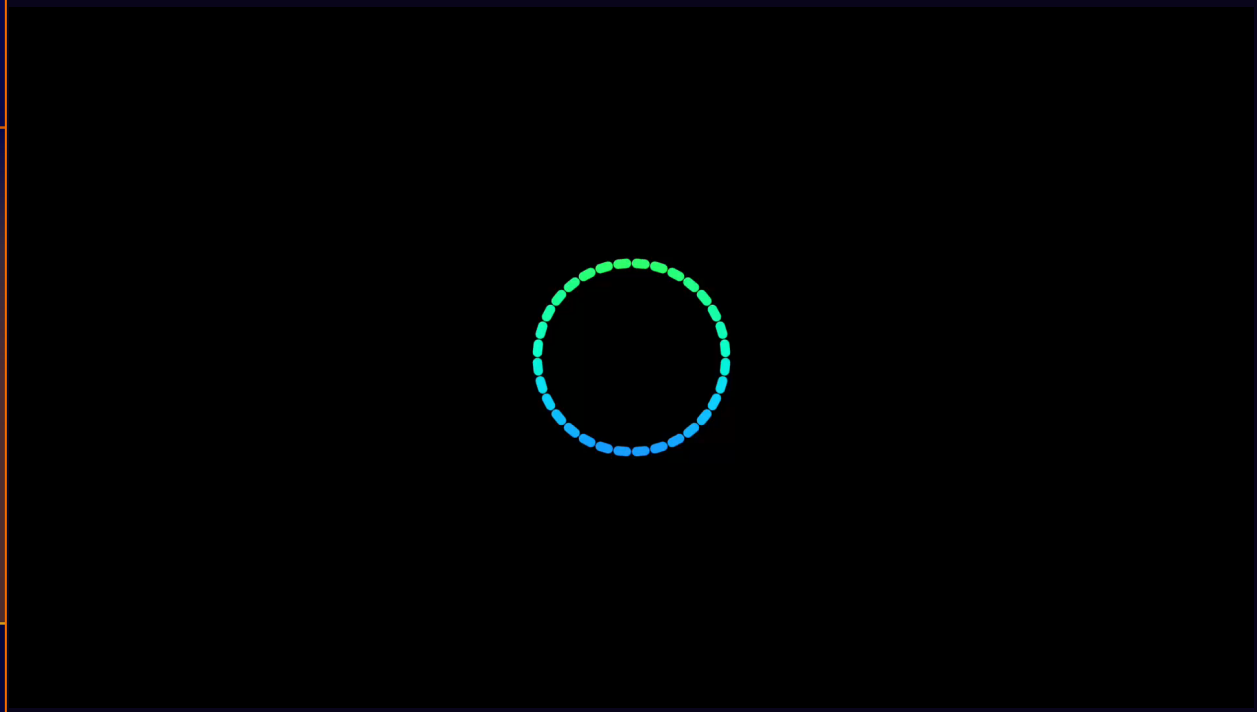


Amazon Nova Speech-to-Speech



Amazon Nova Sonic In Action

Customer service call automation



Recapping Amazon Nova

Comprehensive suite of models and services with industry leading price performance

AGENT SERVICES

Amazon
Nova Act

CUSTOMIZATION

Amazon
Nova Forge

SECOND GENERATION MODELS

Amazon
Nova 2 Lite

Amazon
Nova 2 Sonic

PREVIEW MODELS

Amazon
Nova 2 Pro

Amazon
Nova 2 Omni

GENAI BOOTCAMP

Notebook Exercises

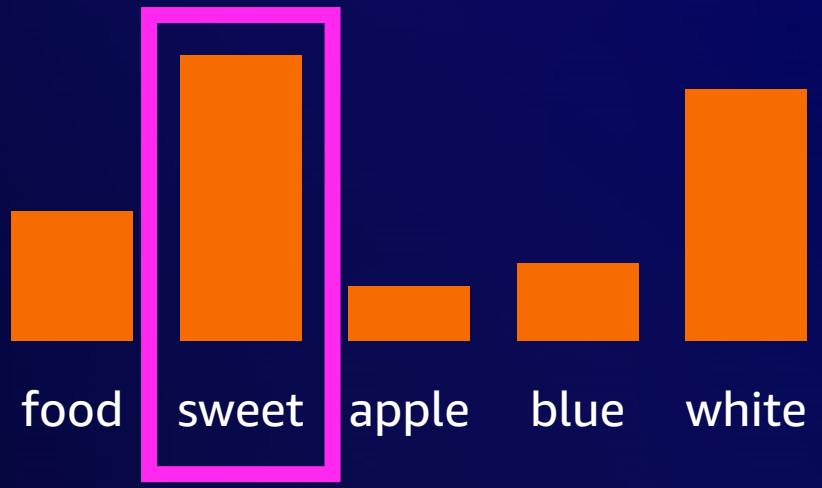
Retrieval Augmented Generation (RAG)

Large Language Models Recap

Roses are red, violets are blue, sugar is _____

Large Language Models Recap

Roses are red, violets are blue, sugar is sweet



Private Data

What is AnyCompany's vacation policy?

How do we "teach" the model about AnyCompany?

Use Cases

Improved content quality

E.g., helps in reducing hallucinations and connecting with recent knowledge including enterprise data

Contextual chatbots and question answering

E.g., enhance chatbot capabilities by integrating with real-time data

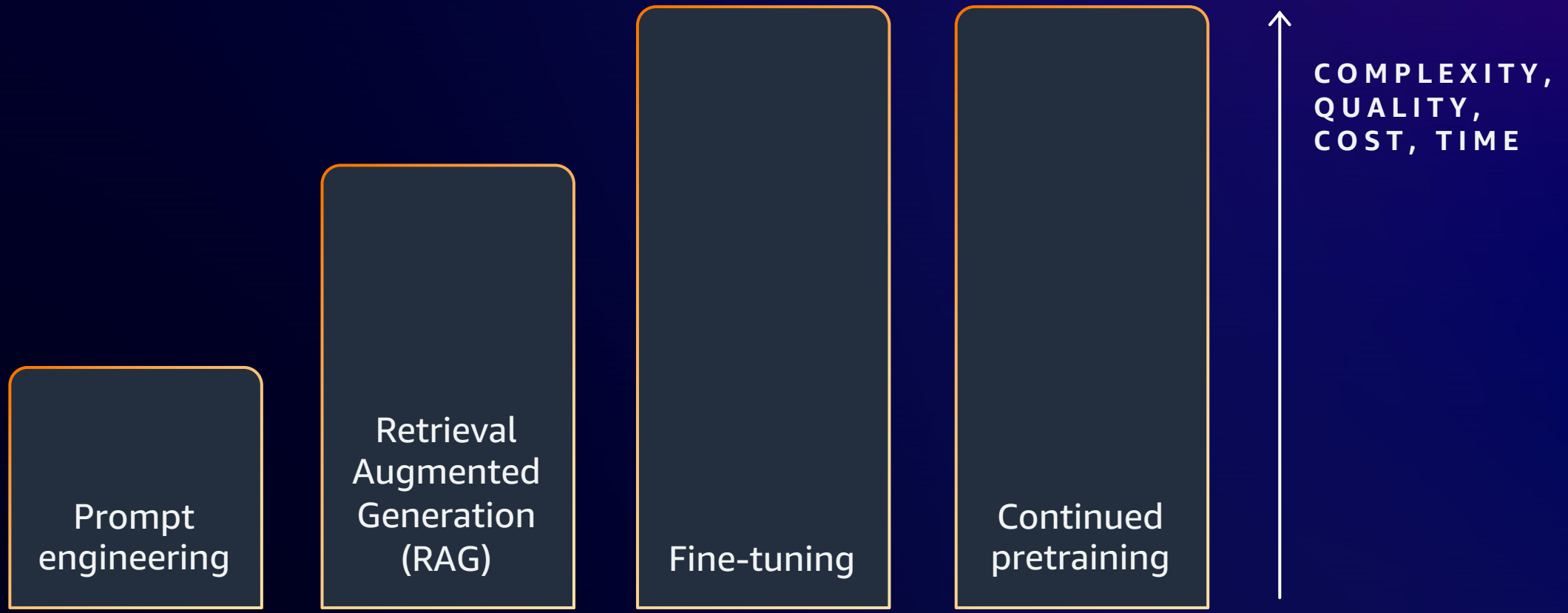
Personalized search

E.g., searching based on user previous search history and persona

Real-time data summarization

E.g., retrieving and summarizing transactional data from databases, or API calls

Customizing LLMs



File Attachment

Based on the following file, what is the vacation policy?

File Attachment

Based on the following file, what is the vacation policy?

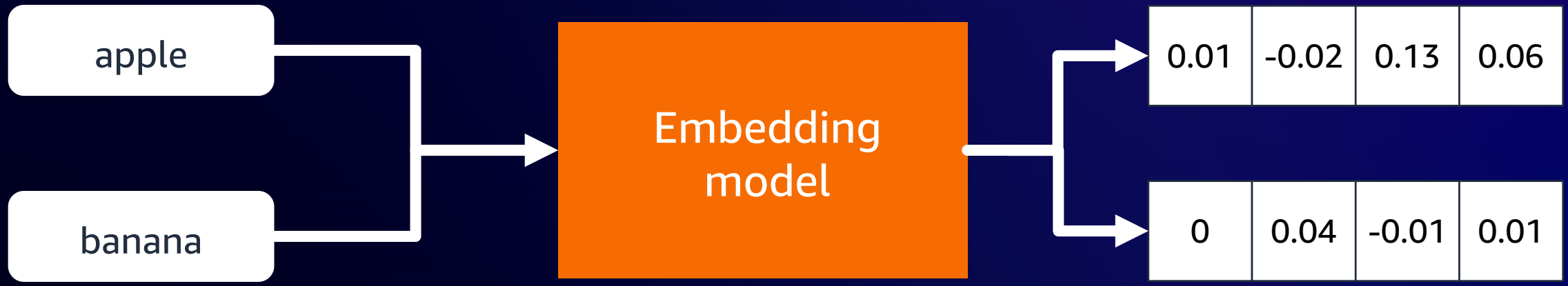
<file_content>
Employees are entitled to 10 days (80 hours) of paid vacation annually after completing 6 months of employment. Vacation time must be requested at least 2 weeks in advance and approved by the immediate supervisor.
</file_content>

Based on our company policy, you get 10 paid vacation days (or 80 hours) each year once you've been with us for 6 months. Just make sure to submit your vacation requests at least 2 weeks ahead of time so your supervisor can approve them. Let me know if you have any other questions!

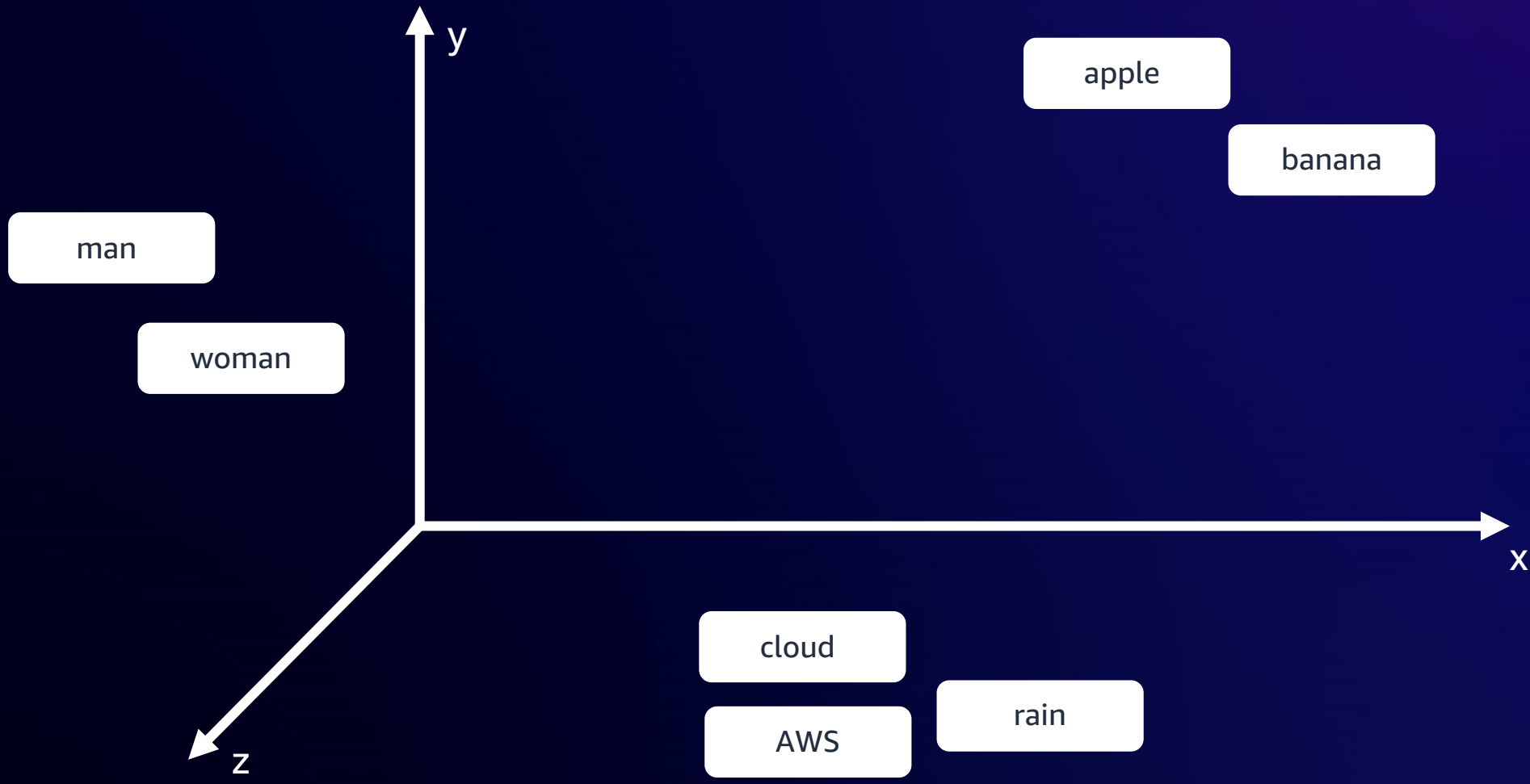
We know how to use private data!

But how do we find the right file?

Embeddings



Embeddings



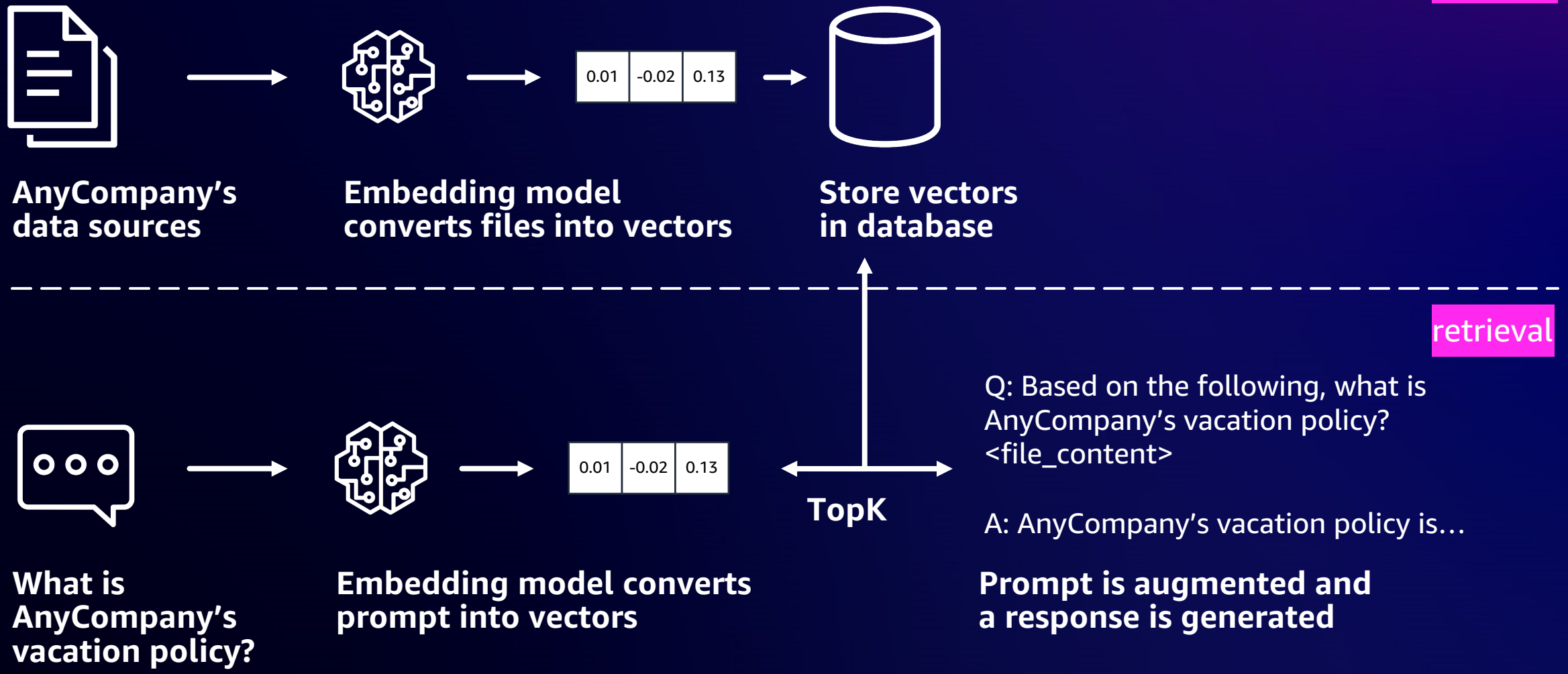
Vector Databases

“Retrieve the top 2 closest vectors to [0.04, -0.02, 0.1, 0.3]”

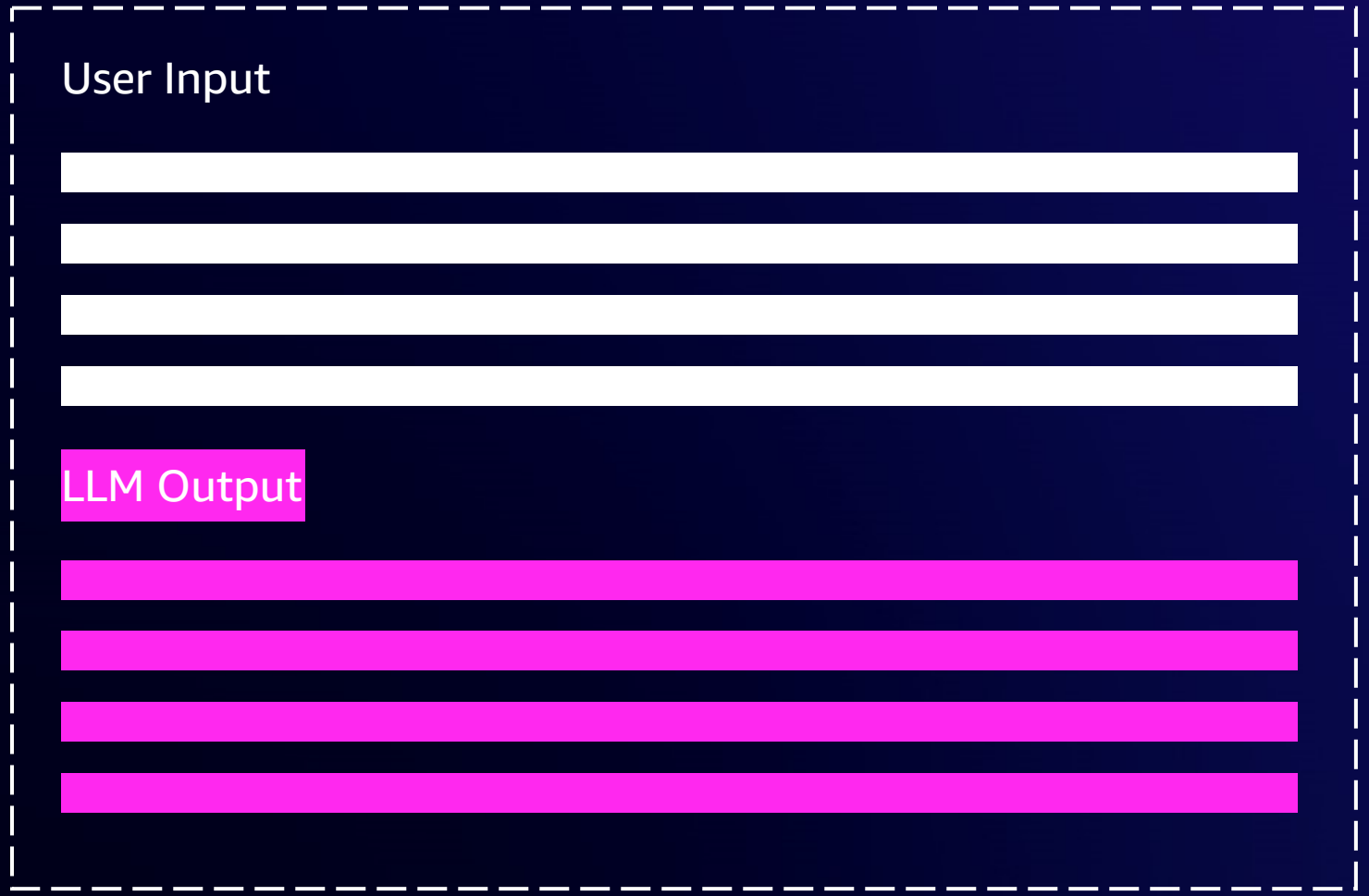
0.01	-0.02	0.13	0.06
-0.11	-0.02	0.49	0

0.01	-0.02	0.13	0.06	apple
-0.11	-0.02	0.49	0	banana
0	0.13	-0.4	0.01	AWS

Putting It All Together



Context Window



Max tokens LLM can handle at once (input + output)

So what about big files?

Chunking

SECTION 3: COMPENSATION AND SALARY STRUCTURE

3(a) Base Salary Qualification Requirements

i. Veterinarians holding a Doctor of Veterinary Medicine (DVM) degree from an AVMA-accredited institution shall receive a minimum base salary of $85,000 per annum.

ii. Veterinarians holding additional board certifications shall receive a premium of $15,000 per certification above the base salary.

iii. Non-licensed veterinary professionals or those with pending licensure shall receive 70% of the base salary until full licensure is obtained.

3(b) Experience-Based Compensation

i. For each year of post-DVM clinical experience, employees shall receive an annual increment of $2,500, capped at 15 years.

ii. Specialized experience in emergency medicine, surgery, or exotic animal care shall warrant an additional $7,500 per annum.

iii. Service as a lead veterinarian or department head shall result in a 12% increase to the adjusted base salary.

3(c) Performance and Additional Compensation

i. Annual performance reviews resulting in "Exceeds Expectations" ratings shall merit a 5% bonus of the total annual salary.

ii. On-call duties shall be compensated at a rate of $200 per weekday night and $300 per weekend day/night.

iii. Continuing education completion beyond the minimum required hours shall be rewarded at $500 per 10 additional hours, with an annual cap of $2,500.

Chunking

SECTION 3: COMPENSATION AND SALARY STRUCTURE

3(a) Base Salary Qualification Requirements

i. Veterinarians holding a Doctor of Veterinary Medicine (DVM) degree from an AVMA-accredited institution shall receive a minimum base salary of $85,000 per annum.

ii. Veterinarians holding additional board certifications shall receive a premium of $15,000 per certification above the base salary.

iii. Non-licensed veterinary professionals or those with pending licensure shall receive 70% of the base salary until full licensure is obtained.

3(b) Experience-Based Compensation

i. For each year of post-DVM clinical experience, employees shall receive an annual increment of $2,500, capped at 15 years.

ii. Specialized experience in emergency medicine, surgery, or exotic animal care shall warrant an additional $7,500 per annum.

iii. Service as a lead veterinarian or department head shall result in a 12% increase to the adjusted base salary.

3(c) Performance and Additional Compensation

i. Annual performance reviews resulting in "Exceeds Expectations" ratings shall merit a 5% bonus of the total annual salary.

ii. On-call duties shall be compensated at a rate of $200 per weekday night and $300 per weekend day/night.

iii. Continuing education completion beyond the minimum required hours shall be rewarded at $500 per 10 additional hours, with an annual cap of $2,500.

Chunking

SECTION 3: COMPENSATION AND SALARY STRUCTURE

3(a) Base Salary Qualification Requirements

i. Veterinarians holding a Doctor of Veterinary Medicine (DVM) degree from an AVMA-accredited institution shall receive a minimum base salary of $85,000 per annum.

ii. Veterinarians holding additional board certifications shall receive a premium of $15,000 per certification above the base salary.

iii. Non-licensed veterinary professionals or those with pending licensure shall receive 70% of the base salary until full licensure is obtained.

3(b) Experience-Based Compensation

i. For each year of post-DVM clinical experience, employees shall receive an annual increment of $2,500, capped at 15 years.

ii. Specialized experience in emergency medicine, surgery, or exotic animal care shall warrant an additional $7,500 per annum.

iii. Service as a lead veterinarian or department head shall result in a 12% increase to the adjusted base salary.

3(c) Performance and Additional Compensation

i. Annual performance reviews resulting in "Exceeds Expectations" ratings shall merit a 5% bonus of the total annual salary.

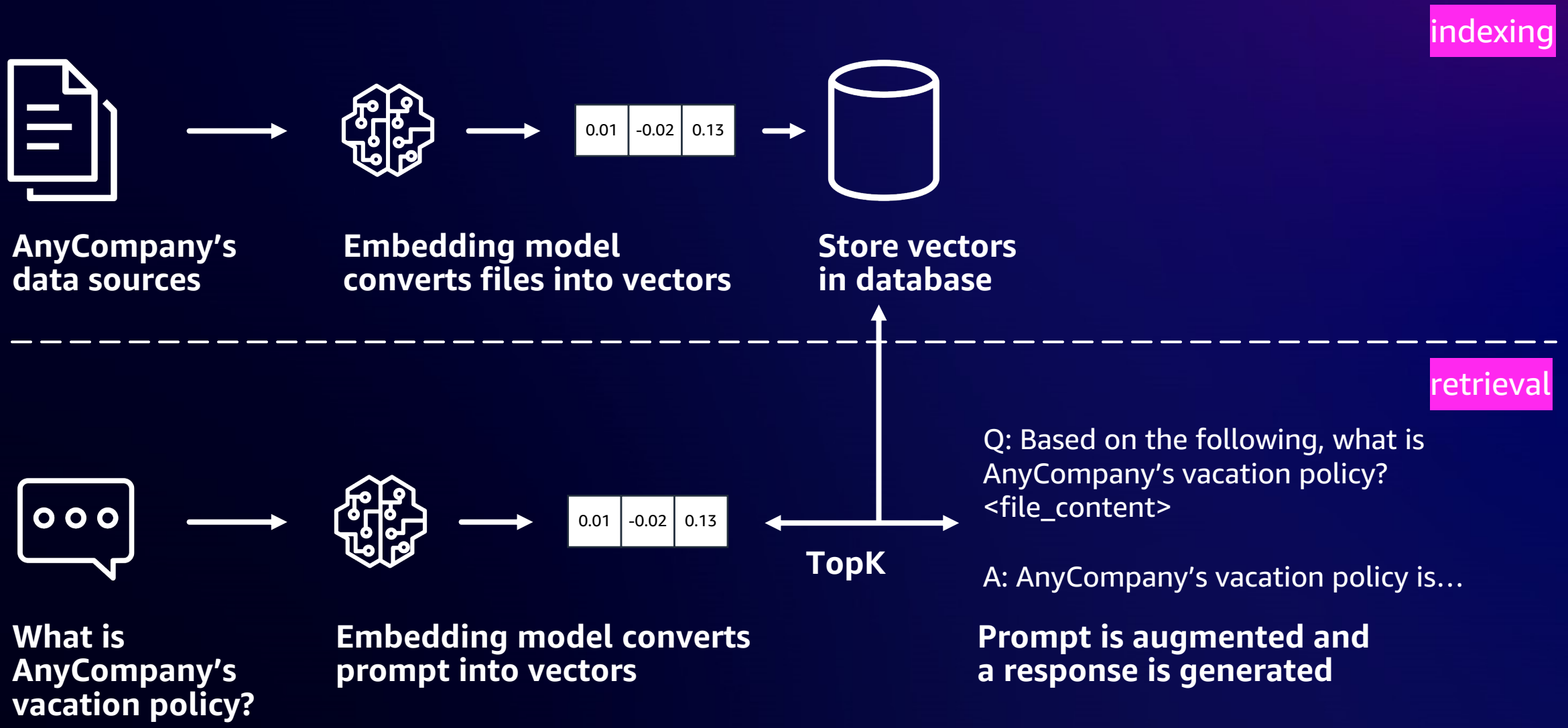
ii. On-call duties shall be compensated at a rate of $200 per weekday night and $300 per weekend day/night.

iii. Continuing education completion beyond the minimum required hours shall be rewarded at $500 per 10 additional hours, with an annual cap of $2,500.

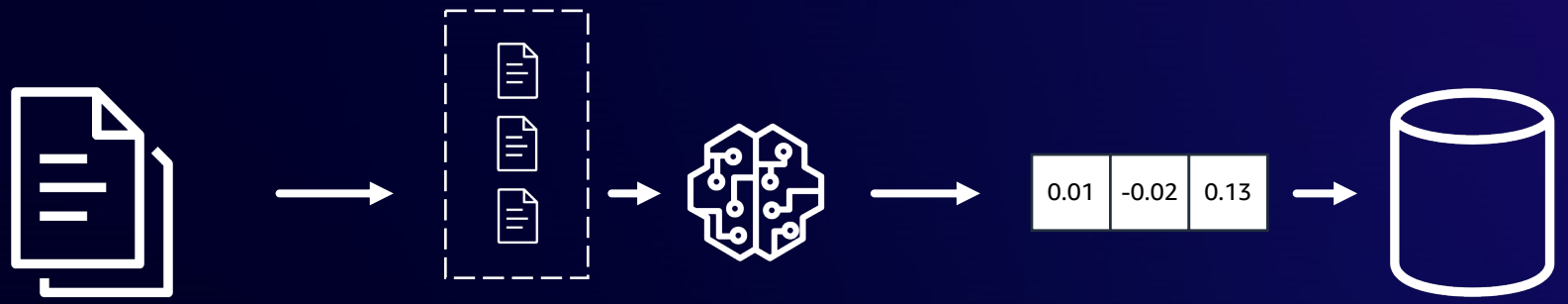
Chunking Strategy

	Fixed size	Semantic	Hierarchical	Default	No-chunking
What?	Configure chunks by specifying the number of tokens per chunk and an overlap percentage	Intelligently splits content based on meaning and context, so that related information stays together while maintaining semantic coherence	Creates a two-tiered structure of parent and child chunks, where larger sections contain smaller, related segments while preserving document organization and contextual relationships	Splits content into chunks of approximately 300 tokens, preserving sentence boundaries	Treats each document as a single text chunk
Pros	• Easy to implement • Quick processing	• Preserves meaning and context • Handle variable content	• Maintains structure • Rich context preservation to improve search relevance	• No configuration needed	• Complete control with custom implementation
Cons	• Might break semantic meaning • Less context aware	• Unpredictable chunk sizes • Variable processing time	• Requires a clear structure	• Limited control	• More development time • Higher maintenance
When?	• Uniform structure • Need predictable chunking	• Variable content structure • Context preservation is crucial	• Clear parent-child relationship	• Quick implementation • No special requirements	• Need custom processing • Special data format
Example	Logs API doc with standardized format	News/Blog article Customer support conversations	Software doc Technical Manuals Legal Documents Academic Papers	Meeting notes	Multiformat doc requiring custom processing Employee data in CSV

Putting It All Together



Putting It All Together



indexing

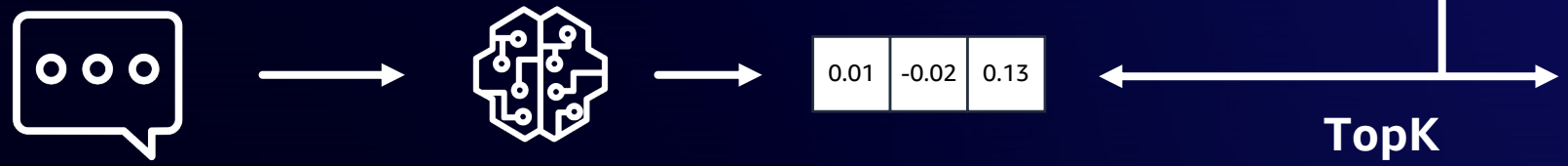
AnyCompany's data sources

Chunk

Embedding model converts files into vectors

Store vectors in database

retrieval



What is AnyCompany's vacation policy?

Embedding model converts prompt into vectors

TopK

Q: Based on the following, what is AnyCompany's vacation policy? <file_content>

A: AnyCompany's vacation policy is…

Prompt is augmented and a response is generated

Knowledge Bases for Amazon Bedrock

Fully-managed native support for retrieval augmented generation

Fully managed support for end-to-end RAG workflow

Securely connect FMs and agents to data sources

Automatically converts text documents into embeddings

Stores embeddings in your vector database

Retrieves embeddings and augments prompts

Provide source attribution

Building RAG-based Apps on Amazon Bedrock

Choose your favorite knowledge base or vector search enabled service

NATIVELY AVAILABLE BEDROCK KNOWLEDGE BASES:

Vector engine for Amazon OpenSearch Serverless

Amazon Aurora PostgreSQL with pgvector

Pinecone

Redis Enterprise Cloud

MongoDB

AMAZON DATABASES ENABLED WITH VECTOR SEARCH:

Amazon OpenSearch Serverless

Amazon OpenSearch Service

Amazon Aurora PostgreSQL

Amazon RDS PostgreSQL

Amazon DynamoDB via zero-ETL

Amazon MemoryDB

Amazon Neptune

Amazon DocumentDB

Amazon S3 Vectors

First cloud object store with native support to store and query vectors

Sub-second

vector query performance at scale

up to 90%

lower overall costs for uploading, storing, and querying vectors

Billions

of vectors stored at low cost

Using Code

```python
import boto3

client = boto3.client('bedrock-runtime')

response = client.retrieve(
    knowledgeBaseId='string',
    retrievalQuery={
        'text': 'string'
    }
)

response = client.retrieve_and_generate(
    input={
        'text': 'string'
    },
    retrieveAndGenerateConfiguration={
        'type': 'KNOWLEDGE_BASE',
        'knowledgeBaseConfiguration': {
            'knowledgeBaseId': 'string',
            'modelArn': 'string',
        }
    }
)
```

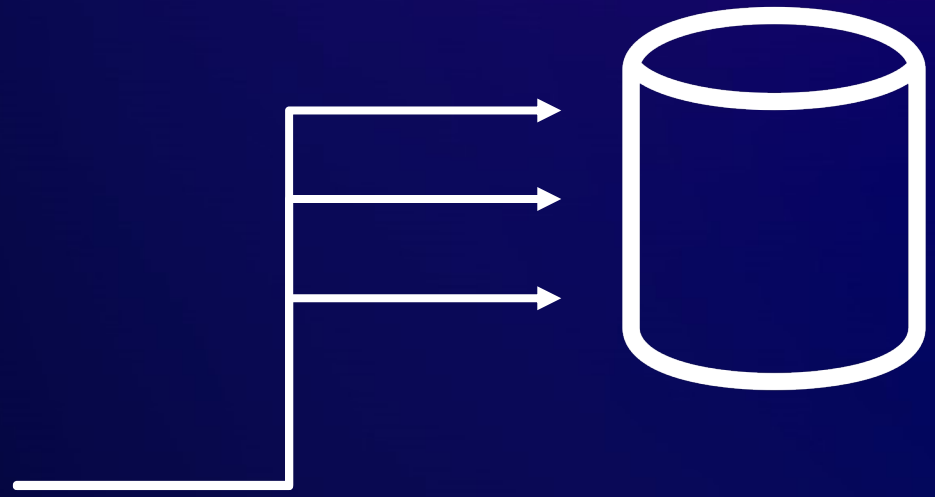
Using Code

```python
import boto3

client = boto3.client('bedrock-runtime')

response = client.retrieve(
    knowledgeBaseId='string',
    retrievalQuery={
        'text': 'string'
    }
)

response = client.retrieve_and_generate(
    input={
        'text': 'string'
    },
    retrieveAndGenerateConfiguration={
        'type': 'KNOWLEDGE_BASE',
        'knowledgeBaseConfiguration': {
            'knowledgeBaseId': 'string',
            'modelArn': 'string',
        }
    }
)
```

Advanced Techniques

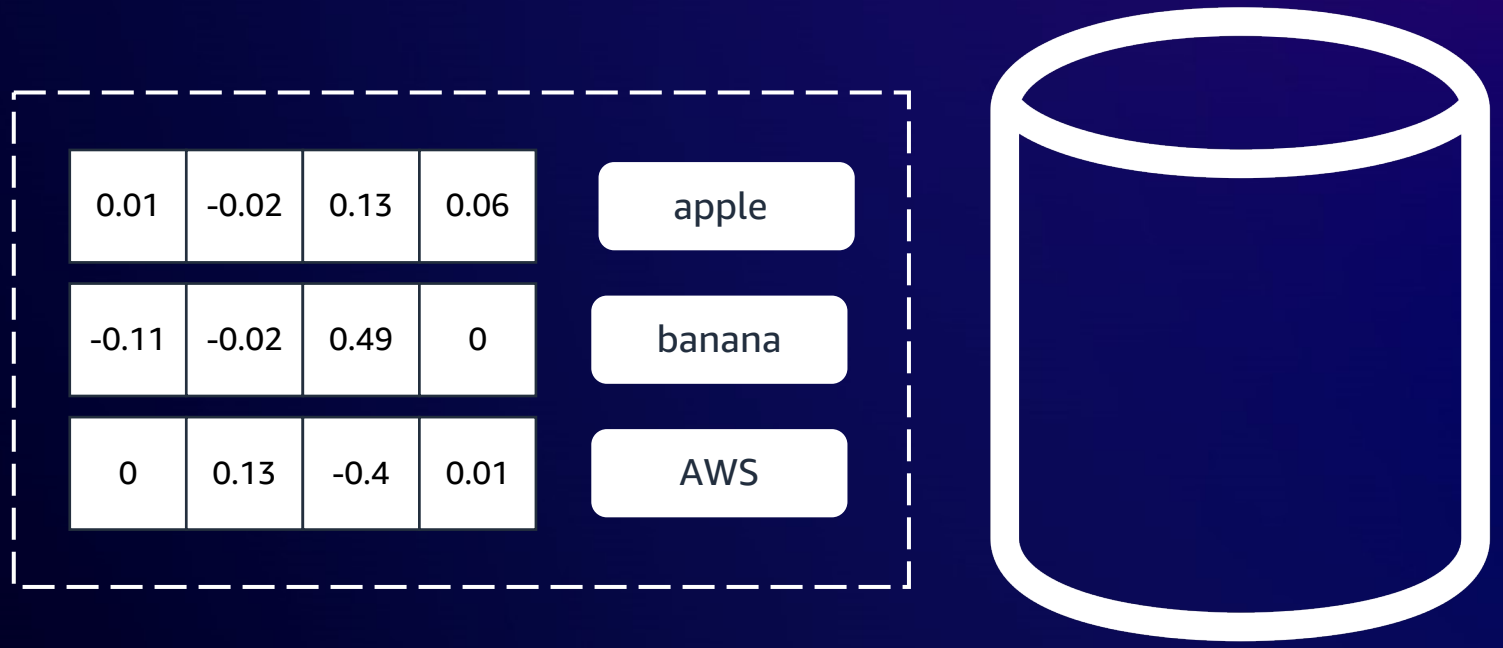
Query Reformulation

“Where is the Octank company waterfront building located and how does the whistleblower scandal hurt the company and its image?”



1. How did the whistleblower scandal affect Octank’s reputation and public image?
2. What is the whistleblower scandal involving Octank?
3. Where is the Octank Waterfront building located?

Metadata Filtering



Metadata Filtering

“Retrieve the top 2 closest vectors to [0.04, -0.02, 0.1, 0.3] only if type=food”

HyDE/Hypothetical Questions

HyDE

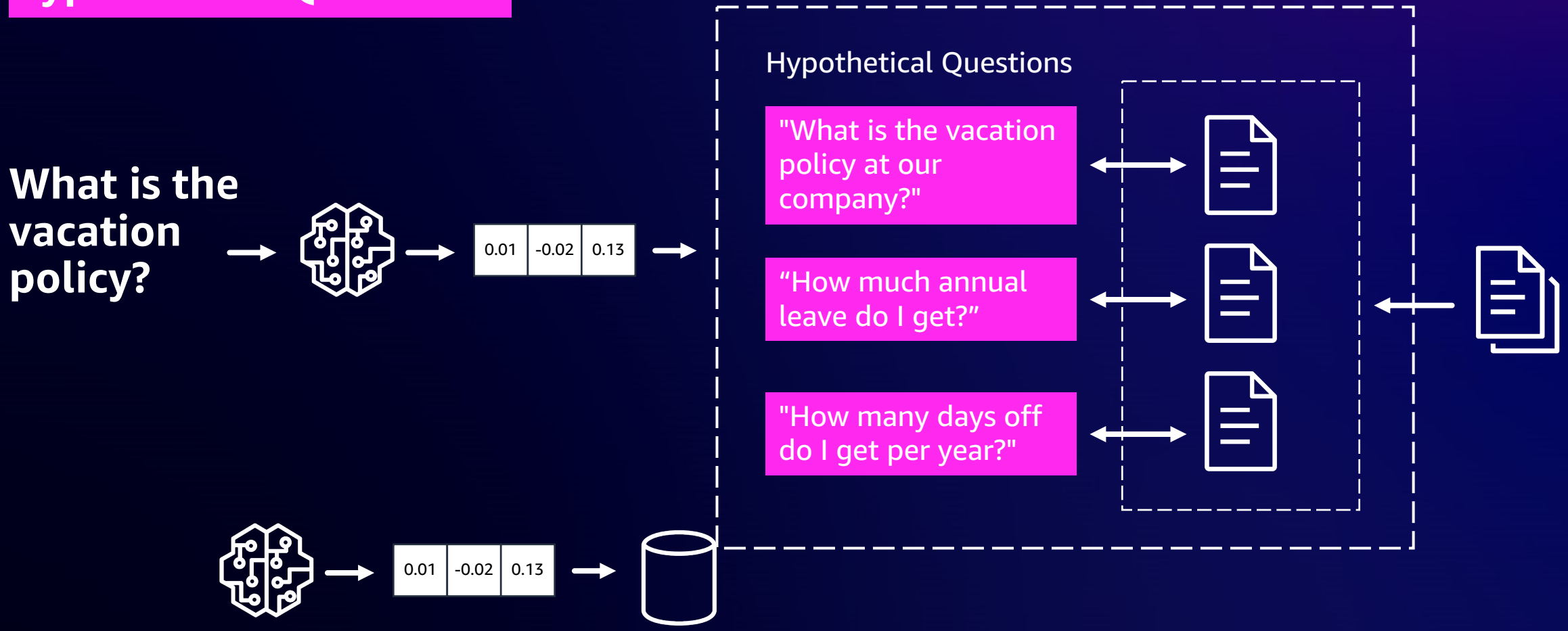
What is the vacation policy?

Vacation Day Entitlement at XYZ Corp:
At XYZ Corp vacation day entitlement is structured as follows: 1. New employees: 10 days per year 2. After 2 years of service: 15 days per year 3. After 5 years of service: 20 days per year 4. Managers and above: Additional 5 days per year Factors affecting vacation entitlement: - Length of service - Position level - Full-time vs. part-time status - Any negotiated terms in individual contracts Vacation days are accrued monthly and can be used after a 3-month probationary period. Unused vacation days can be carried over to the next year, up to a maximum of 5 days.

Hypothetical Document Embedding (HyDE)

HyDE/Hypothetical Questions

Hypothetical Questions



GENAI BOOTCAMP

Notebook Demo

Unstructured data with Amazon Bedrock Data Automation

The multi-modal challenge

80% of data is unstructured in document, image, video, and audio formats, while…

<20% of organizations are able to take advantage of unstructured data at scale

Diverse formats and variations in asset types

Hard to get the desired accuracy

Lack of standardized tooling

ML is promising but requires specialized expertise

Complex postprocessing to adapt and integrate ML output with downstream systems

Amazon Bedrock Data Automation

A generative AI-powered capability to transform multi-modal unstructured content from documents, images, video, and audio into structured data easily, accurately, and at scale.

Amazon Bedrock Data Automation

Reduces development time by up to **80%** with industry leading accuracy at up to **35%** lower cost compared to 0ff the shelf FM Alternatives

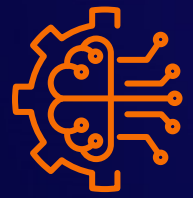
Accurate and trustworthy output helps reduce reliance on manual reviews with confidence

Simpler development effort allows you to focus on building differentiated applications at a lower cost, no ML expertise necessary

Easily tailor output requirements to diverse business-specific needs and application types

Key Features

Simple and intuitive interface to define output schemas and fine-grained business rules



Orchestration across state-of-the-art task-specific models and foundation models to generate highly accurate, consistent output

Built-in Responsible AI with visual grounding, confidence scores, and toxic content detection

Integration with Amazon Bedrock features and Knowledge base

Single inference API to handle production scale

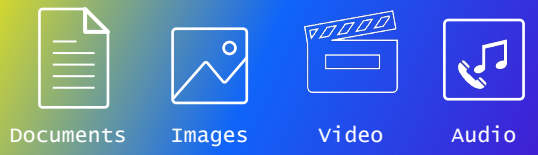
How it works

TYPES OF OUTPUT THAT BDA CAN RETURN

Standard Output

- Linearized text representation of asset
- Gen-AI optimized output: reading / viewing order, semantically related output groupings, etc.
- Controls to optimize output based on downstream systems with simple selection knobs
- Automatic modality routing based on semantic modality, *not just file type*

Supported Modalities for Standard Output



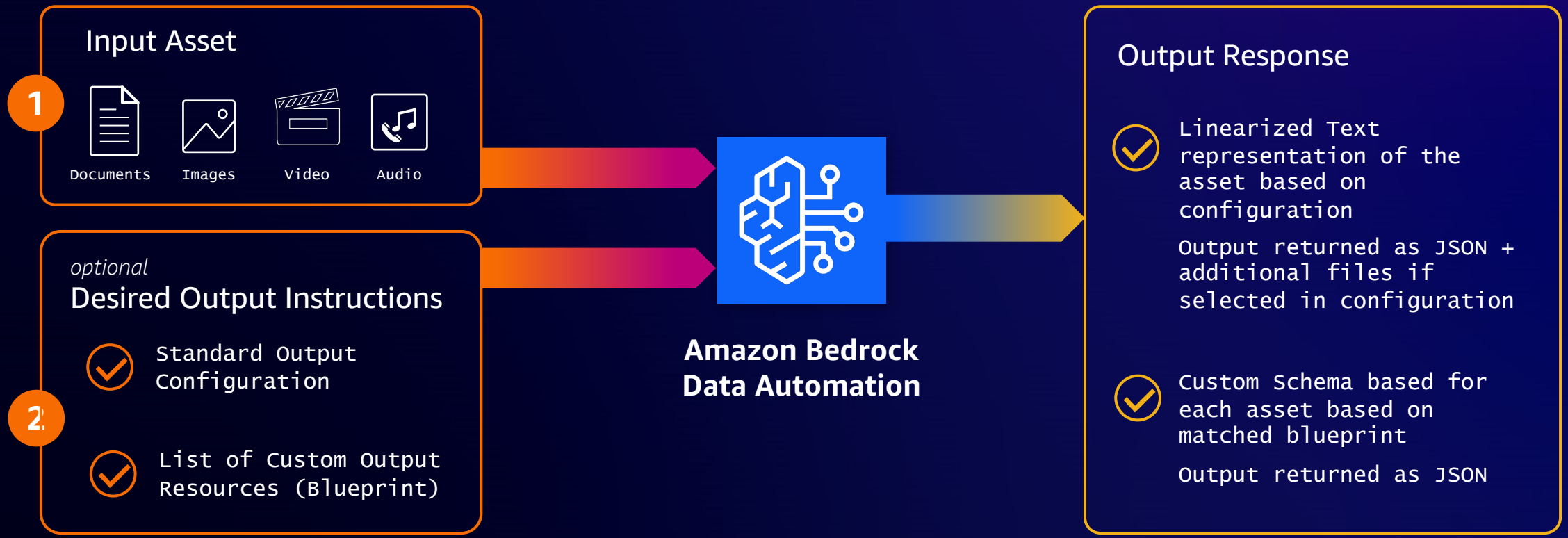
Custom Output

- Developer supplied schema based on your downstream systems (Blueprint)
- Supports tasks such as extraction, key and value normalization, transformations, reasoning, splitting and classification
- Simple interface to define business rules and task logic for each field
- Console-based assistant to create bespoke blueprints in minutes with sample and desired output description

Supported Modalities for Custom Output

Getting Started with BDA is Easy

INPUTS / OUTPUTS



Key usecases

What are the types of applications where multi-modal content processing is needed?

Intelligent Document Search

Media Asset Analysis & Monetization

Intelligent Speech Analytics

Multimodal Intelligent search

Agentic AI Workflow Automation

BDA for Intelligent Document Processing

Document Processing across Industries

Financial Services

Loan applications, Tax processing, Accounts Payable automation, Financial statement analysis, Identity verification (KYC), etc.

Healthcare & Life Sciences

Claims Processing, Patient Enrollment, Medical Billing, Grievances and Appeals, etc.

Public Sector

Small Business loans, Benefits processing, Regulatory approvals etc.

The Typical IDP Workflow

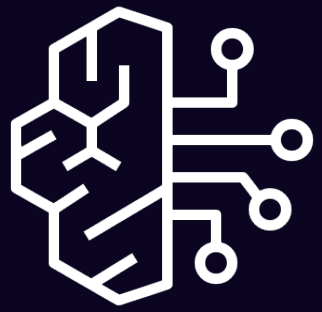
Phase 1: Data Capture / Storage

Phase 2: Doc Classification

Phase 3: Doc Extraction

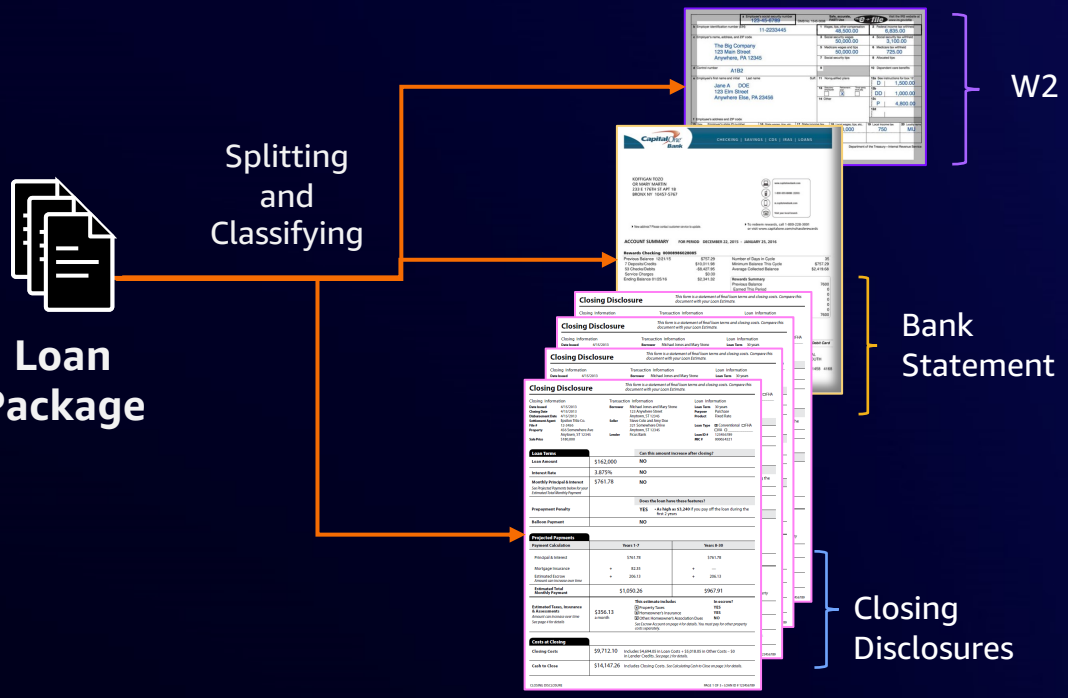
Phase 4: Enrichment

Phase 5: Validation



Amazon Bedrock Data Automation

Use Case: IDP Loan Processing




```
Employee Info
   SSN
   First Name
   Last Name
   Suffix
   Address
   Zip code

Federal Tax Info
   Income Tax
   SS Tax
   Medicare Tax
   Allocated Tips

...
```

```
Properties
   Financial Institutions
   Account Owned by
   Available Balance
   Beginning Balance
   Ending Balance
   Currently
   Account number

Account Holder Details
   First Name
   Last Name
   Address

...
```

Specific Schema and Data Points of interest by Document Type

JSON

Loan Origination Systems

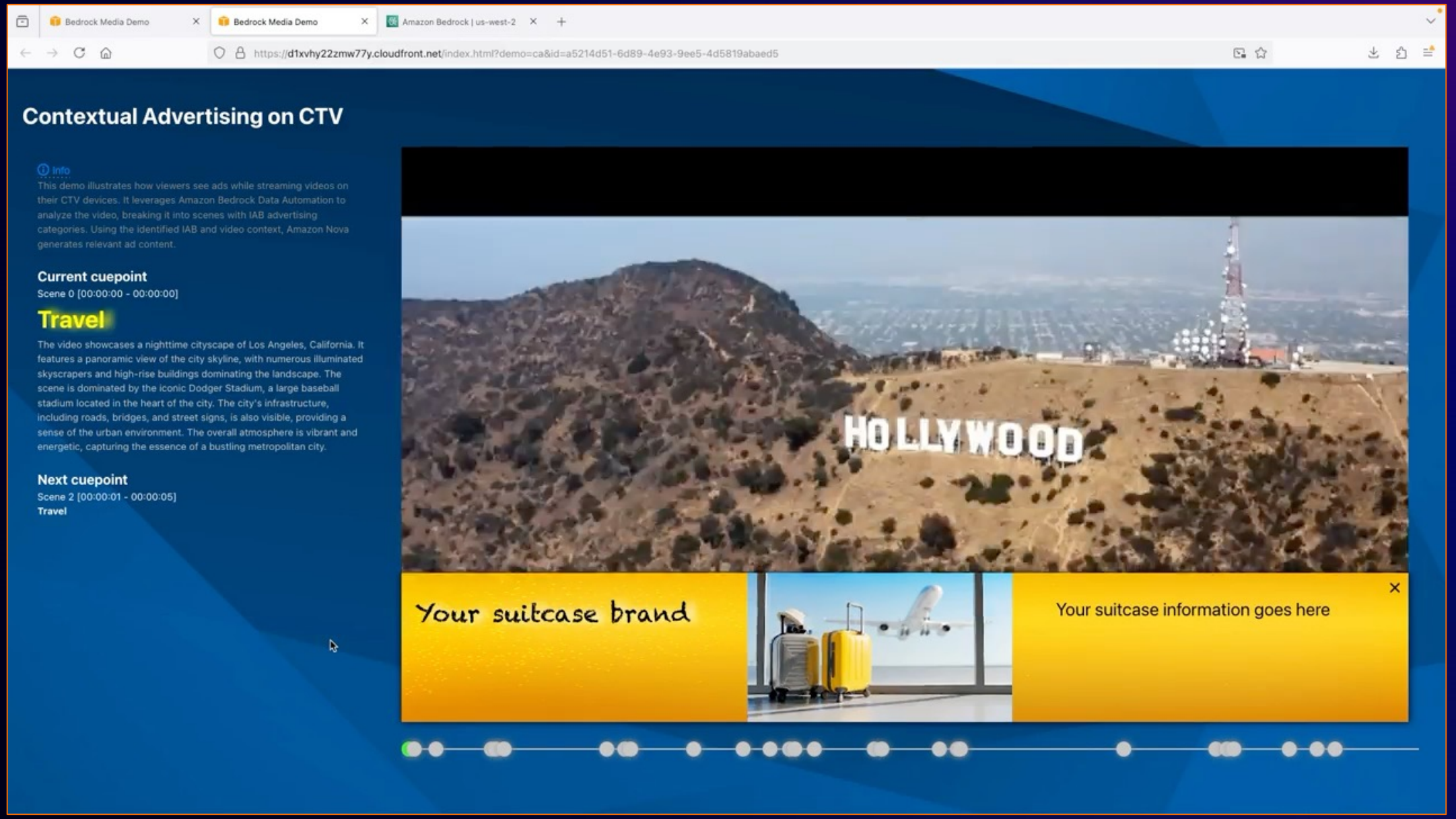
Risk Assessment platforms

Amazon Redshift

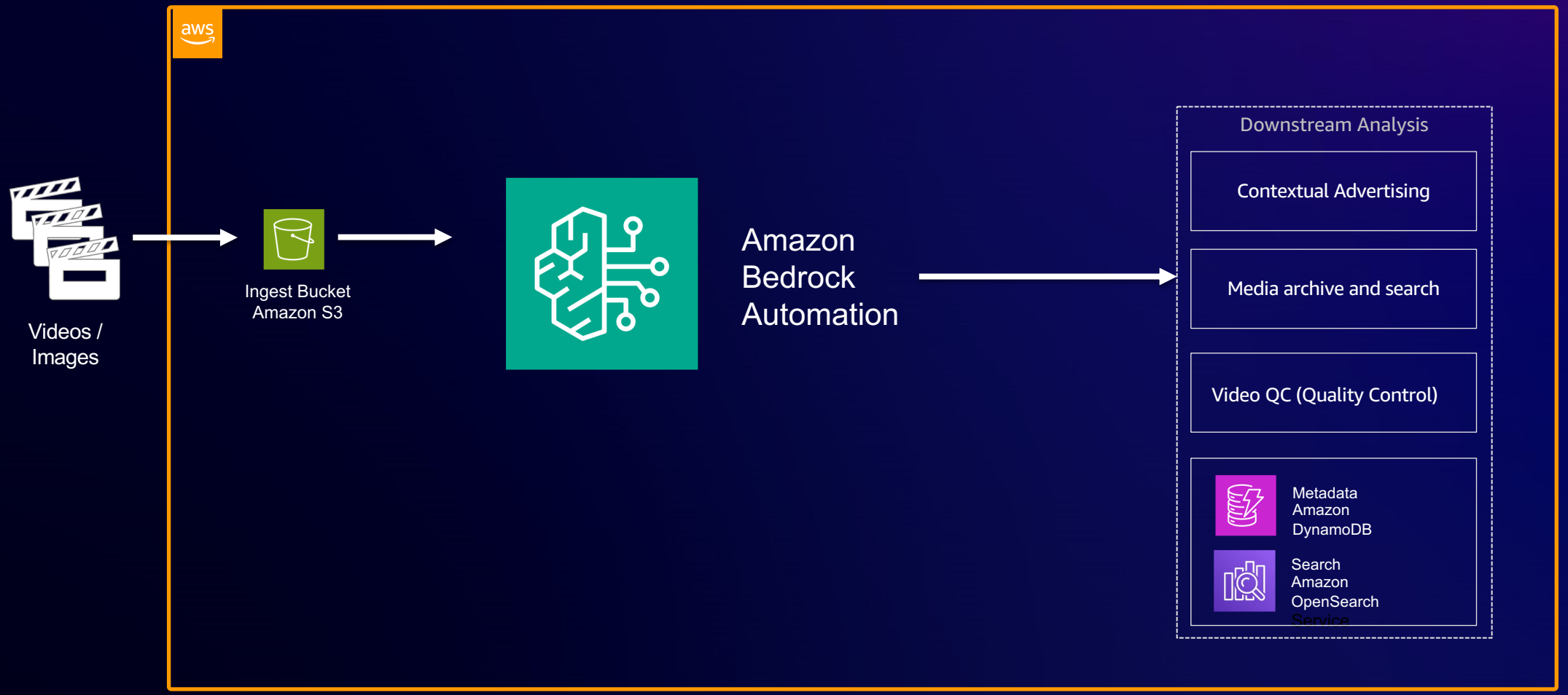
Putting it together for IDP

Demo – BDA for Media Analysis

Demo – Media Analysis

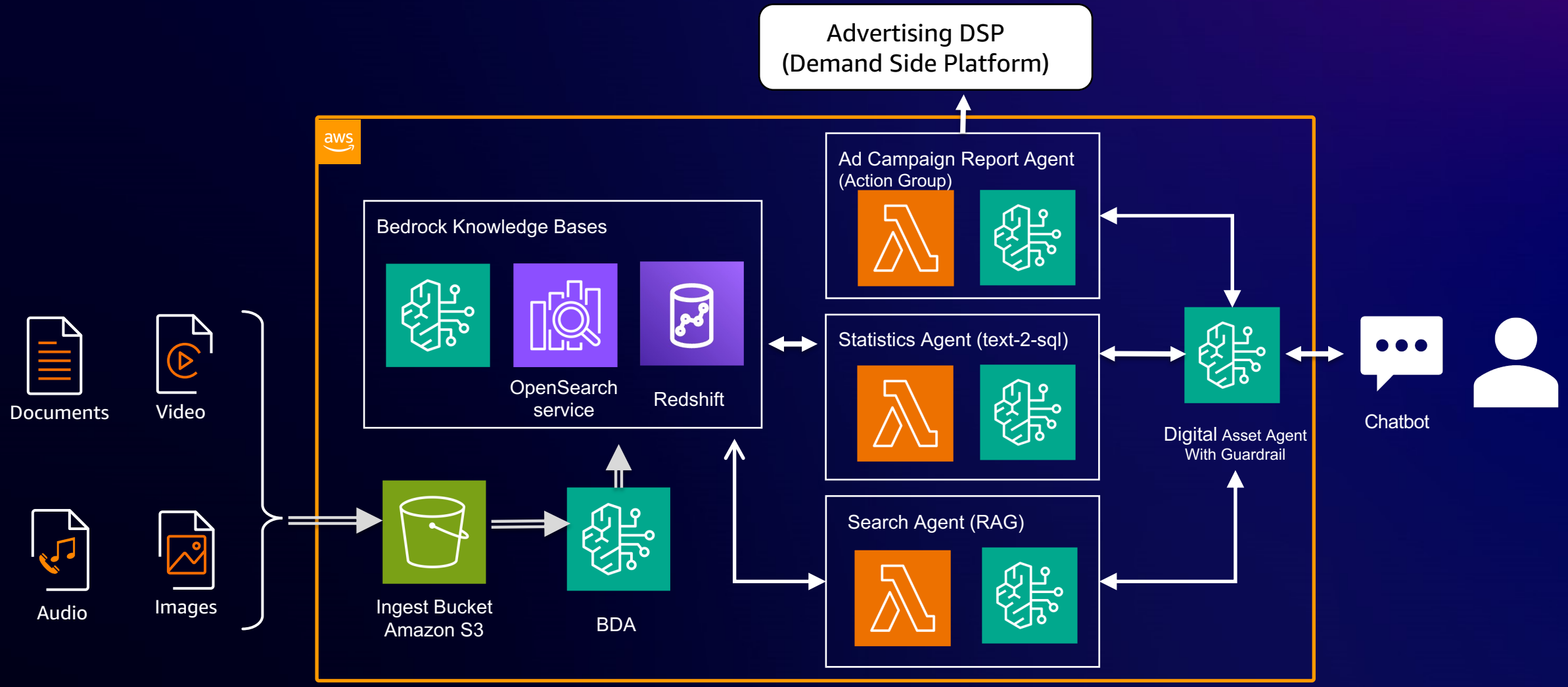


Media Analysis with BDA



From BDA to Agentic workflows:

WITH BEDROCK KNOWLEDGE BASES, MULTI-AGENTS, GUARDRAIL, AND AMAZON NOVA



GENAI BOOTCAMP

Notebook Demo

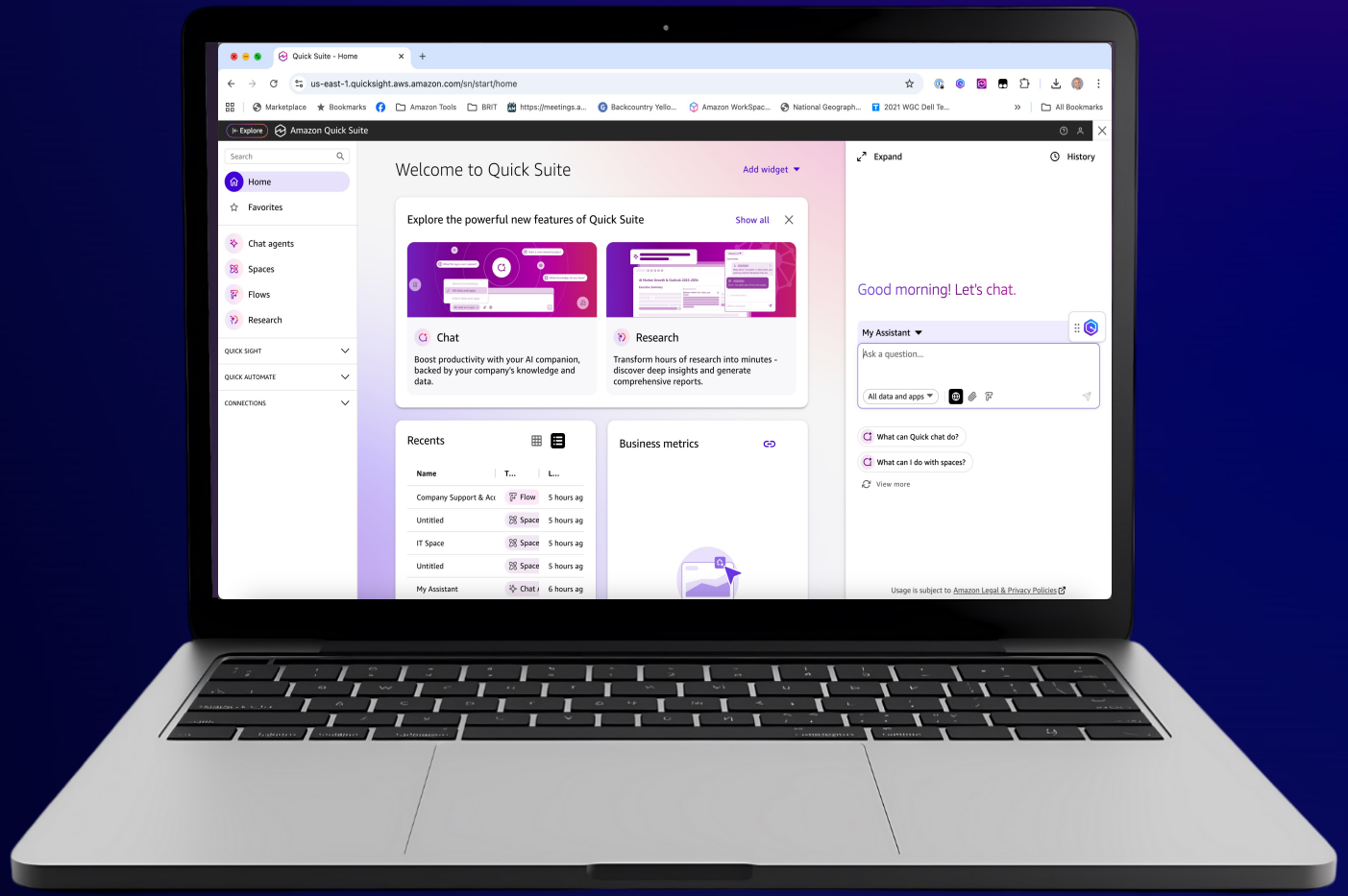
Amazon Quick Suite

Introducing
Amazon Quick Suite

Unify your data sources to get clear, context rich answers

Take action alongside agentic AI teammates

Experience an intuitive AI experience your company will love



Amazon Quick Suite

Spaces
- Organize files, dashboards, and data sources
- Collaborate workspaces for your projects

Chat Agents
- Custom AI assistants with your business knowledge
- Share and collaborate across teams

Research
- Deep-dive analysis and comprehensive reports
- Professional, exportable documentation

Quick Sight
- Business Intelligent and data visualization
- Interactive dashboards and analytics

Flows
- Automate repetitive tasks with pre-defined steps
- No-code workflow automation

Automate
- Complex, multi-step workflow automation
- Transform entire business processes

Governance, data security | Access controls | Guardrails | Responsible AI | Regulatory compliance

Company data: Spaces and datasets

40+ Data connectors

User uploaded files

Quick sight data

World Knowledge

Bedrock models

Web search

Actions

Actions in 3P apps

Save time searching thanks to Indexed Data

Quick Suite can connect to information across internal repositories, documents applications, and services

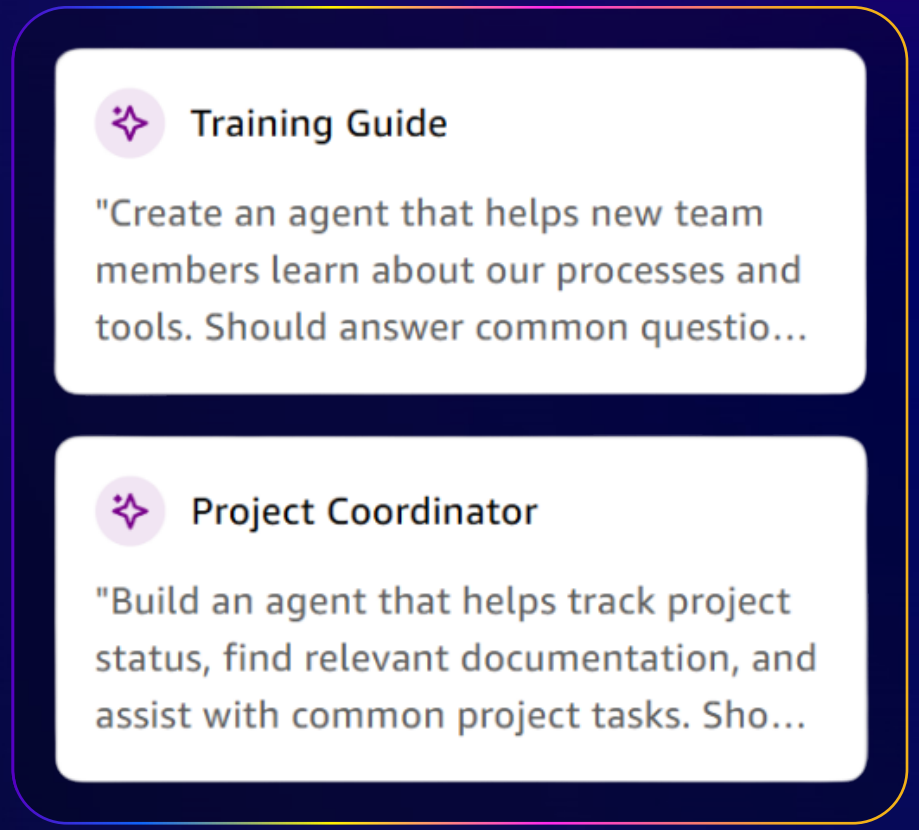
Quick Suite unites teams and agents

Create a custom chat agent that understands your unique role and your team's specific needs

Describe your team's needs in natural language

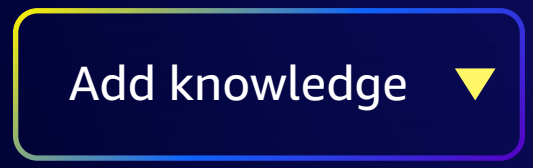
Upload your team's relevant data sets to receive accurate, context-rich answers

Get recommendations on how to improve your agent's actions and responses



Use AI to collaborate with your team with spaces in Quick Suite

Spaces allow you to create a custom knowledge hub for your team

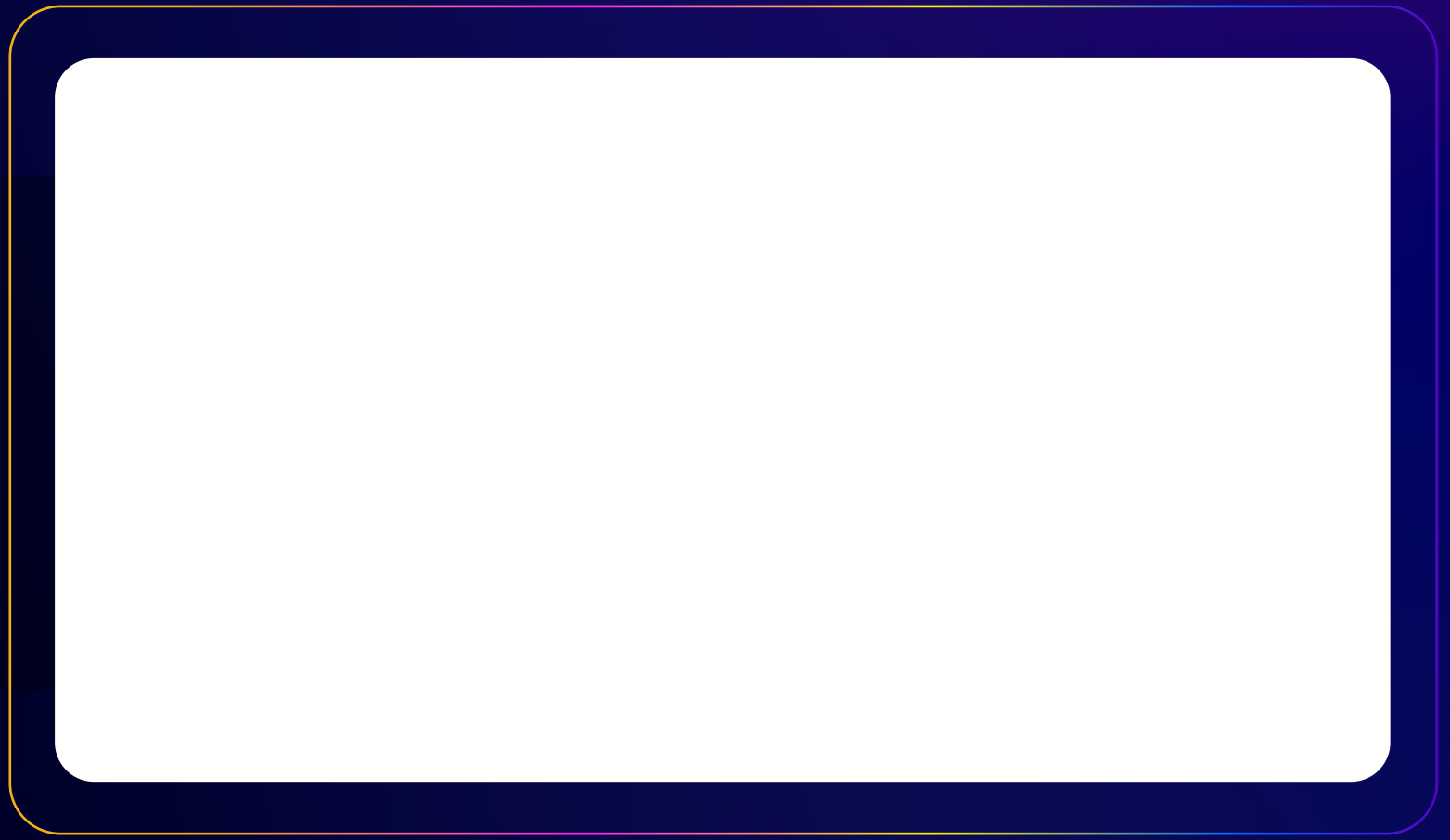


Upload your team's specialized data, documents, and resources

You can focus agents on the data contained within the space

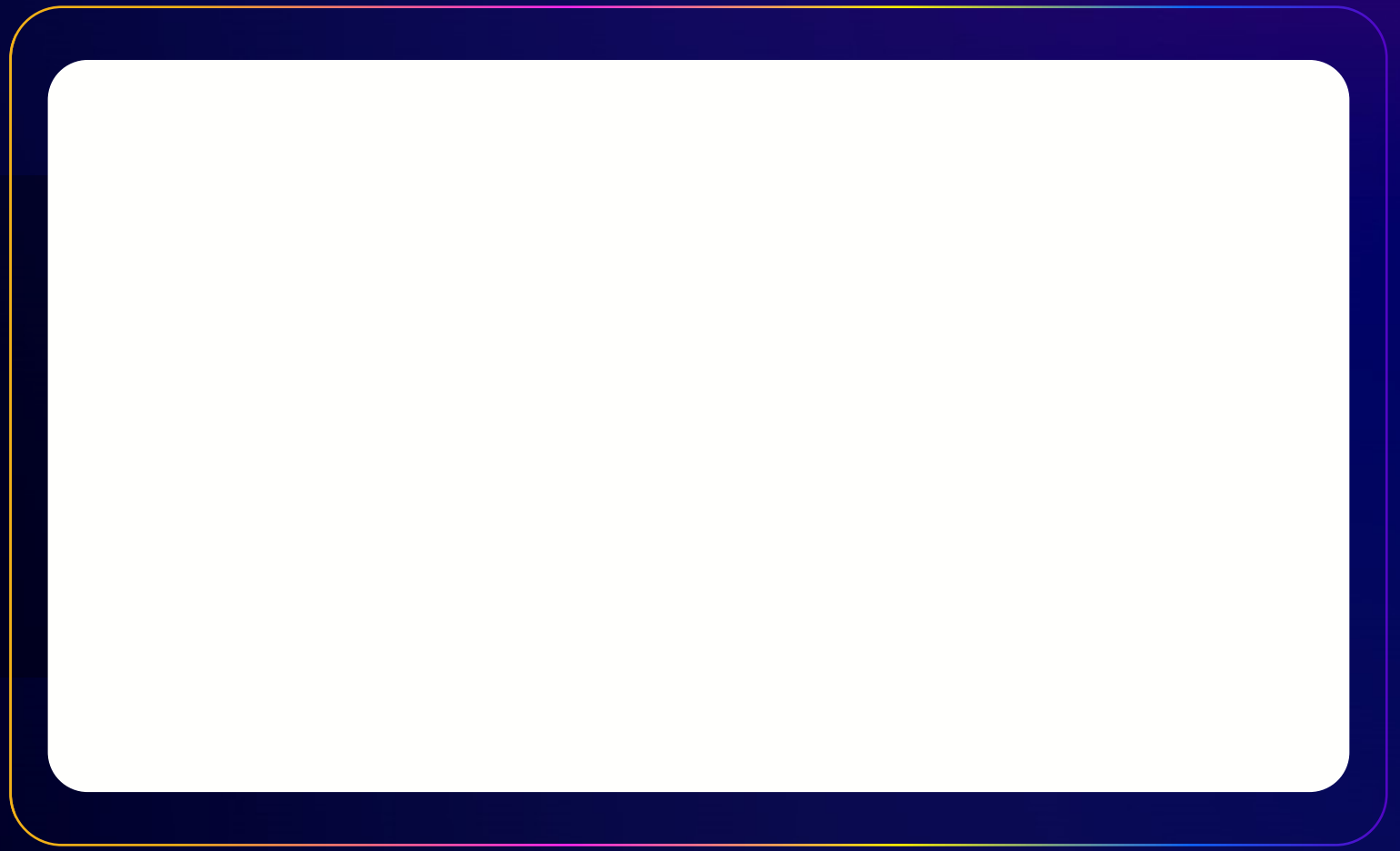
Dive deep on your business with Quick Sight

- Perform advanced data analysis
- Build dashboards
- Generate rich data stories



Become an expert on a new topic with Quick Research

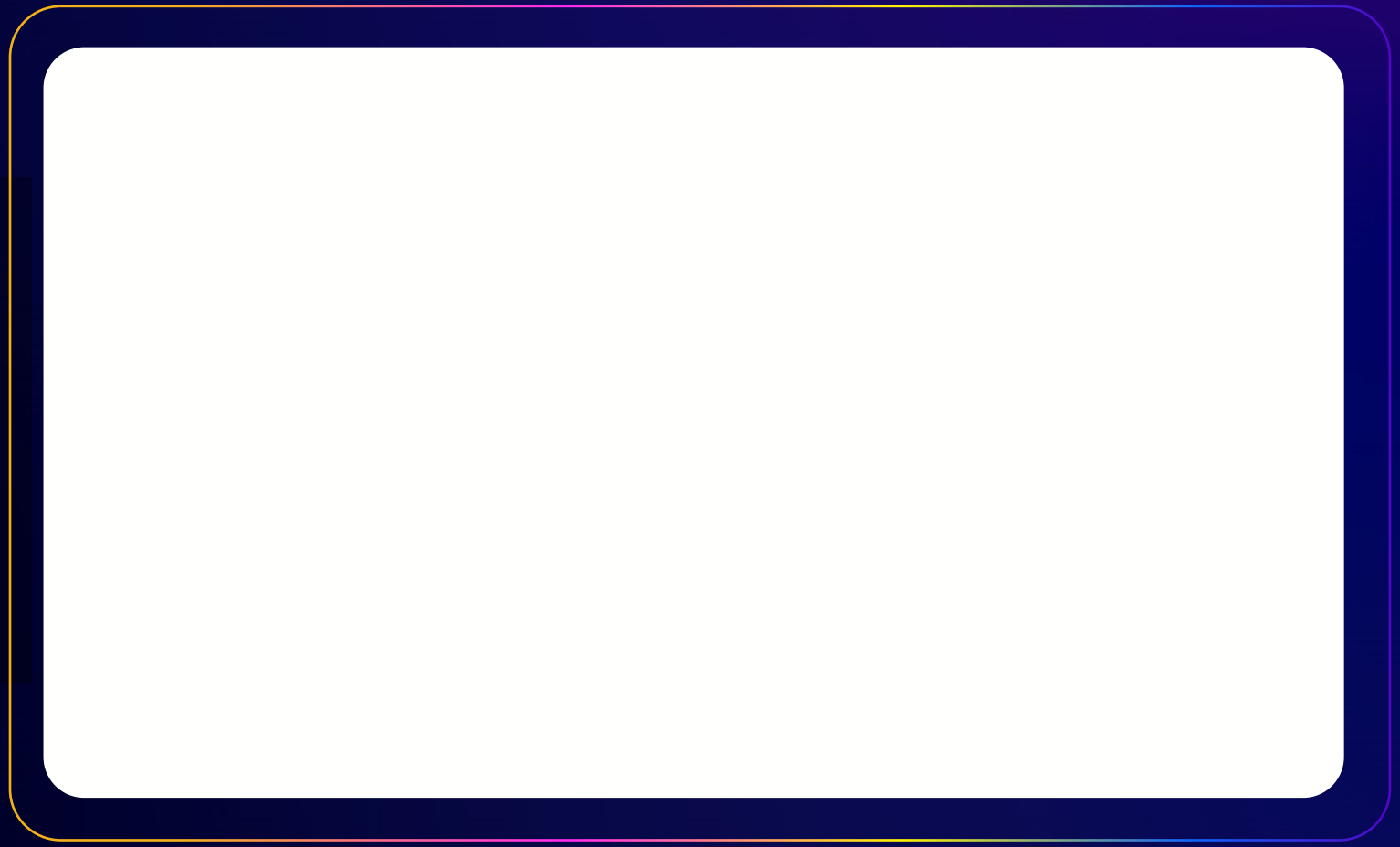
- Tackle complex and time-intensive research
- Leverage all your data, private third-party data, and data from the internet



Every user can save time on routine tasks

Quick Flows

- Save time and energy by automating repetitive tasks
- Create your own flows using natural language



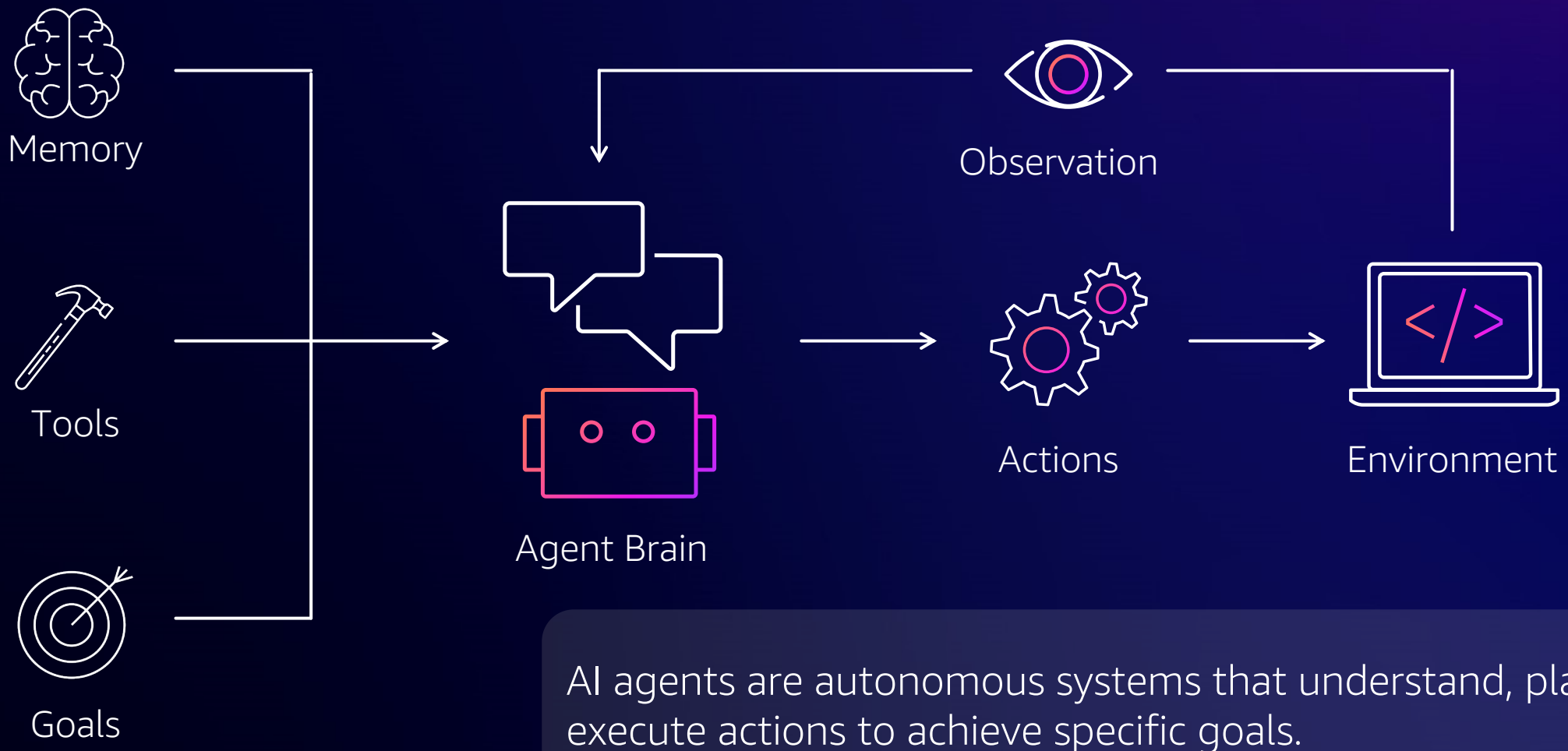
Automate complex workflows with Quick Automate

- Bring multiple agents together to streamline complex business processes

Welcome to the 2nd day

Intro to Agentic AI

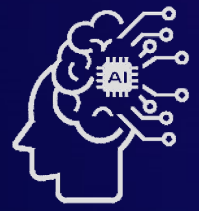
What is agentic AI?



AI agents are autonomous systems that understand, plan, and execute actions to achieve specific goals.

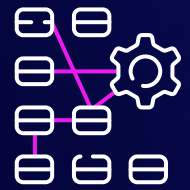
"By 2028, at least **15% of daily work decisions** will be made by **AI agents"**

Gartner, "Top Strategic Technology Trends: agentic AI – The evolution of Experience" February 2025



Model Reasoning & Tool execution

Price-performance

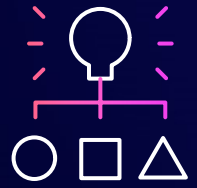


Adaptive Automation

Spectrum of Agentic AI

SIMPLE ASSISTANTS

- Basic query responses & actions
- Single step tasks
- Model as Tool caller



DETERMINISTIC AGENTS

- Multistep tasks
- Plan is predetermined
- Model as Flow Router

AUTONONMOUS AGENTS

- Self directed Agents
- **Expanded toolset – Computer/Browser**
- **Mimic human actions**

AGENTIC VIRTUAL WORKERS

- Multi-agent coordination
- Cross Org boundaries
- **Long running systems**
- **Mimic human teams**

INCREASING AUTONOMY AND BUSINESS IMPACT →

Agentic AI use cases

ACTION ENABLED CHAT AGENTS

Real-time chat interactions with API capabilities

WORKFLOW AUTOMATION AGENTS

Event-triggered task automation

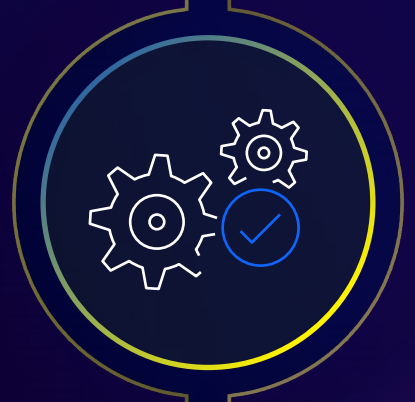
AGENTIC PLATFORMS

Create and manage reusable agents with enterprise-wide controls and governance

When to use (and not use) agentic AI

Use agentic AI for:

- Complex workflows with variable paths and decisions
- Coordination across multiple tools and data sources
- Tasks requiring understanding context and adapting responses
- Long-running processes needing memory across sessions



Traditional solutions are Better for:

- Simple, single-step automation tasks
- Ultra-low latency operations (microsecond responses)
- Mission-critical systems requiring 100% deterministic results

Agentic use cases across industries

AUTOMOTIVE

- AI-powered development and validation
- Intelligent contact center
- Supply chain optimization

ENERGY & UTILITIES

- Energy trading
- Renewable energy planning
- Upstream production optimization

FINANCIAL SERVICES

- Insurance underwriting optimization
- Intelligent customer experience
- Investment research assistant

GAMES

- AI for game production
- Code assistance for game development
- In-game AI

MANUFACTURING

- Research intelligence and discovery
- Maintenance management
- Supply chain optimization

TELECOM

- Customer experience
- Enterprise productivity
- Network automation

MEDIA & ENTERTAINMENT

- Multi-agent collaboration for video understanding
- Media planning and campaign optimization
- Multi-agent collaboration for advertising sales data

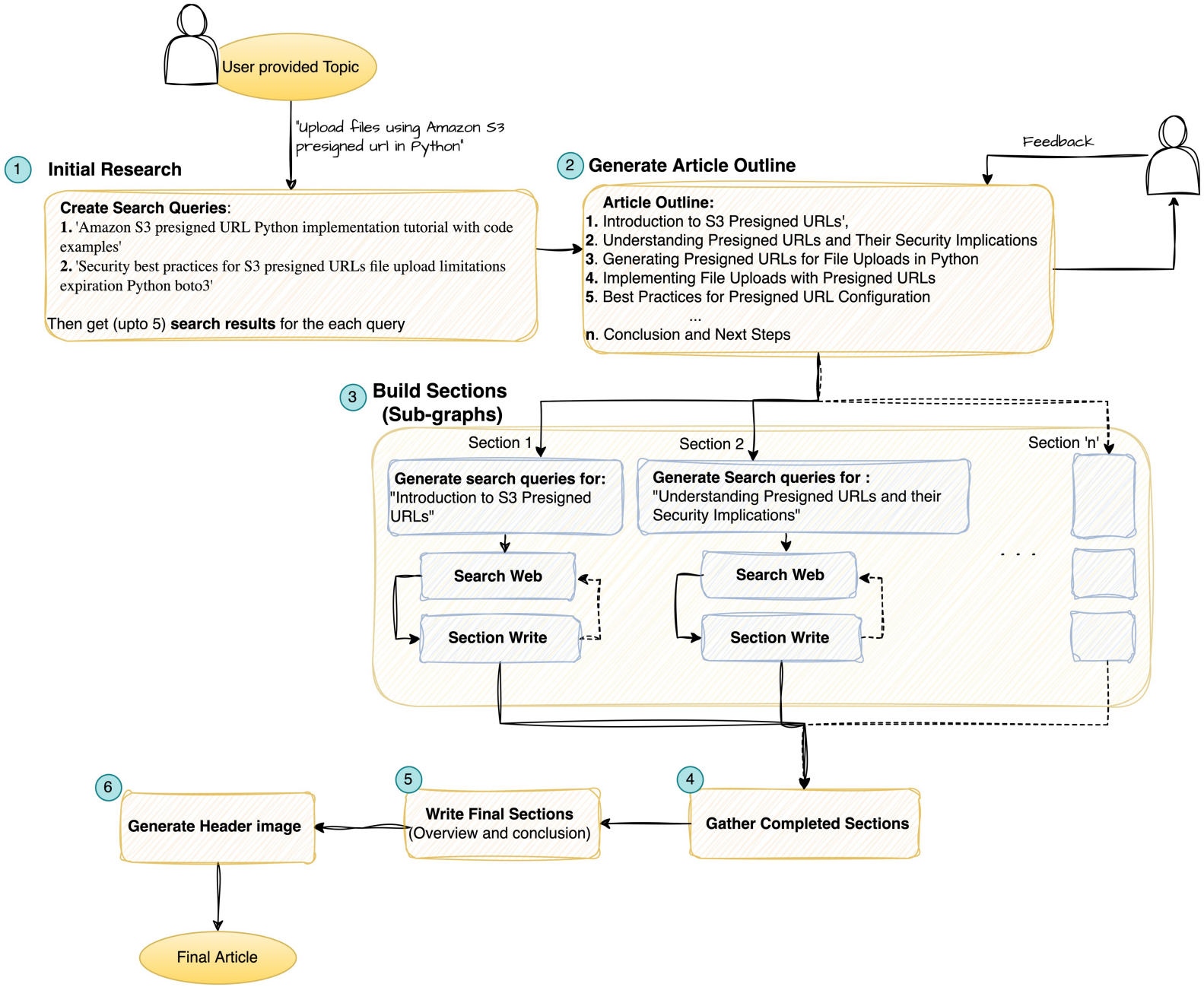
RETAIL & CONSUMER GOODS

- Agentic shopping assistants
- Supply chain reallocation
- Pricing agents

HEALTHCARE & LIFE SCIENCES

- Clinical workflow automation
- Prior authorization automation
- Patient and member engagement

Common
Deep Research Agent



https://github.com/aws-samples/sample-bedrock-deep-researcher

The broadest choice for building and deploying agents

SPECIALIZED

Amazon Q
Pre-built products that use agents for enhanced productivity

FULLY-MANAGED

Amazon Bedrock Agents
with built-in FM-powered orchestration

DIY

Strands Agents
Simple, flexible, and lightweight open source SDK for building agents

OUT-OF-THE-BOX AGENTS

TOOLS FOR BUILDING AGENTS

Example : Octank Ecommerce – Orders Assistant

Before

- Voice menus, long waits
- Heavy call center volume
- No chat experience
- Simple web site
- Limited data, static views

With multi-agent collaboration

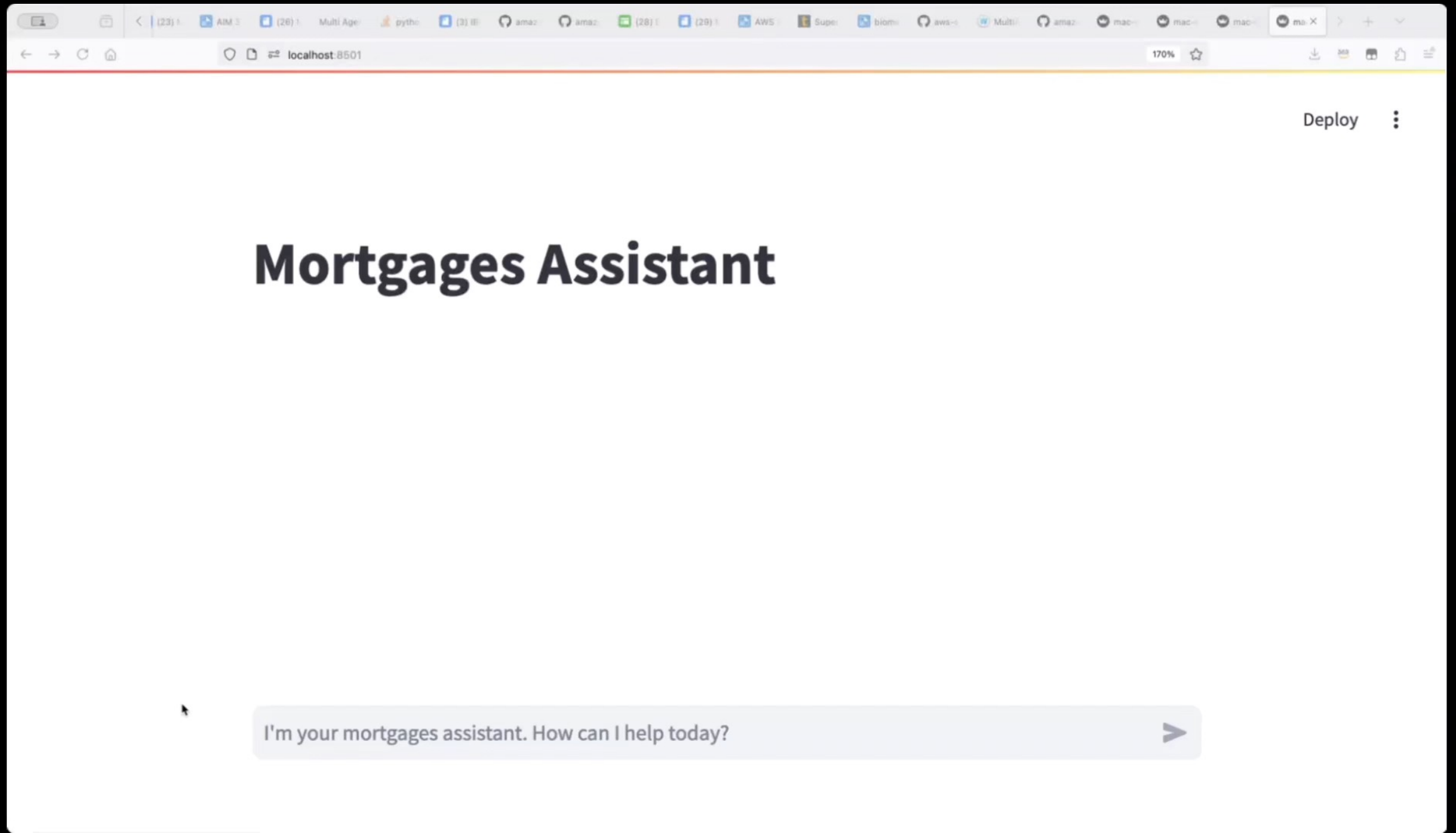
Orders Assistant

Existing Orders Q&A

Place New Orders

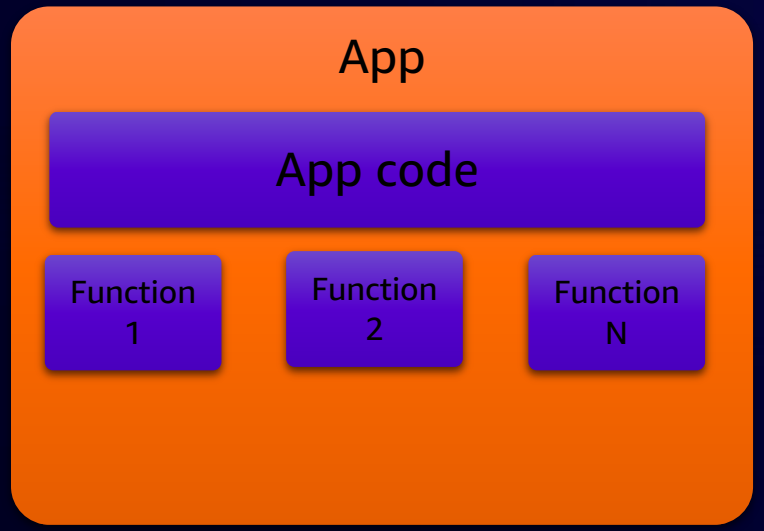
General Products Q&A

- Answers in seconds
- Flexible and personalized
- Call center relief
- Extensible


localhost:8501
170%
Deploy
Mortgages Assistant
I'm your mortgages assistant. How can I help today?

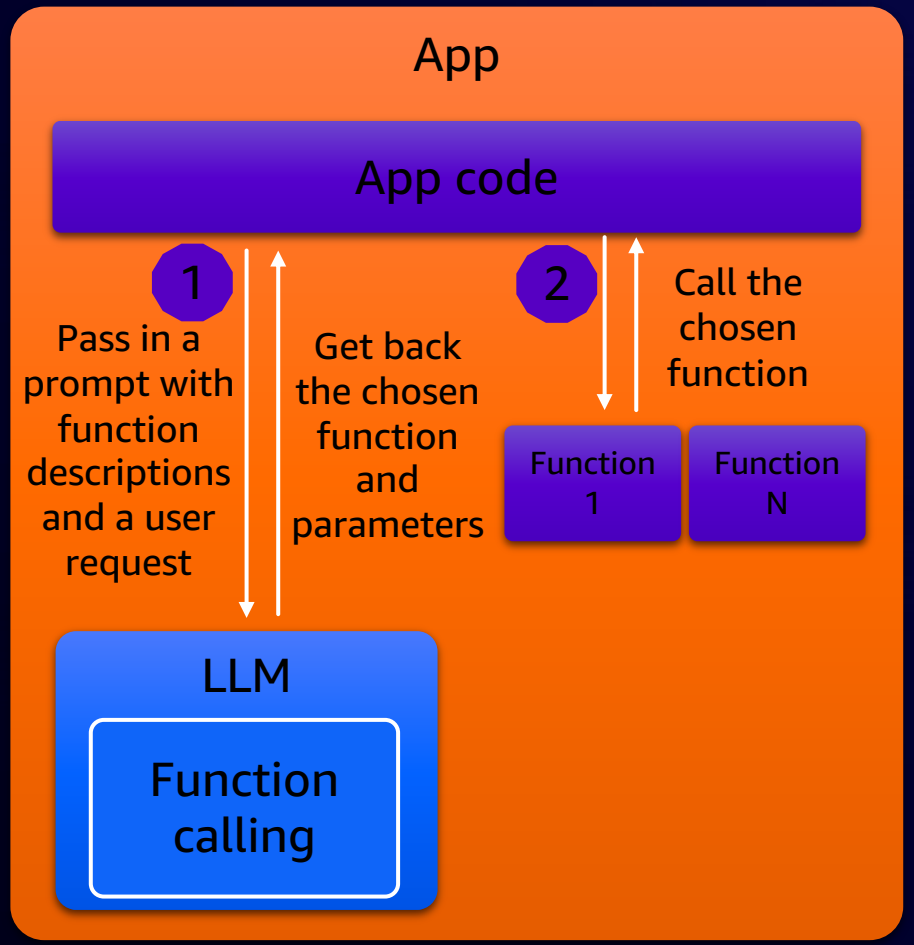
Understanding Agents

From Apps to Function calling to Agents…



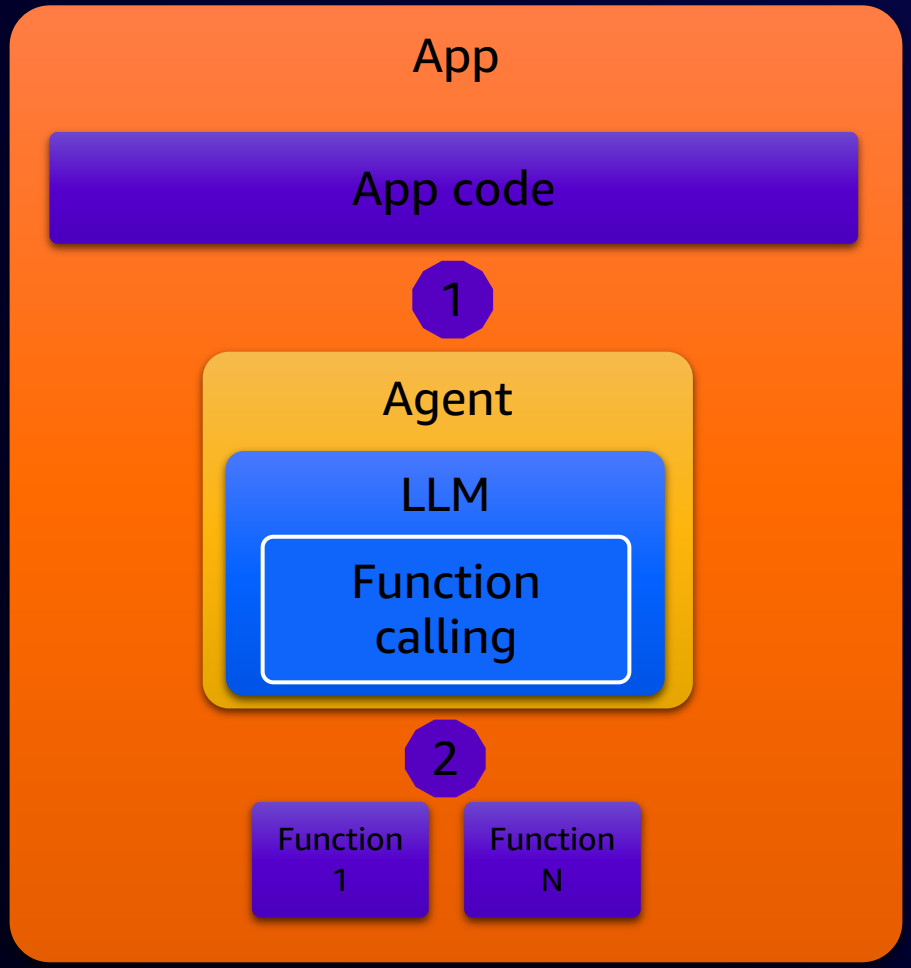
- Has been possible for decades
- Tried and true
- App is coded to accomplish specific tasks and knows what functions to call and how to call them

From Apps to Function calling to Agents…



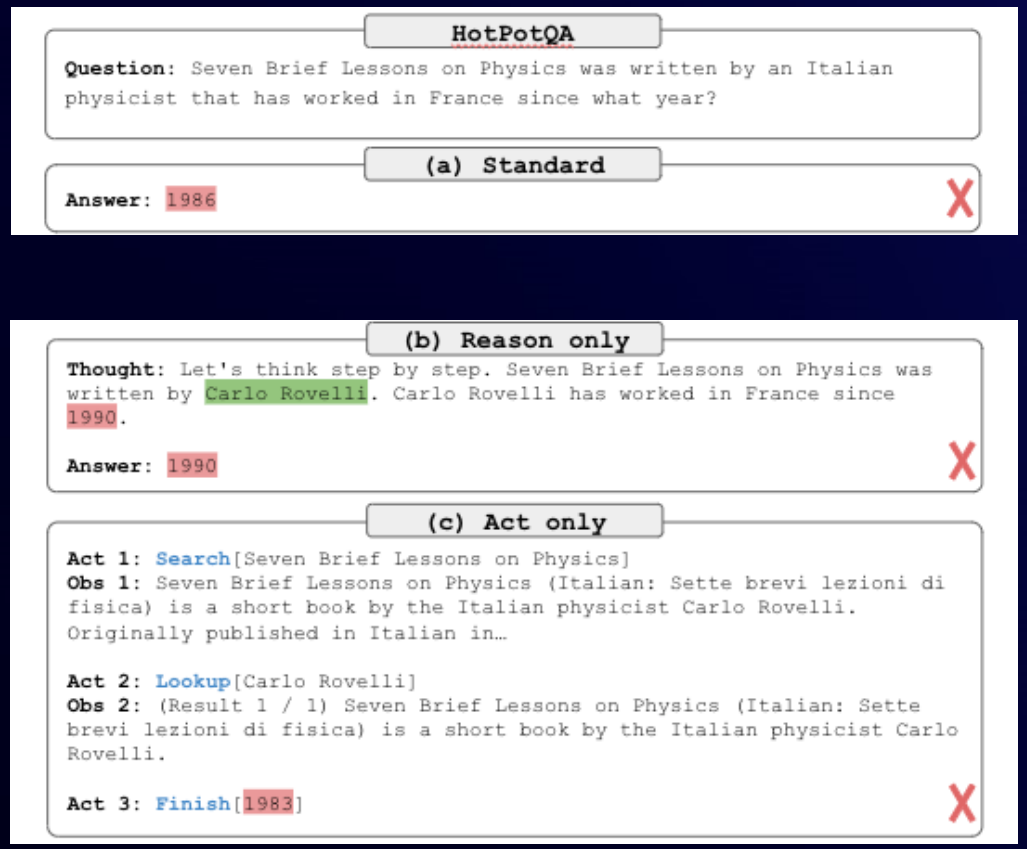
- Now we give the LLM rich descriptions of our available functions and let the LLM tell the app what to call next and with what parameters
- Now a single “app” can handle a full range of tasks without one-off coding

From Apps to Function calling to Agents…

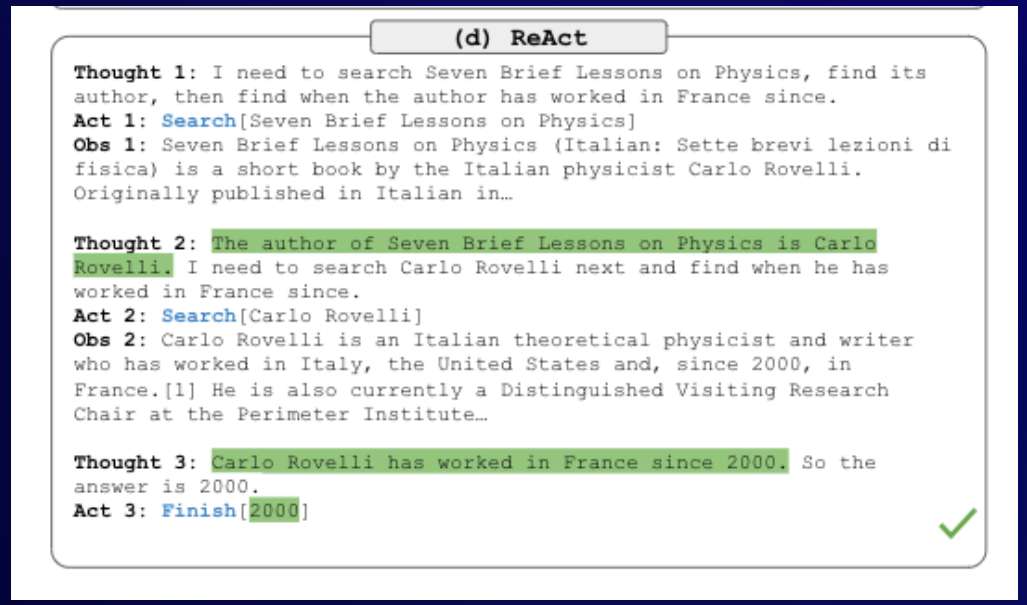


- Now I can create Agents, give them instructions and available actions, and voila, automation with even less coding and more flexibility
- Agent decides what actions, in what order, and even executes those actions
- Many possible implementations - LangChain Agents, Agents for Bedrock

ReAct (Reasoning + Action) Prompting



Combine text reasoning and actions in a single model



ReAct

ReAct is a general paradigm that combines reasoning and acting with LLMs. ReAct prompts LLMs to generate verbal reasoning traces and actions for a task.

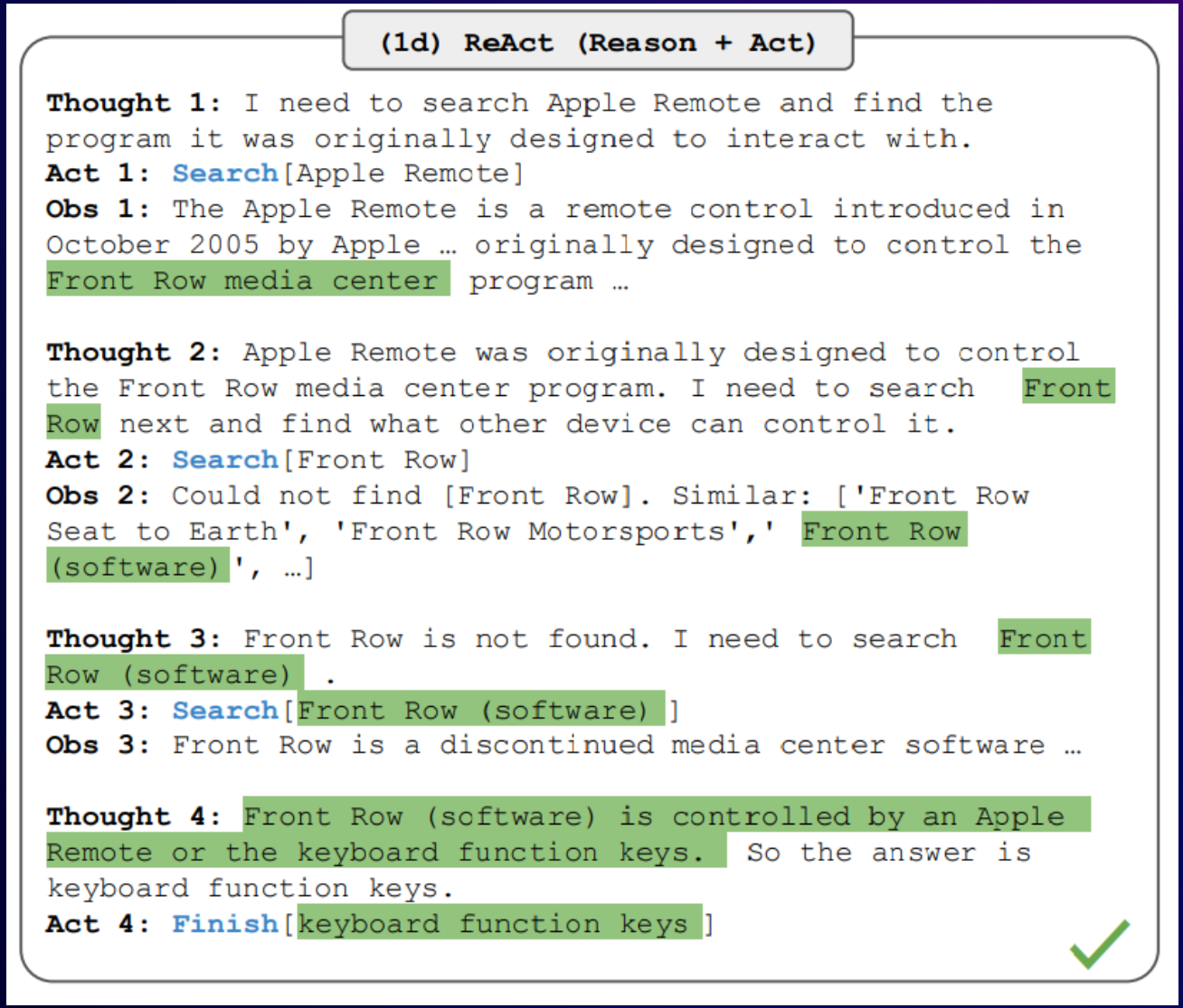


Image Source: Yao et al., 2025

ReAct Agents as a loop of LLM calls

You can think of LLM agents as a simple while loop, where an LLM decides which tools to be used, is presented with the results of that tool execution, until it finds the final answer or reaches a limit of number of iterations.

ReAct

In [3]:

```python
import json
import sys

folder = './prompts/'
prompt_file = 'fever.json'
with open(folder + prompt_file, 'r') as f:
    prompt_dict = json.load(f)

webthink_prompt = prompt_dict['webthink_simple3']

def webthink(idx=None, prompt=webthink_prompt, to_print=True):
    question = env.reset(idx=idx)
    if to_print:
        print(idx, question)
    prompt += question + "\n"
    n_calls, n_badcalls = 0, 0
    for i in range(1, 8):
        n_calls += 1
        thought_action = llm(prompt + f"Thought {i}:", stop=[f"\nObservation {i}:"])
        try:
            thought, action = thought_action.strip().split(f"\nAction {i}: ")
        except:
            print('ohh...', thought_action)
            n_badcalls += 1
            n_calls += 1
            thought = thought_action.strip().split('\n')[0]
            action = llm(prompt + f"Thought {i}: {thought}\nAction {i}:", stop=[f"\n"]).strip()
        obs, r, done, info = step(env, action[0].lower() + action[1:])
        obs = obs.replace('\\n', '')
        step_str = f"Thought {i}: {thought}\nAction {i}: {action}\nObservation {i}: {obs}\n"
        prompt += step_str
        if to_print:
            print(step_str)
        if done:
            break
    if not done:
        obs, r, done, info = step(env, "finish[]")
    if to_print:
        print(info, '\n')
    info.update({'n_calls': n_calls, 'n_badcalls': n_badcalls, 'traj': prompt})
    return r, info
```

Source: Yao et al., 2025

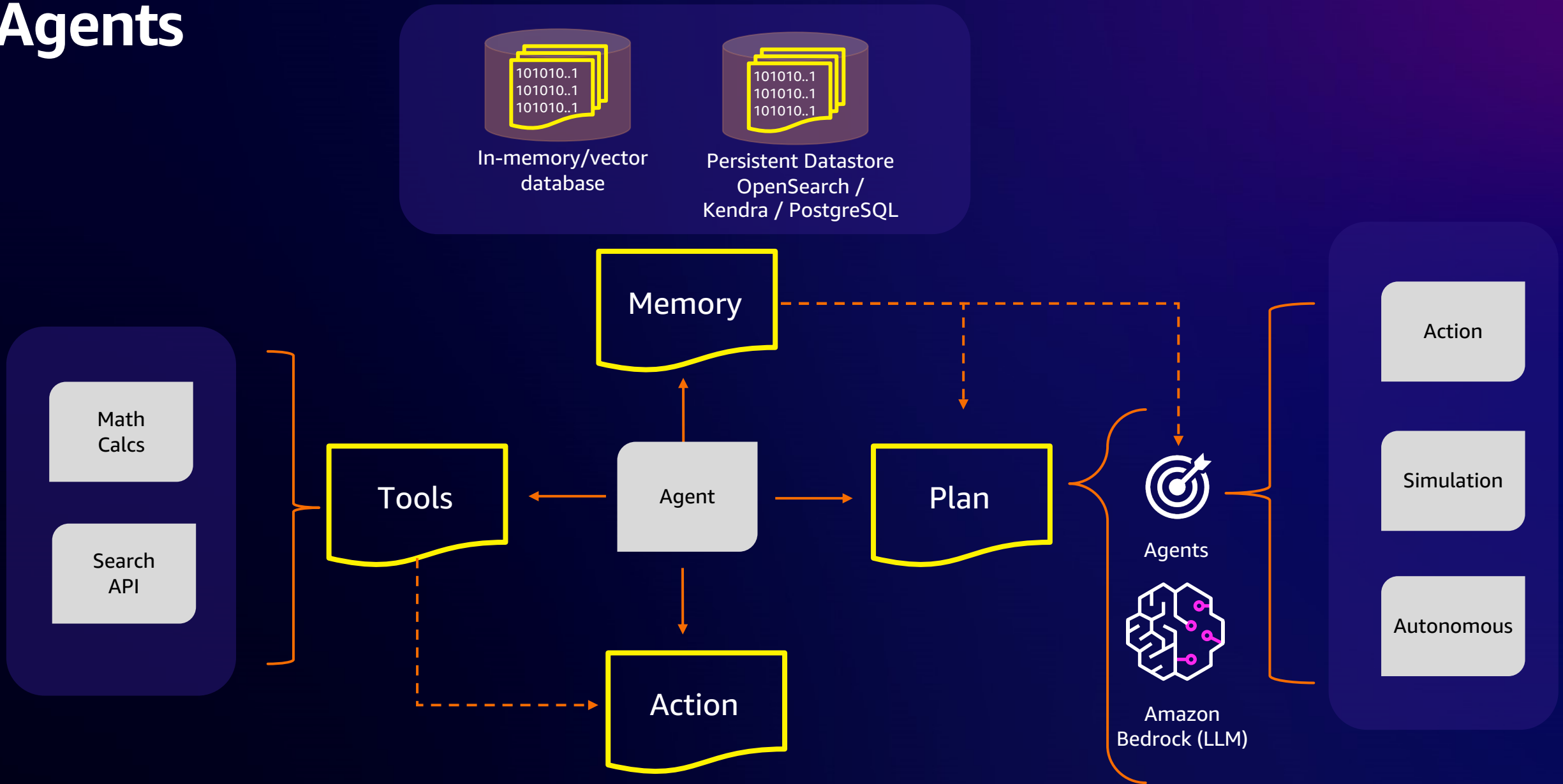
ReAct Agents as a loop of LLM calls

```
REACT_PROMPT_TEMPLATE = """
Answer the following question by thinking step by step.
For each step, follow this format:

Thought [step number]: Reason about the current state and what you need to know.
Action [step number]: Choose from: Search[query], Lookup[term], Calculate[expression], or Finish[answer]
Observation [step number]: (This will be filled with the result of your action)

Question: {question}
"""
```

You can think of LLM agents as a simple while loop, where an LLM decides which tools to be used, is presented with the results of that tool execution, until it finds the final answer or reaches a limit of number of iterations.

```python
def simple_react_agent(question, prompt_template, max_steps=3):
    """
    A simplified ReAct agent implementation

    Args:
        question: The question to answer
        prompt_template: Template with instructions for the LLM
        max_steps: Maximum number of reasoning steps
    """
    # Initialize the conversation with the formatted prompt
    history = prompt_template.format(question=question)

    for step in range(1, max_steps + 1):
        # Get the next thought and action from LLM
        response = llm(history + f"\nThought {step}:")

        # Parse thought and action from response
        thought_action = response.strip()
        thought = thought_action.split(f"Action {step}:")[0].strip()
        action = thought_action.split(f"Action {step}:")[1].strip() if f"Action {step}:" in thought_action else ""

        # Execute the chosen tool based on the action
        if "Search[" in action:
            tool_input = action.split("Search[")[1].split("]")[0]
            observation = search_tool(tool_input)
        elif "Finish[" in action:
            return action.split("Finish[")[1].split("]")[0]
        else:
            observation = "Error: Invalid action format"

        # Update history with the complete step
        history += f"\nThought {step}: {thought}\n"
        history += f"Action {step}: {action}\n"
        history += f"Observation {step}: {observation}\n"

        # Check if we've reached a conclusion
        if "Finish[" in action:
            break

    return "Failed to reach conclusion within step limit"
```

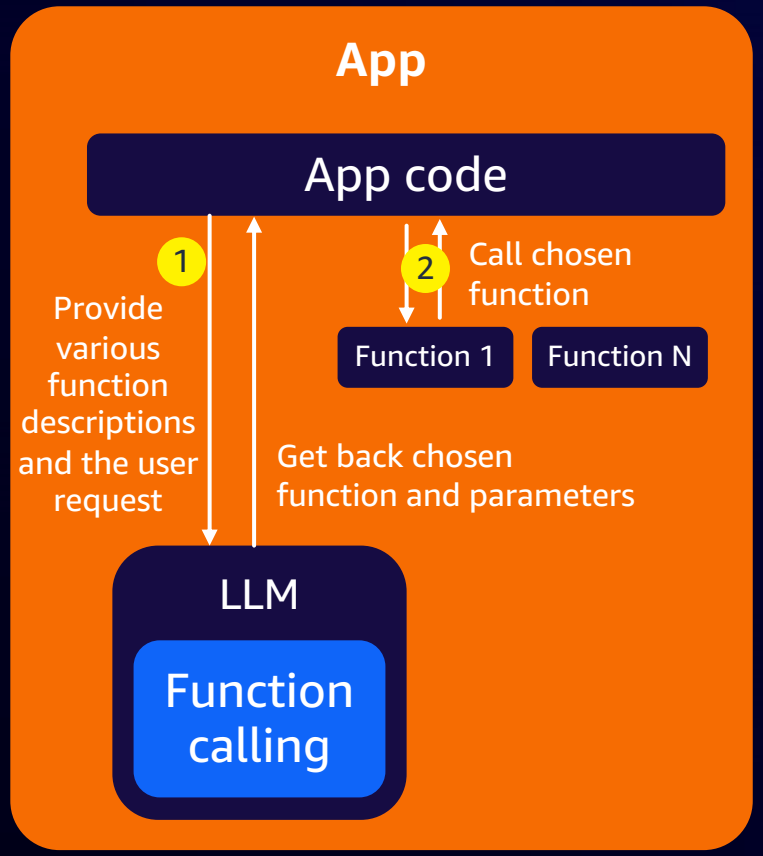
Agents



Function calling, Agents, and MCP

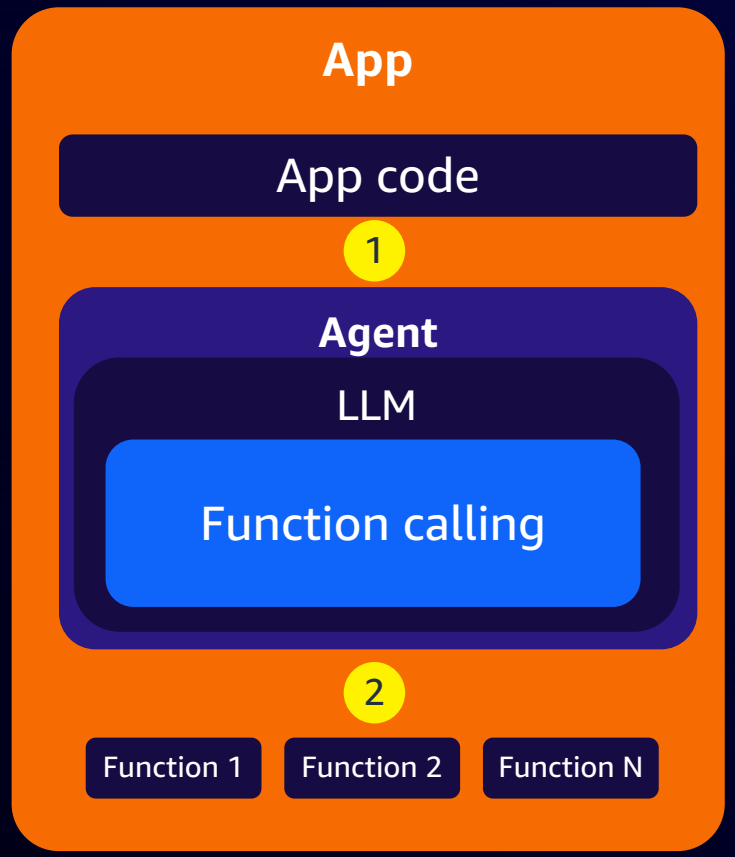
Function calling

LLMs decides which functions to call and how

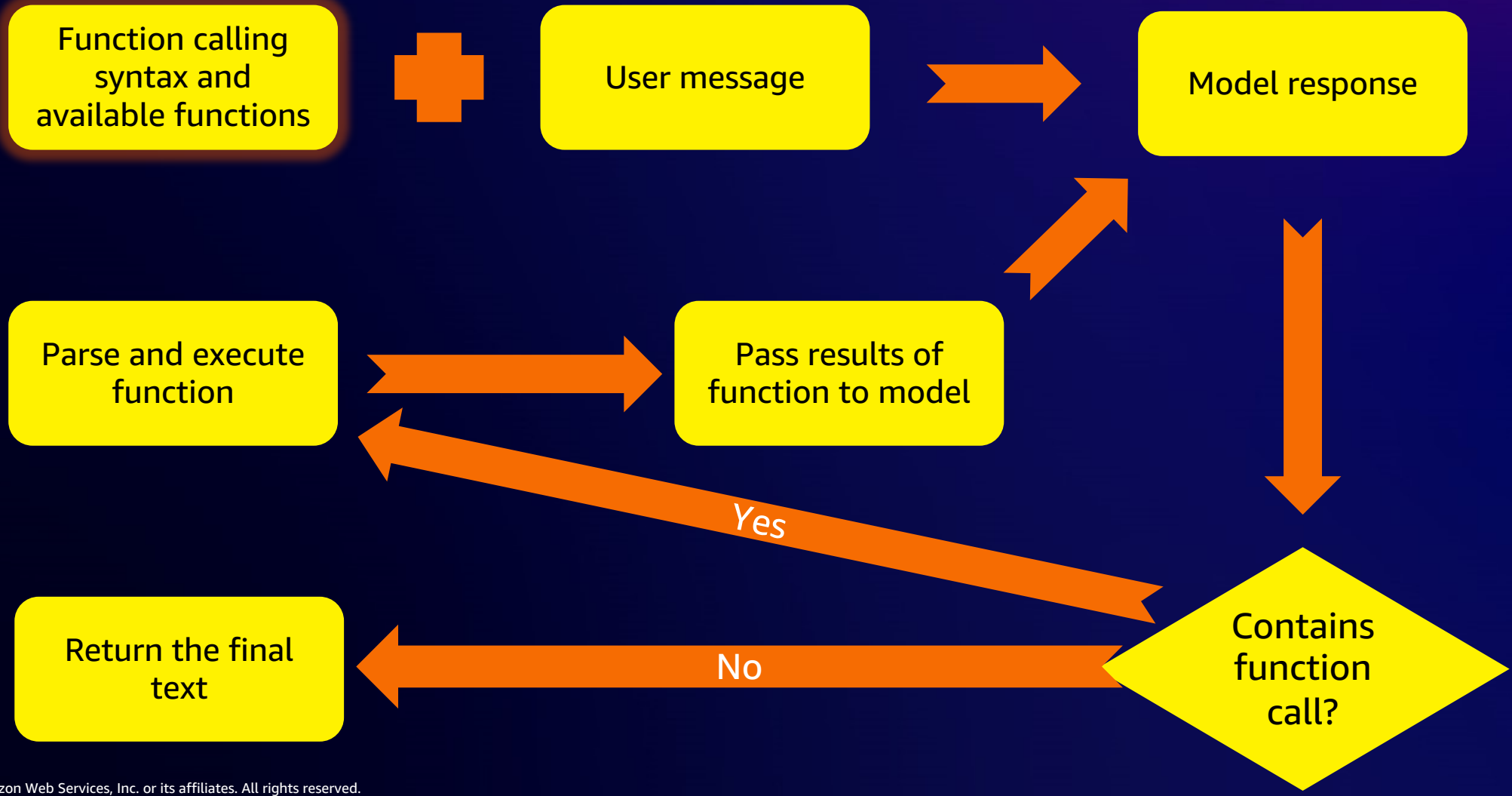


Agents

Let Agent chooses the functions, and also execute them



How function calling works: A closer look



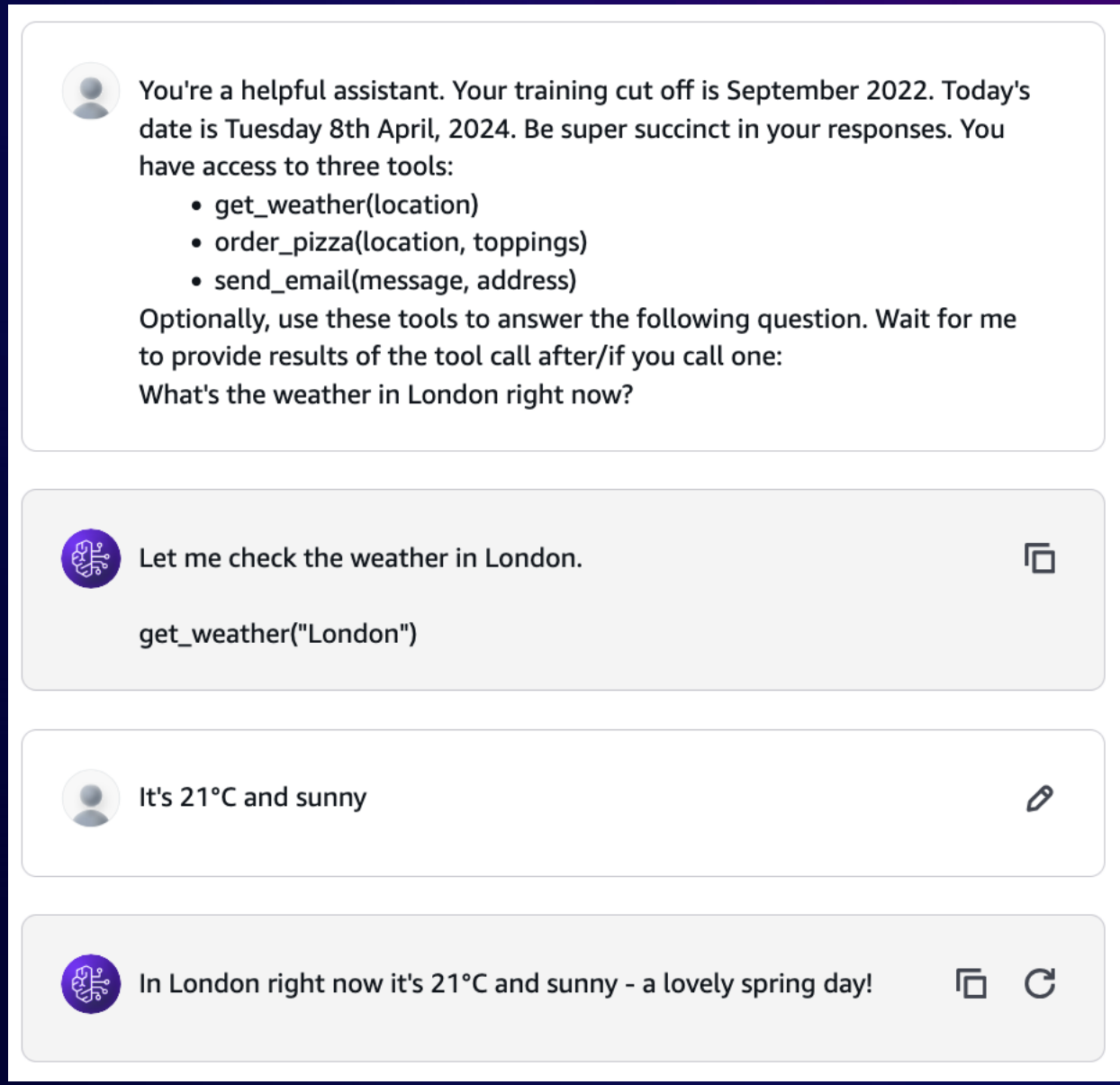
In detail..

WE DEFINE THE FUNCTIONS LLM CAN SELECT FROM

LLM selects a function

We pass the results back to LLM

LLM uses the results to create its answer

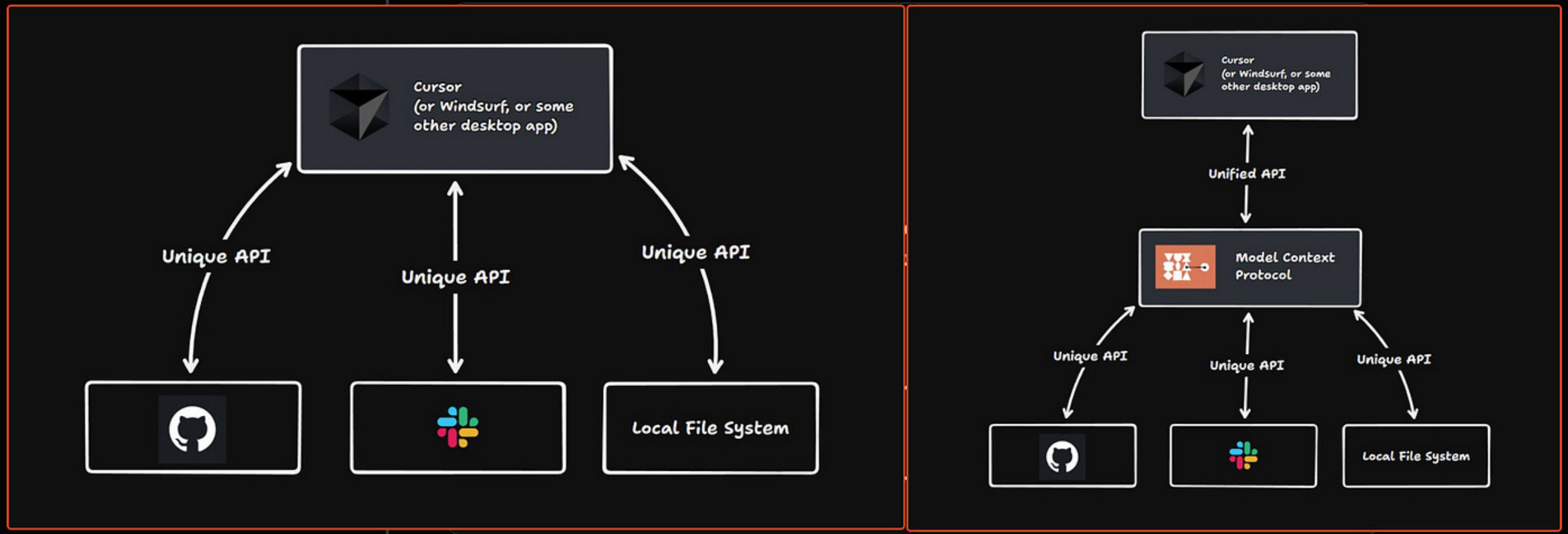


MCP

Matt Pocock @mattpocockuk · Mar 6

Lots of folks feeling FOMO about MCP.

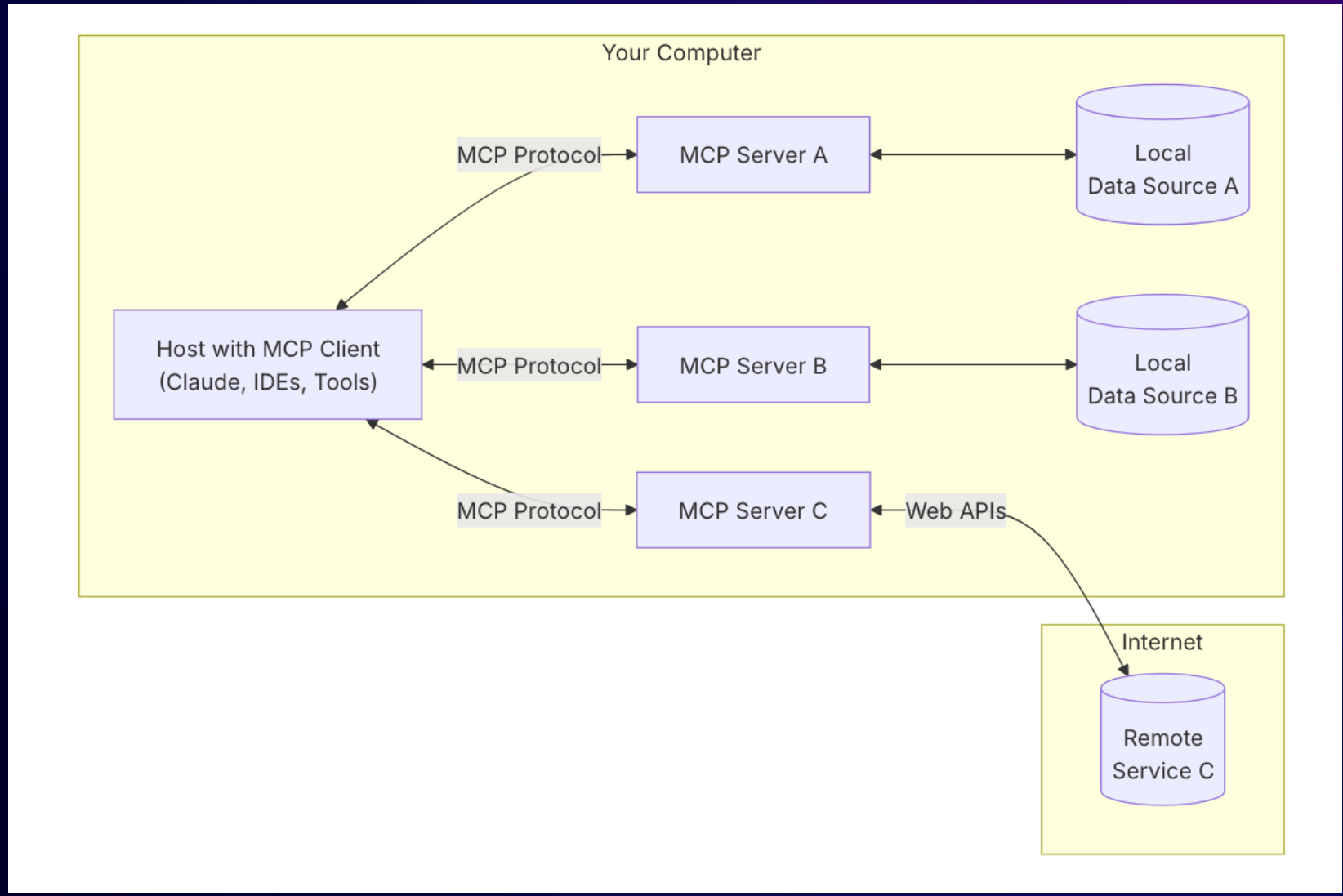
Here's the problem it solves:



249 695 5.6K 656K

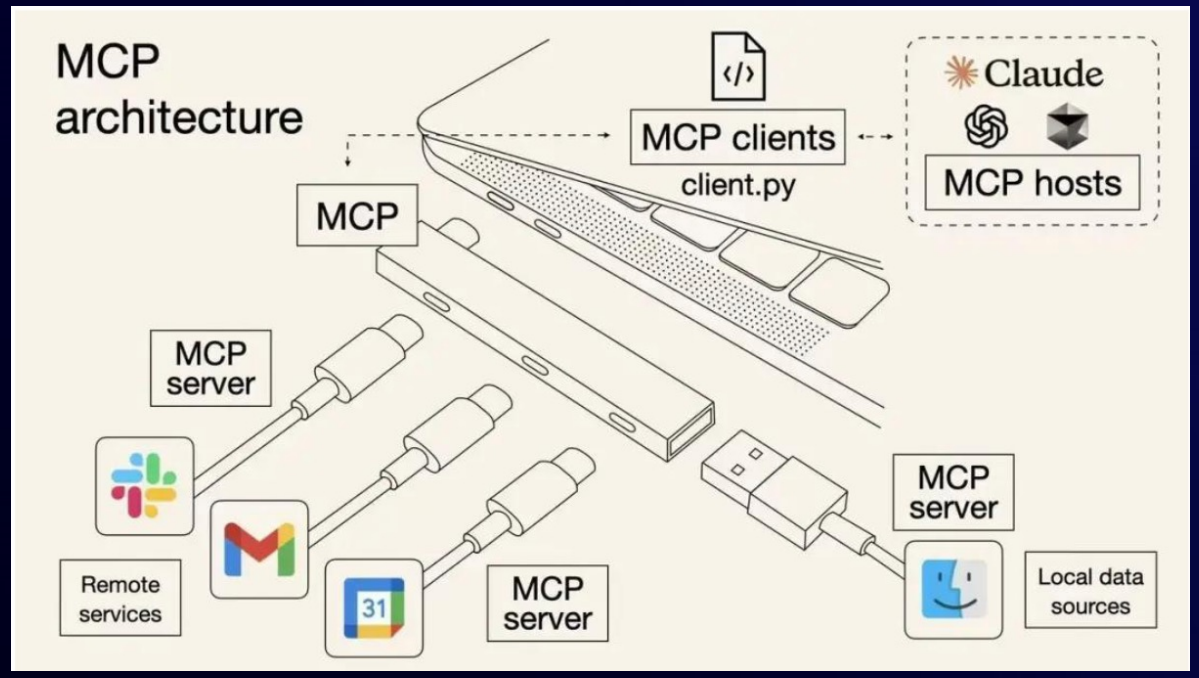
Enter MCP

1. Open standard by Anthropic solving AI integration complexity
2. Client-server protocol connecting AI to data and tools.
3. Enables seamless context switching across tools and datasets.
4. Includes SDKs and pre-built integrations (Drive, Slack, GitHub, Postgres).
5. Standardizes AI workflows for simplicity, consistency, and scale.



MCP Docs: https://docs.anthropic.com/en/docs/agents-and-tools/mcp

MCP in simple term



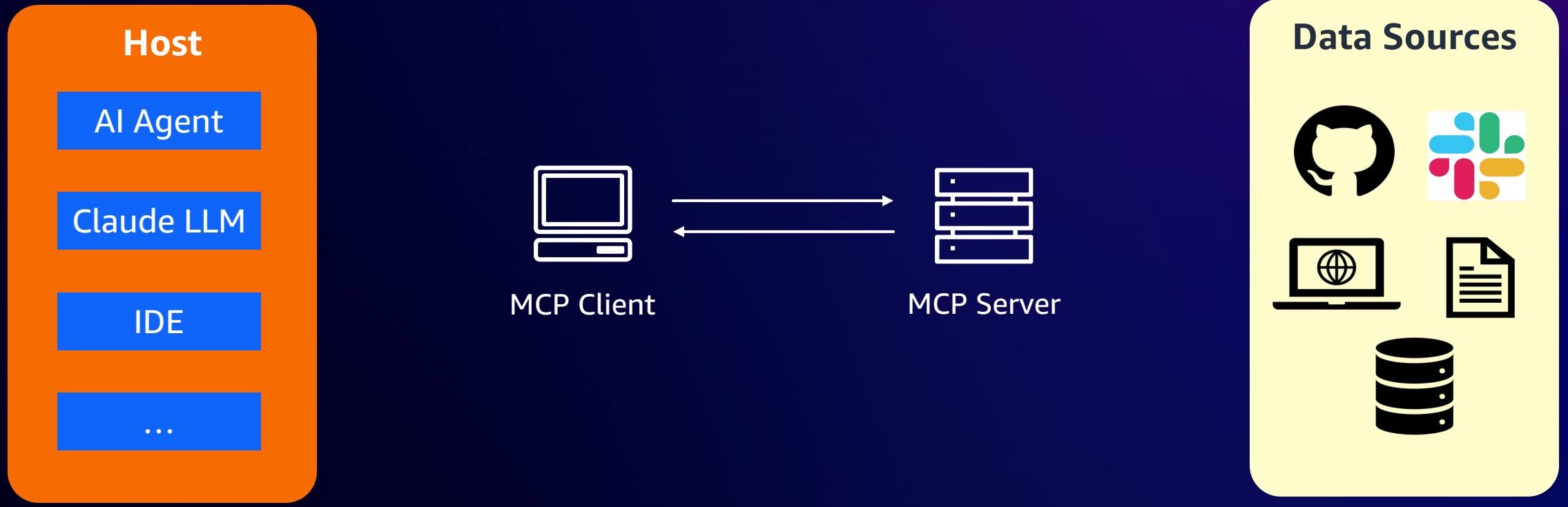
* for business users analogy only, as this is not accurated from technical perspective

- MCP helps reduced complexity
- MCP decouples AI development from specific backend system
  - MCP client focuses on LLM, agent, application layer focused
  - MCP server focuses on data sources connectivity
- Hence, if you have new data sources, just connect to MCP server to make discoverable by MCP client (hence, AI agent)

Agents with MCP

Connect the host (app) to MCP client, and request execution

MCP main components

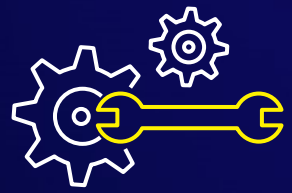


1. **Host:** A program, AI tool, or AI agents requiring access to data through MCP
2. **MCP Client:** Protocol clients maintaining one-to-one connections with servers
3. **MCP Server:** Lightweight programs exposing capabilities through standard MCP
4. **Data sources:** both local (databases, files) and remote services (APIs) that MCP server can access

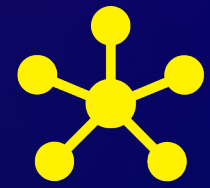
Why is MCP important

Standardization

Integration with External Data Sources



Enhanced Functionality



Flexibility and Scalability

MCP Concept

Resources

- Allows Servers to expose data
- File contents, database records
- Application-controlled (client decides how and when to use)

Prompts

- Reusable prompt template and workflows
- User-controlled i.e., from servers to clients and then to user

Tools

- Enable LLMs to perform actions through server
- Model-controlled, expose from server to client

In summary …

Aspect	Before MCP	After MCP
Integration effort	Custom code for each system	Standardized, reusable protocol
Data/tool access	Limited, siloed, often outdated	Real-time, multi-source, live
AI context/memory	Lost between tools and sessions	Persistent, shared across tools/models
User experience	Fragmented, repetitive, impersonal	Seamless, personalized, consistent
Vendor lock-in	High (integration tied to model/tool)	Low (model-agnostic, interoperable)
Developer productivity	Slow, error-prone, high maintenance	Fast, reliable, low maintenance
Ecosystem growth	Fragmented, duplicated effort	Network effects, rapid expansion

Strands Agents

The broadest choice for building and deploying agents

SPECIALIZED

Amazon Q
Pre-built products that use agents for enhanced productivity

FULLY-MANAGED

Amazon Bedrock Agents
with built-in FM-powered orchestration

DIY

Strands Agents
Simple, flexible, and lightweight open source SDK for building agents

OUT-OF-THE-BOX AGENTS

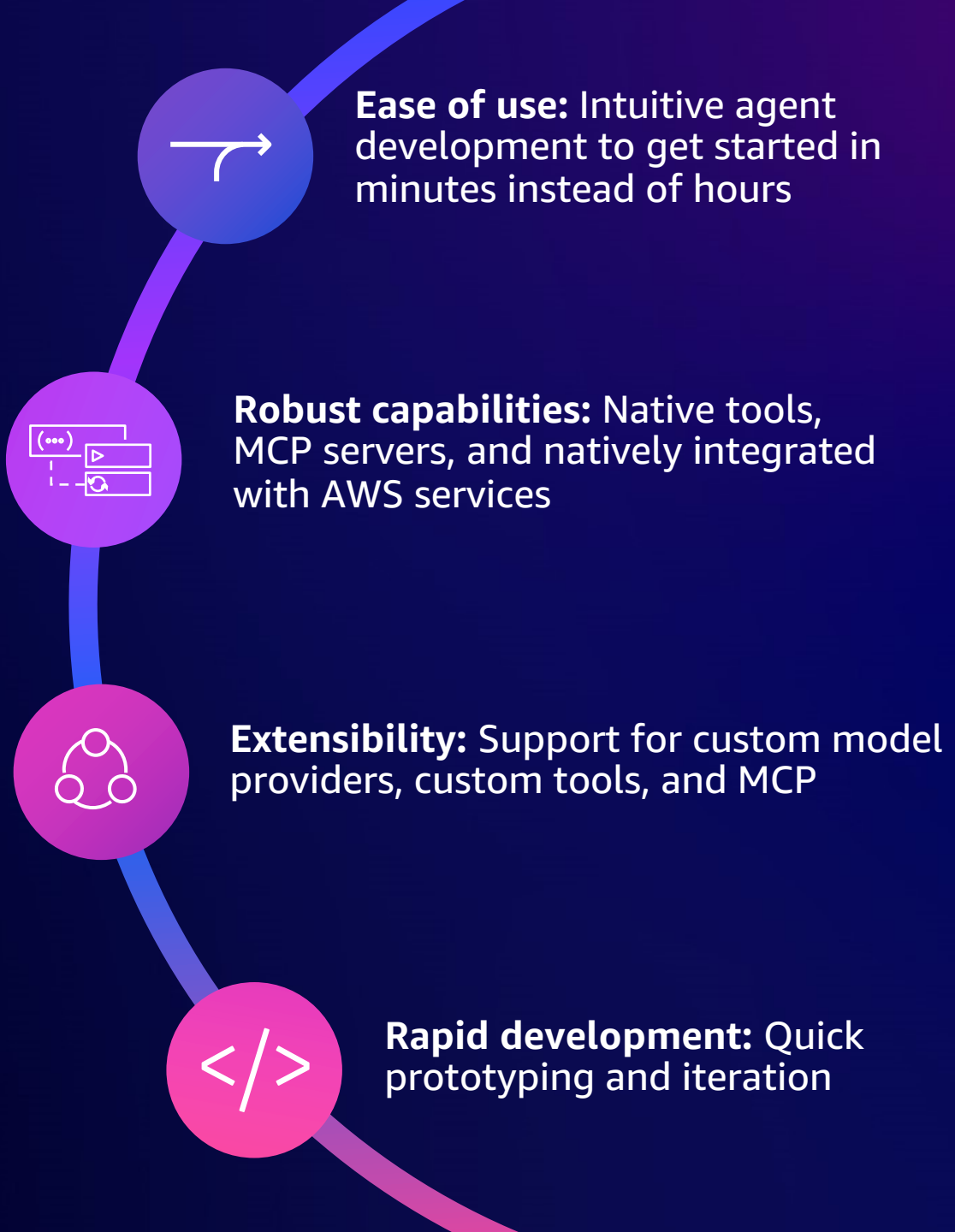
TOOLS FOR BUILDING AGENTS

Framework	AWS integration	Autonomous multi-agent support	Autonomous workflow complexity	Multimodal capabilities	Foundation model selection	LLM API integration	Production deployment
AutoGen	Weak	Strong	Strong	Adequate	Adequate	Strong	Do it yourself (DIY)
CrewAI	Weak	Strong	Adequate	Weak	Adequate	Adequate	DIY
LangChain/ LangGraph	Adequate	Strong	Strongest	Strongest	Strongest	Strongest	Platform or DIY
Strands Agents	Strongest	Strong	Strongest	Strong	Strong	Strongest	DIY

https://docs.aws.amazon.com/prescriptive-guidance/latest/agentic-ai-frameworks/comparing-agentic-ai-frameworks.html

Strands Agents

Strands Agents is an open source SDK for building agents using just a few lines of code



Ease of use: Intuitive agent development to get started in minutes instead of hours

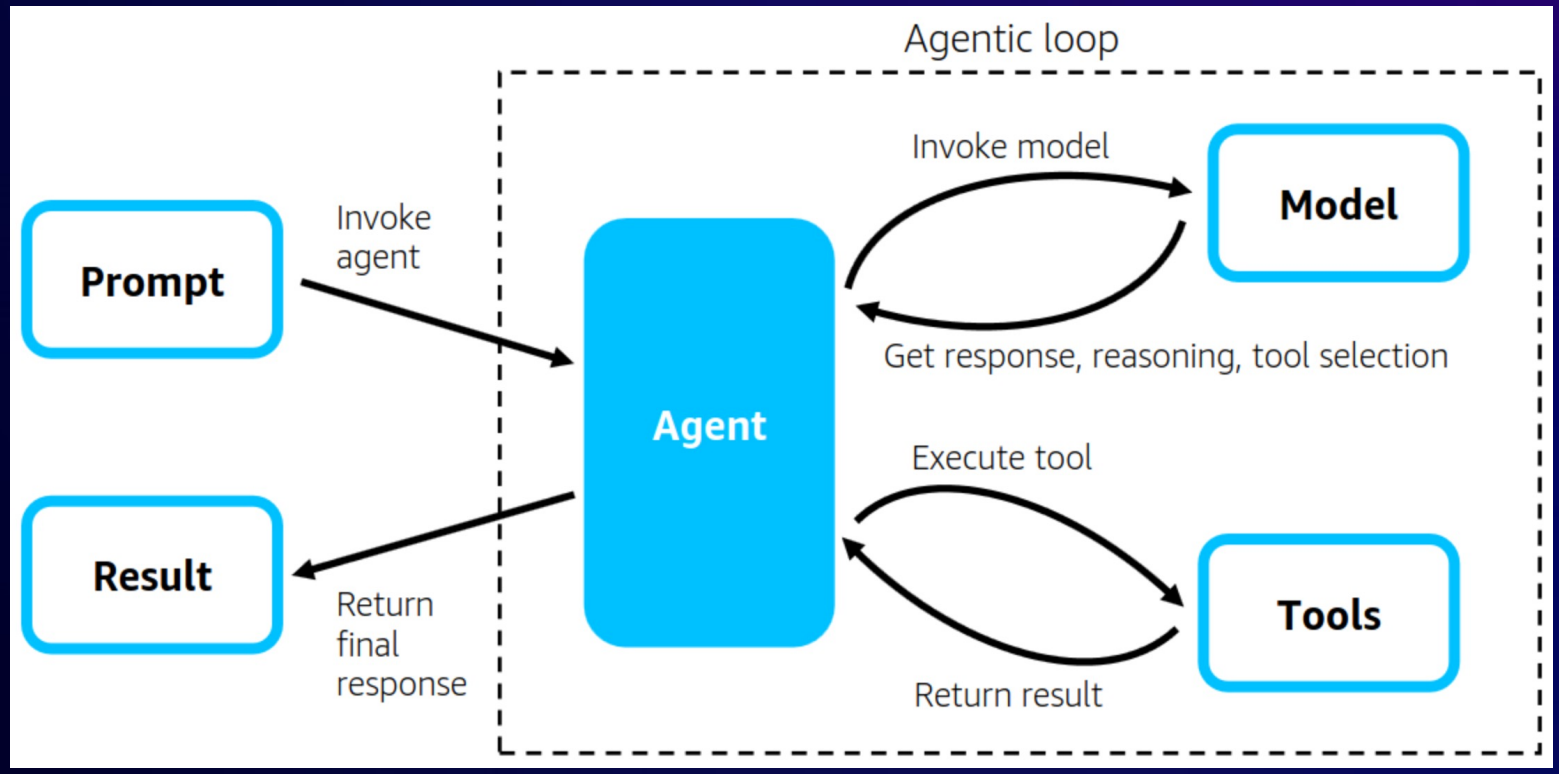
Robust capabilities: Native tools, MCP servers, and natively integrated with AWS services

Extensibility: Support for custom model providers, custom tools, and MCP

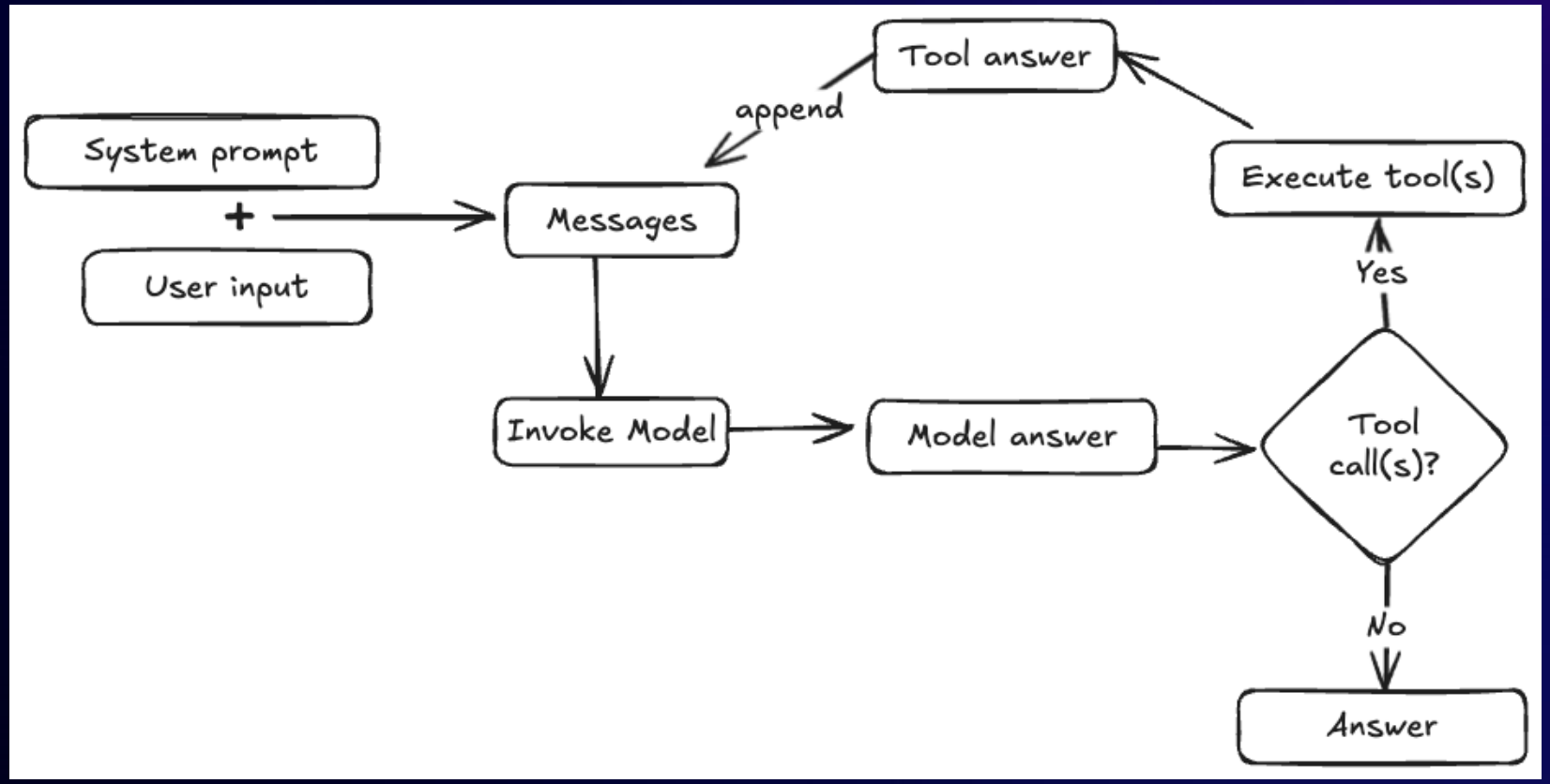
Rapid development: Quick prototyping and iteration

Strands see Agents as “models using tools in cycles”

At the core of Strands Agents is a simple agentic loop which enables models to autonomously plan next steps and execute tools.



Understand the control flow



Built for builders

Who value flexibility, speed, and simplicity

Model & deployment choice

- ✓ Model choice
- ✓ Custom model providers
- ✓ Deploy anywhere

Highly flexible

- ✓ Safeguard with guardrails
- ✓ Native observability
- ✓ Monitoring
- ✓ Evaluation
- ✓ Session Management

Broad selection of tools

- ✓ MCP integration
- ✓ Custom tools
- ✓ Coordinate multiple agents
- ✓ Multi-modal
- ✓ Fetch web data
- ✓ Read and write files
- ✓ Interpret code

Integrations

- ✓ Use AWS services
- ✓ LiteLLM
- ✓ Mem0
- ✓ Ragas
- ✓ Tavily
- ✓ Langfuse
- ✓ MCP
- ✓ A2A

Simple hello world Strands Agent

```
from strands import Agent
from strands_tools import calculator

agent = Agent(tools=[calculator])
response = agent("What is 80 / 4?")
```

Model choice – Bedrock

```
from strands import Agent
from strands.models import BedrockModel

bedrock_model1 = BedrockModel(
    model_id= "us.amazon.nova-premier-v1:0",
    params={"max_tokens": 1600, "temperature": 0.7}
)
bedrock_model2 = BedrockModel(
    model="us.anthropic.claude-3-7-sonnet-20250219-v1:0"
)

agent = Agent(model=bedrock_model1)
response = agent("Tell me about Bedrock")
```

Model choice – LiteLLM

```
from strands import Agent
from strands.models.litellm import LiteLLMModel

litellm_model = LiteLLMModel(
    model_id= "azure/gpt-4o-mini",
    params={"max_tokens": 1600,
            "temperature": 0.7}
)
agent = Agent(model=litellm_model)
response = agent("Tell me about LiteLLM")
```

Choice of providers

- Amazon Bedrock
- Anthropic API
- LiteLLM
- Llama API
- MistralAI
- Ollama
- OpenAI
- Writer
- Cohere
- **Custom Providers**

LiteLLM

```python
from strands import Agent
from strands.models.litellm import LiteLLMModel
from strands_tools import calculator

model = LiteLLMModel(
    # **model_config
    client_args={
        "api_key":"sk-a                                                    qxhk",
        "base_url":"https://api.siliconflow.cn/"
    },
    model_id="openai/Qwen/Qwen3-235B-A22B",
    params={
        "max_tokens": 1000,
        "temperature": 0.7,
    }
)

agent = Agent(model=model, tools=[calculator])
response = agent("What is 2+2")
print(response)
```

OpenAI

```python
from strands.models.openai import OpenAIModel
from strands_tools import calculator
import os

model = OpenAIModel(
    # **model_config
    client_args={
        "api_key":os.environ["OPENAI_API_KEY"],
        "base_url":"https://api.siliconflow.cn/"
    },
    model_id="Pro/deepseek-ai/DeepSeek-V3",
    max_tokens =  1000,
    temperature = 0.7,
)

agent = Agent(model=model, tools=[calculator])
response = agent("what is 2+2")
```

Tools – repo of pre-built tools to get started quickly

```
from strands import Agent
from strands_tools import http_request

agent = Agent(tools=[http_request])
print(agent("Where is the “ +
    “International Space Station?"))
```

Tool categories

RAG & Memory
File Operations
Shell & System
Code Interpretation
Web & Network
Multi-modal
AWS Services
Utilities
Agents & Workflows

Prebuilt Tools Available

- 📁 File Manipulation - Read, write, and edit files with smart modification
- 🖥️ Shell integration - execute and interact with shell commands
- 🧠 Memory tool - stores user and agent memories during operation
- 🌐 HTTP client - makes API requests, supports authentication
- 🐍 Python execution - Run Python code snippets with state persistence
- 🧮 Math tools - Use symbolic math features to perform advanced calculations
- ☁️ AWS integration — Common AWS service access tools such as S3, EC2, DDB, etc.
- 🖼️ Image processing - Generating and processing images for AI applications
- 🔄 Environmental management - safe handling of environmental variables
- 📝 Logging - Create and manage structured journals
- 🧠 Advanced Reasoning - A tool for complex thinking and reasoning skills
- 🐝 Swarm swarm intelligence - Coordinates multiple agents to solve problems in parallel
- 🌐 Graph/Workflow Tool — Tool for creating Agent graphs/workflows

Tools – create custom tools

```python
from strands import Agent, tool
@tool
def weather_forecast(city: str, days: int = 3) -> str:
    """Get weather forecast for a city.
    Args:
        city: The name of the city
        days: Number of days for the forecast
    """
    return f"Weather forecast for {city} for the next {days} days..."

agent = Agent(tools=[weather_forecast])
print(agent("What's the weather in Seattle tmw?"))
```

Remember MCP (Model Context Protocol)

Is a standard protocol to connect Agents to Tools

Creating agents with MCP tools

Natively integrates with MCP to extend agent capabilities through external tools and services. Strands Agents provides several ways to connect to MCP servers:

Standard I/O (stdio)

For command-line tools and local processes that implement the MCP protocol

Streamable HTTP

For HTTP-based MCP servers that use Streamable-HTTP Events transport

Server-Sent Events (SSE)

For HTTP-based MCP servers that use Server-Sent Events transport (deprecated)

Custom Transport with MCPClient

You can also implement a custom transport mechanism using MCPClient class directly

How to use MCP with Strands Agents? - Stdio

```python
from mcp import stdio_client, StdioServerParameters
from strands import Agent
from strands.tools.mcp import MCPClient

stdio_mcp_client = MCPClient(lambda: stdio_client(
    StdioServerParameters(command="uvx",
        args=["awslabs.aws-documentation-mcp-server@latest"]))
)
with stdio_mcp_client:
    tools = stdio_mcp_client.list_tools_sync()
    agent = Agent(tools=tools)
    agent("What does Bedrock InvokeInlineAgent do?")
```

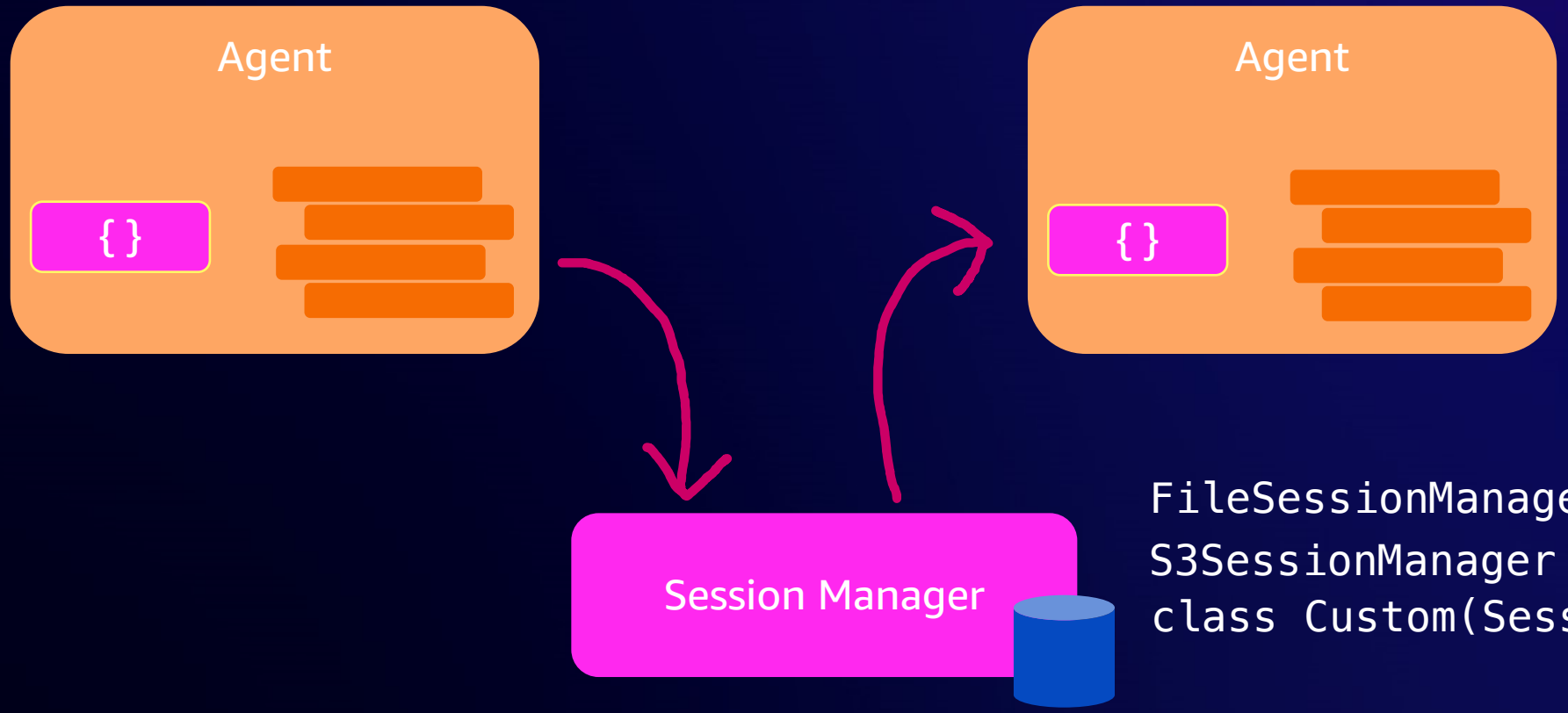
How to use MCP with Strands Agents? - HTTP

```python
from mcp.client.streamable_http import streamablehttp_client
from strands import Agent
from strands.tools.mcp.mcp_client import MCPClient

streamable_http_mcp_client =MCPClient(
      lambda: streamablehttp_client( "http://localhost:8000/mcp")
)

with streamable_http_mcp_client:
    tools = streamable_http_mcp_client.list_tools_sync()
    agent = Agent(tools=tools)
    agent("What does Bedrock InvokeInlineAgent do?")
```

State & Sessions

Session Management

- **Use case**: Stop and resume an Agent for a given session Id




```
FileSessionManager
S3SessionManager
class Custom(SessionRepository)
```

State & Sessions

Strands Agents state is maintained in several forms:

- **Conversation History:** The sequence of messages between the user and the agent.
- **Agent State**: Stateful information outside of conversation context, maintained across multiple requests.
- **Request State**: Contextual information maintained within a single request.

Conversation History

```
from strands import Agent
agent = Agent()
agent("Hello!")
print(agent.messages)
```

```
from strands import Agent
agent = Agent(messages=[
{"role": "user", "content": [{"text": "Hello, my name is Strands!"}]},
{"role": "assistant", "content": [{"text": "Hi there! How can I help you today?"}]}
])

agent("What's my name?")
```

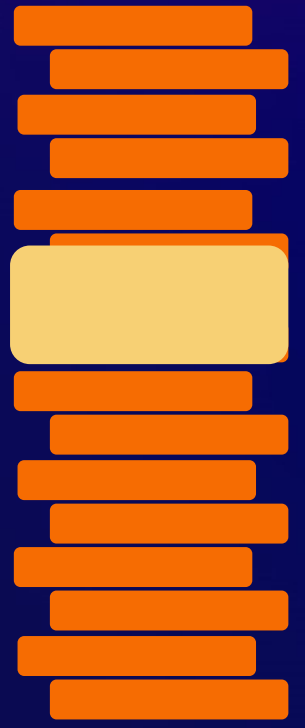
Conversation Management

Use case: Manage context size

No management

Sliding Window (K=40)

Summarization



Conversation Managers

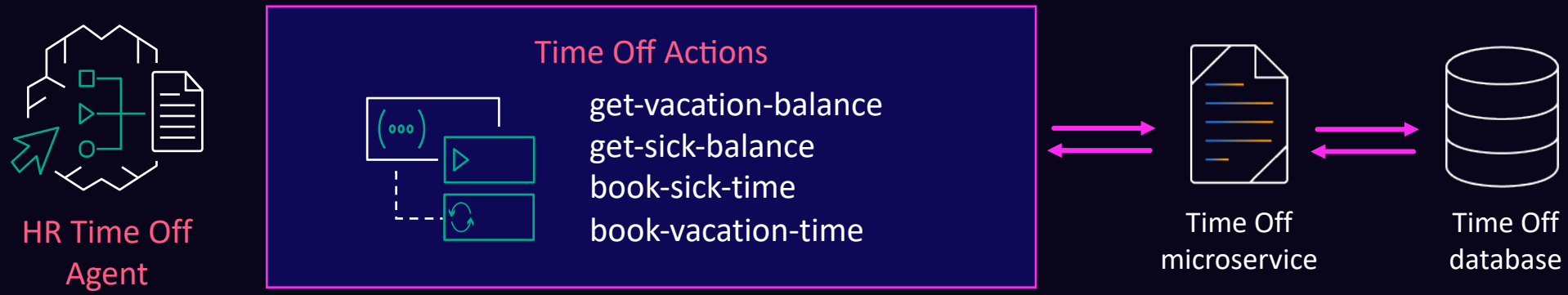
```
from strands import Agent
from strands.agent.conversation_manager import
SlidingWindowConversationManager

# Create a conversation manager with custom window size
conversation_manager = SlidingWindowConversationManager(
      window_size=20, # Maximum number of messages to keep
      should_truncate_results=True, # Enable truncating the result
)
agent = Agent(
      conversation_manager=conversation_manager
)
```

Conversation Managers

```python
from strands import Agent

from strands.agent.conversation_manager import SummarizingConversationManager

# Create the summarizing conversation manager with default settings
conversation_manager = SummarizingConversationManager(
          summary_ratio=0.3, # Summarize 30% of messages when context reduction is needed
          preserve_recent_messages=10, # Always keep 10 most recent messages
)

agent = Agent(
          conversation_manager=conversation_manager
)
```

Multi-agent collaboration patterns

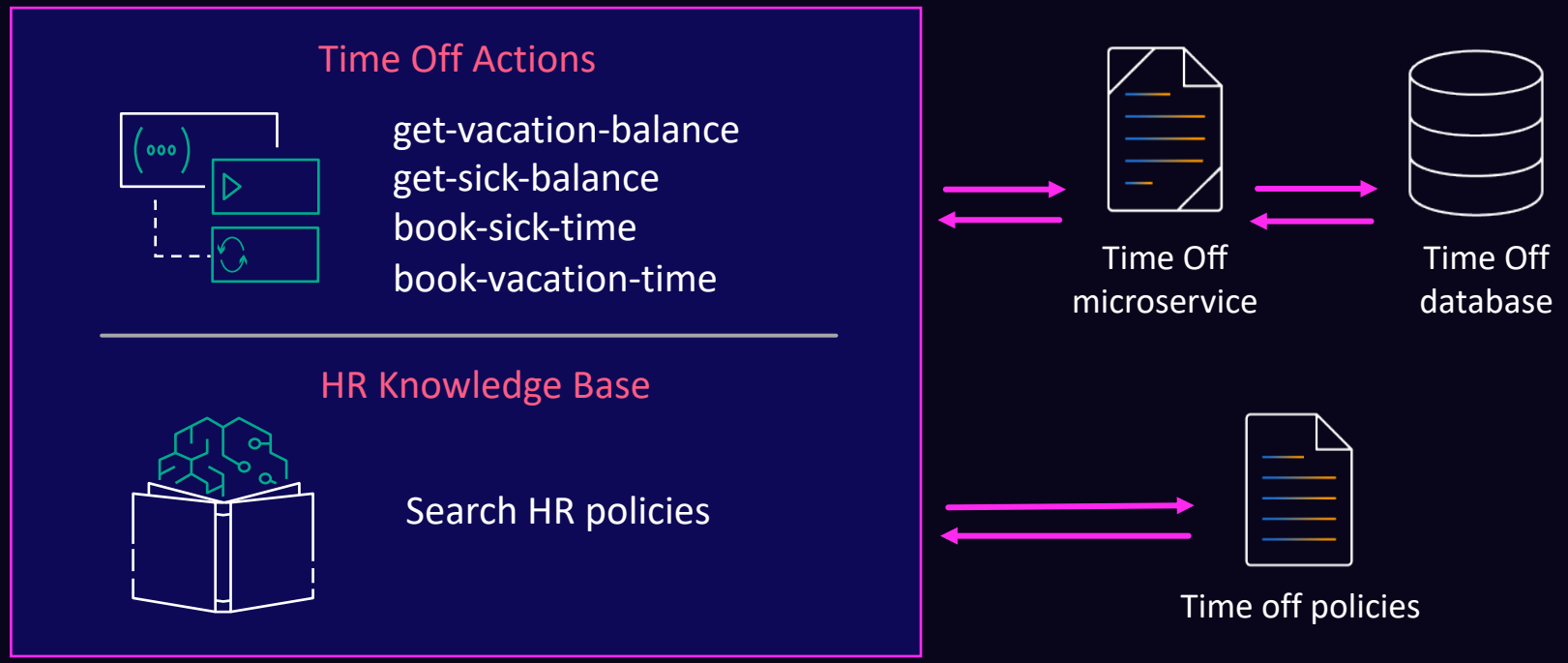
Agents start small and focused . . .



. . . and can be easily expanded

V2

HR Time Off Agent



. . . but if you take a SINGLE agent too far

V4

HR Time Off Agent

Time Off Actions
- get-vacation-balance
- get-sick-balance
- book-sick-time
- book-vacation-time

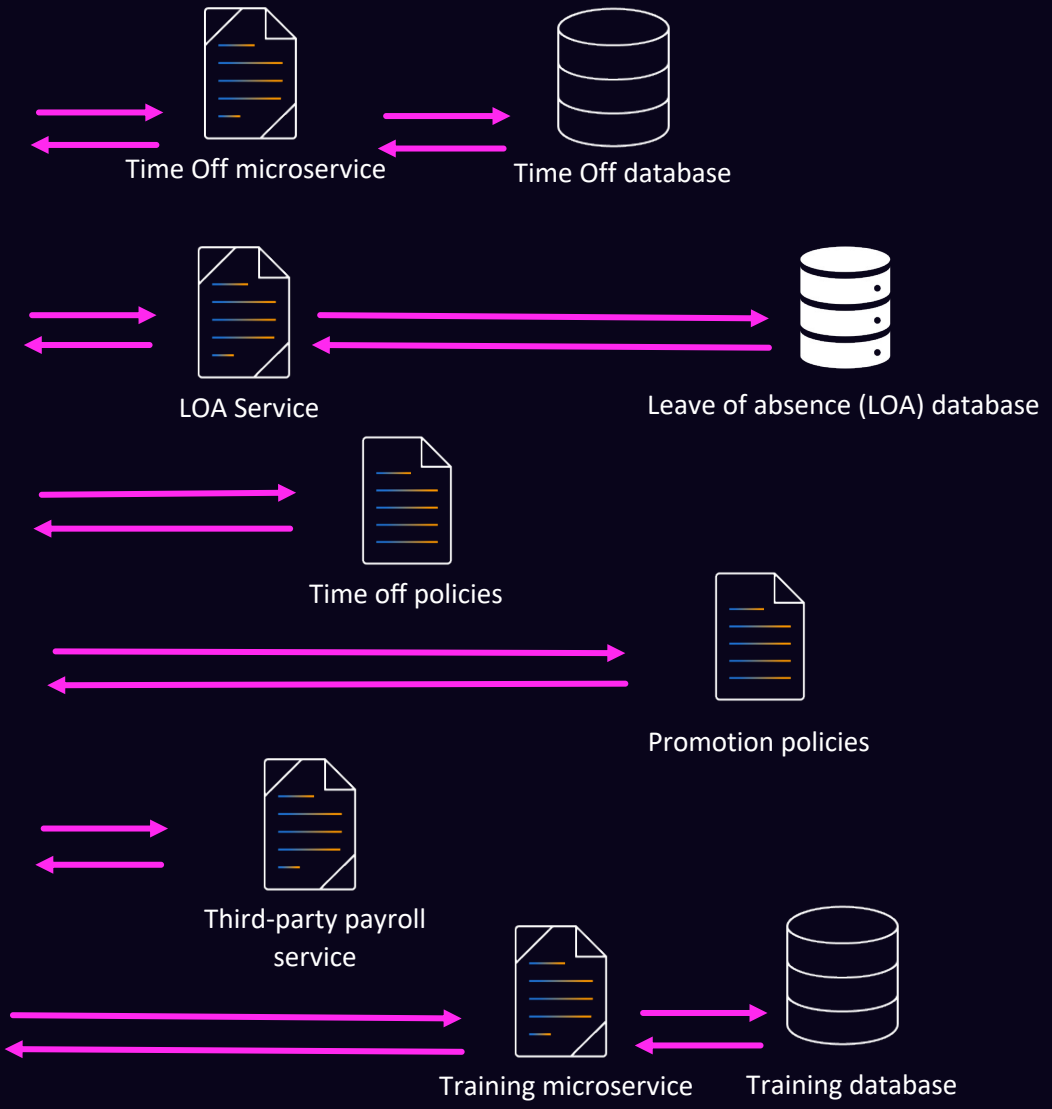
Leave of Absence (LOA) Actions
- start-new-LOA
- extend-LOA
- close-LOA
- get-LOA-details

HR Knowledge Base
- Search HR policies

Promotion Knowledge Base
- Promotion policies

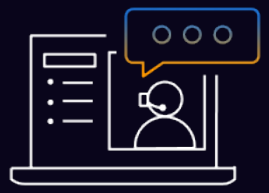
Payroll Actions
- get-latest-pay-stub
- submit-payroll-ticket
- manage-deductions

Training Actions
- find-training
- list-certifications
- register-for-course
- submit-review



. . . it leads to challenges

Coding gets complicated



- Complex prompts to limit hallucinations
- Fragile, hard to maintain

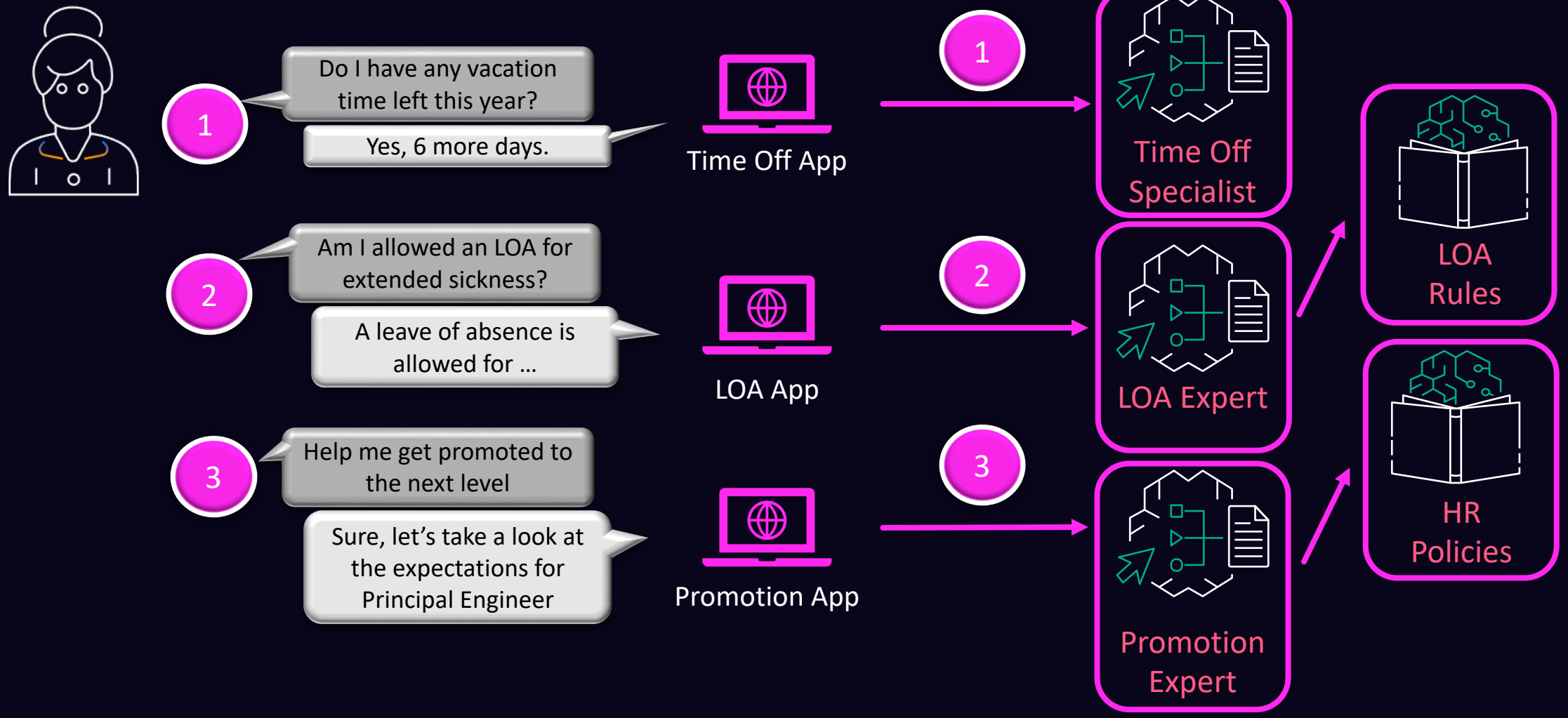
Agent gets confused

- Calling wrong tools
- Passing wrong arguments
- Inconsistent responses

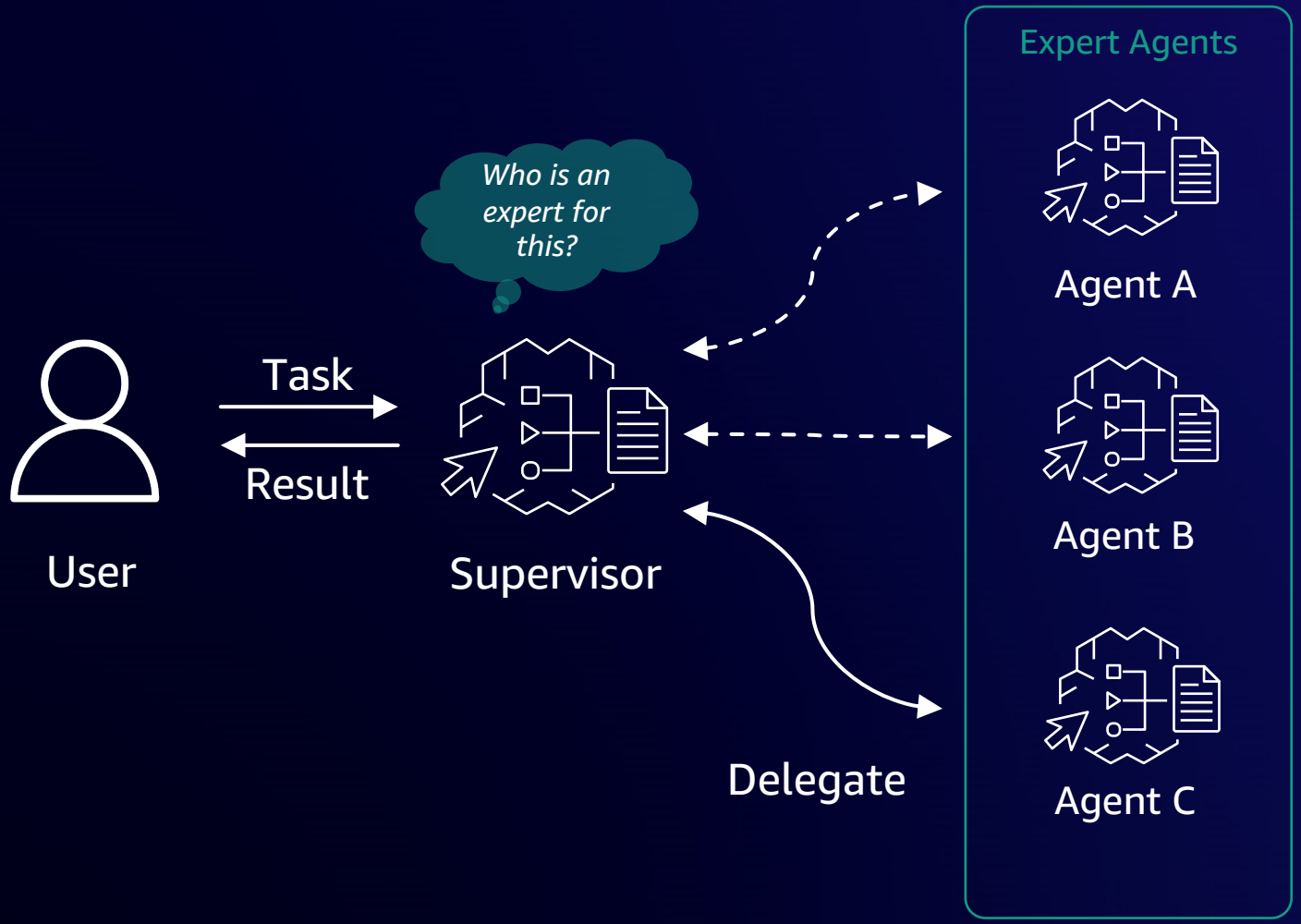
Agent gets slower and more expensive

- Frontier models needed
- Prompt sizes grow
- Agents retry steps

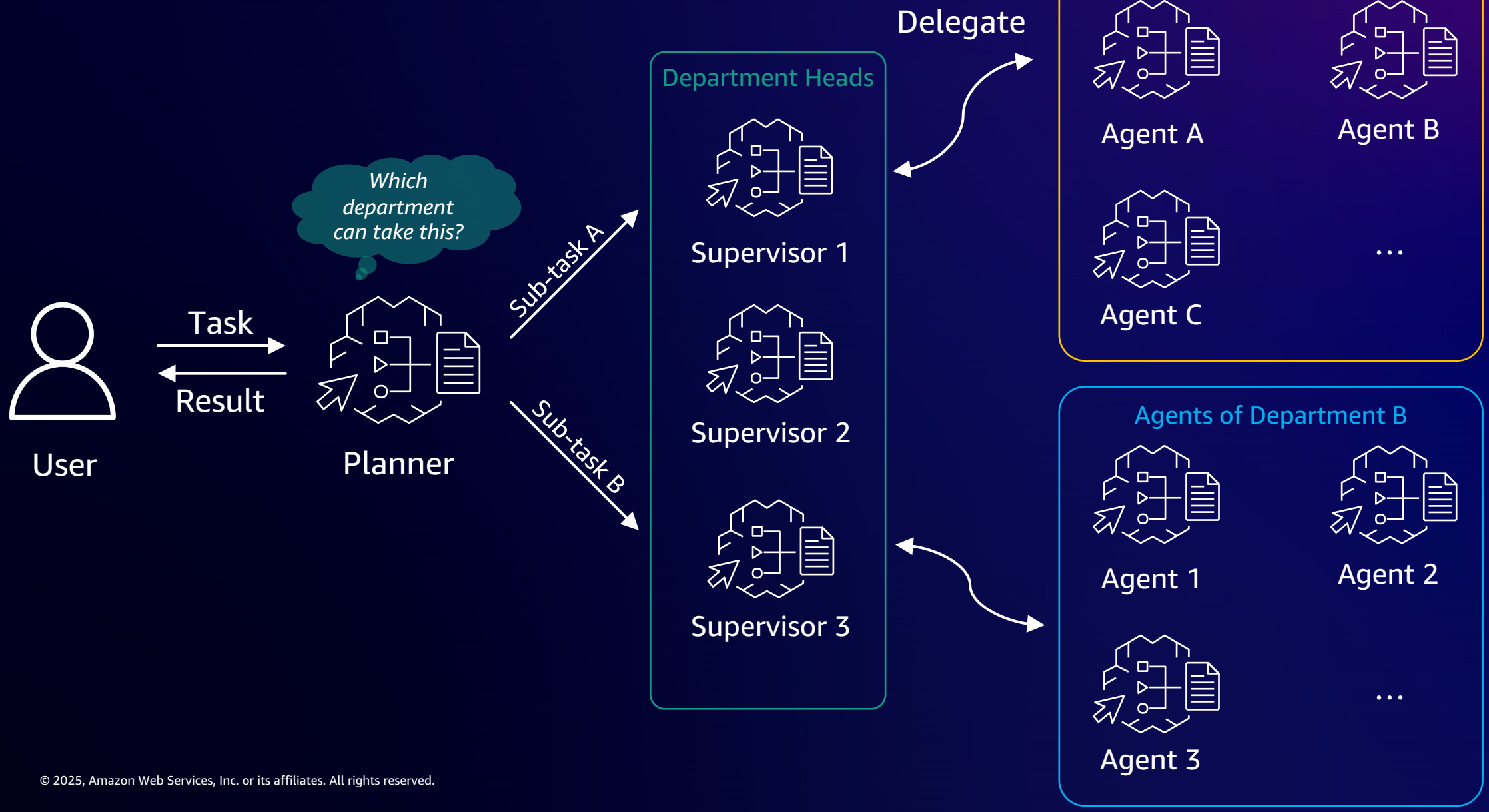
Using multiple agents helps . . .



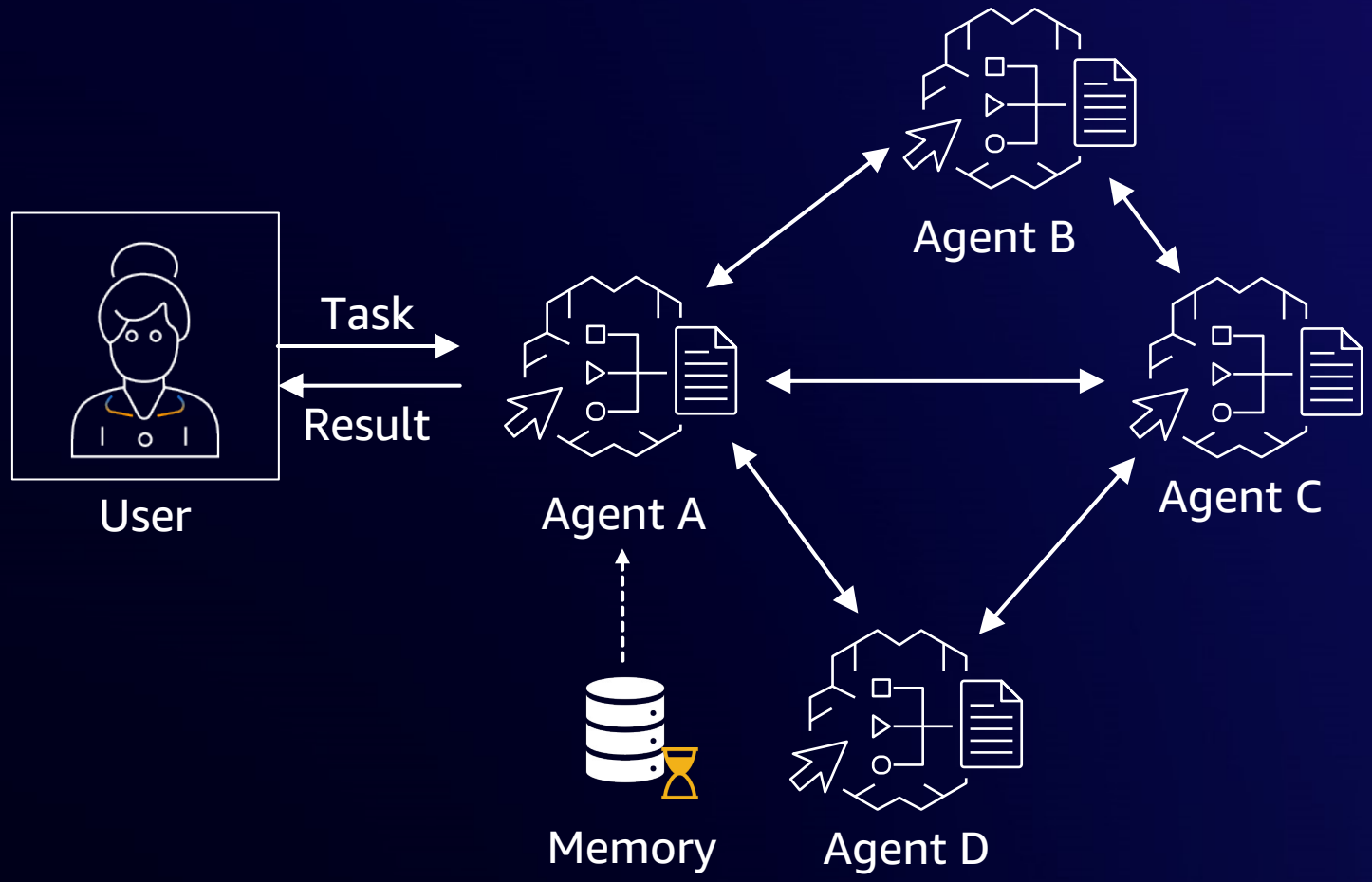
Agents as Tools



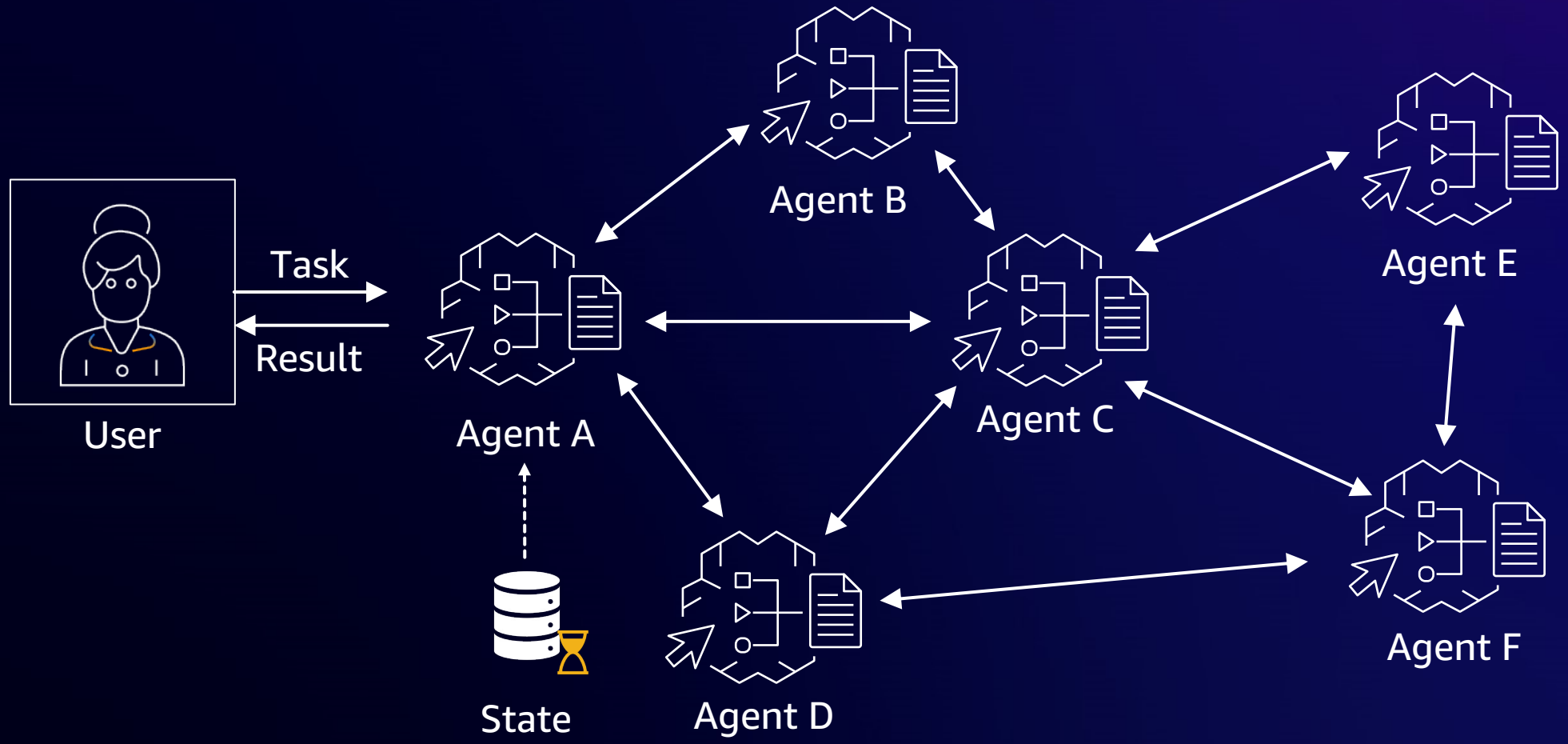
Hierarchical Agents as Tools



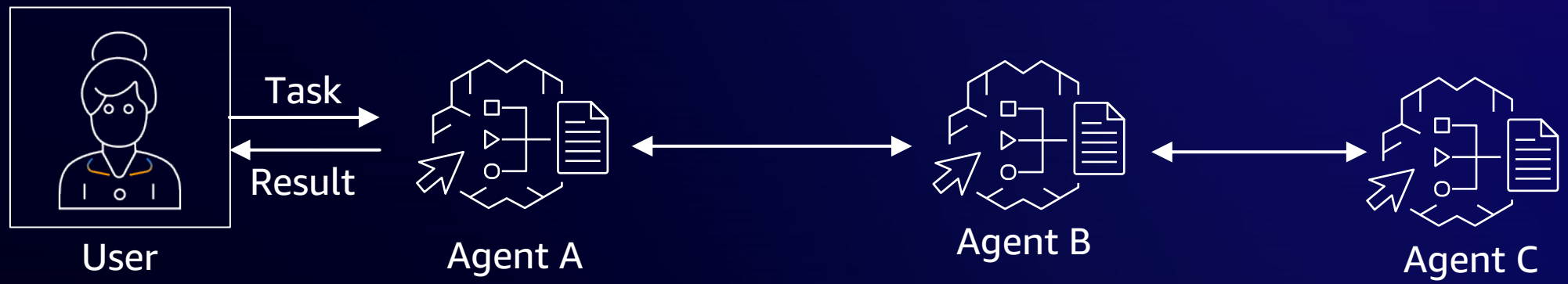
Swarm Agents



Graph Agents



Workflow Agents



Strands Agents multi-agent collaboration patterns

Agent as Tools

Hierarchical structure where:

1. "orchestrator" agent handles user interaction and determines which specialized agent to call

2.Specialized agent performs domain-specific tasks when called by the orchestrator

Swarm

Multi-agent systems where agents interact within the environment. Architecture consists of 4 key components:

1. Communication Patterns
2. Shared Memory Systems
3. Coordination Mechanisms
4. Task Distribution

Graph

Network of agents where each agent represents a node with specific capabilities, and the connections between agents is defined explicitly.

1. Nodes (agents)
2. Edges (connections)
3. Topology Pattern

Workflow

Structured coordination of tasks across agents. Provides explicit control over execution order, dependencies, and information flow.

1. Task Definition and Distribution
2. Dependency Management
3. Information Flow

Observability and Evaluation

Strands Agents Observability and Evaluation

Strands Agents provide metrics and OTEL traces out-of-the-box

Metrics

- **Token usage**: Input tokens, output tokens, and total tokens consumed
- **Performance metrics**: Latency and execution time measurements
- **Tool usage**: Call counts, success rates, and execution time for each tool
- **Event loop cycles**: Number of reasoning cycles and their durations

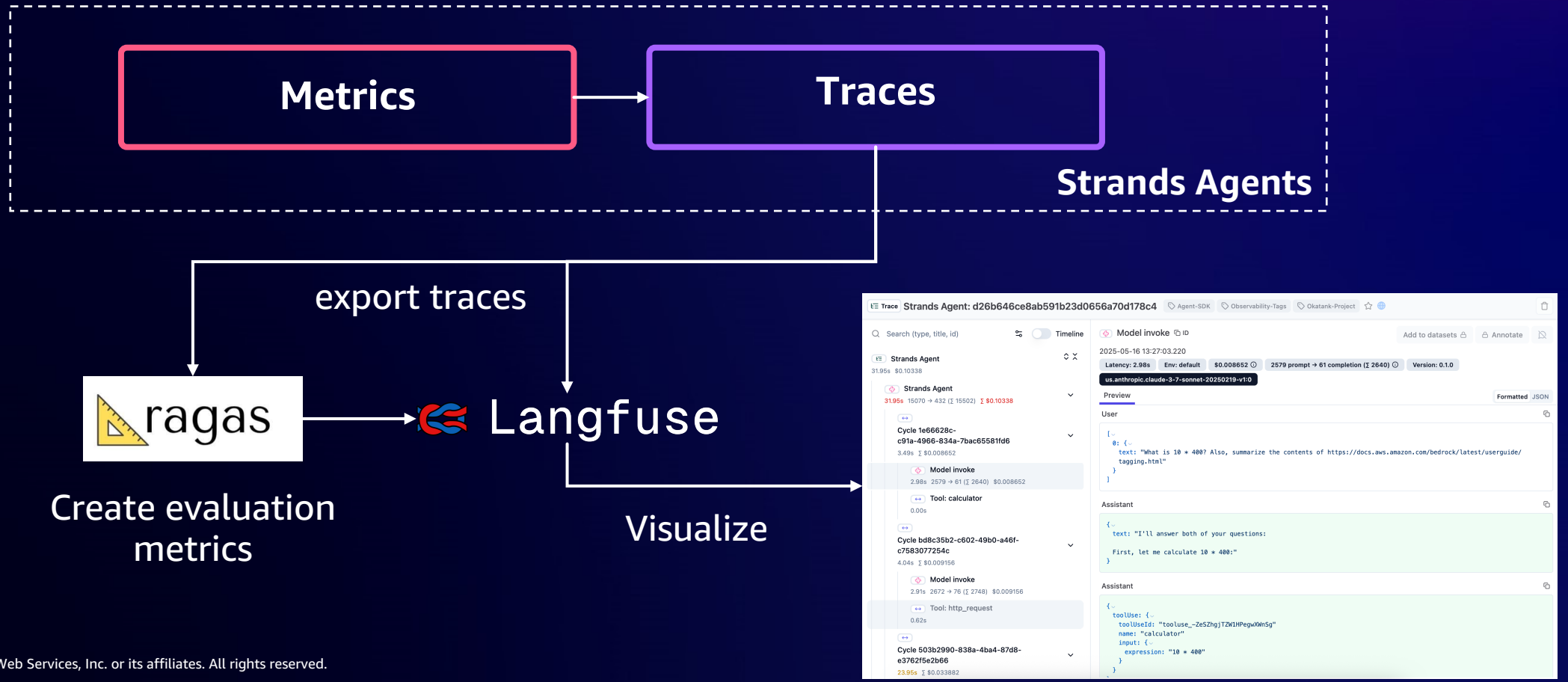
Traces

- **Track the entire agent lifecycle**: From initial prompt to final response
- **Monitor individual LLM calls**: Examine prompts, completions, and token usage
- **Analyze tool execution**: Understand which tools were called, with what parameters, and their results
- **Debug complex workflows**: Follow the exact path of execution through multiple cycles

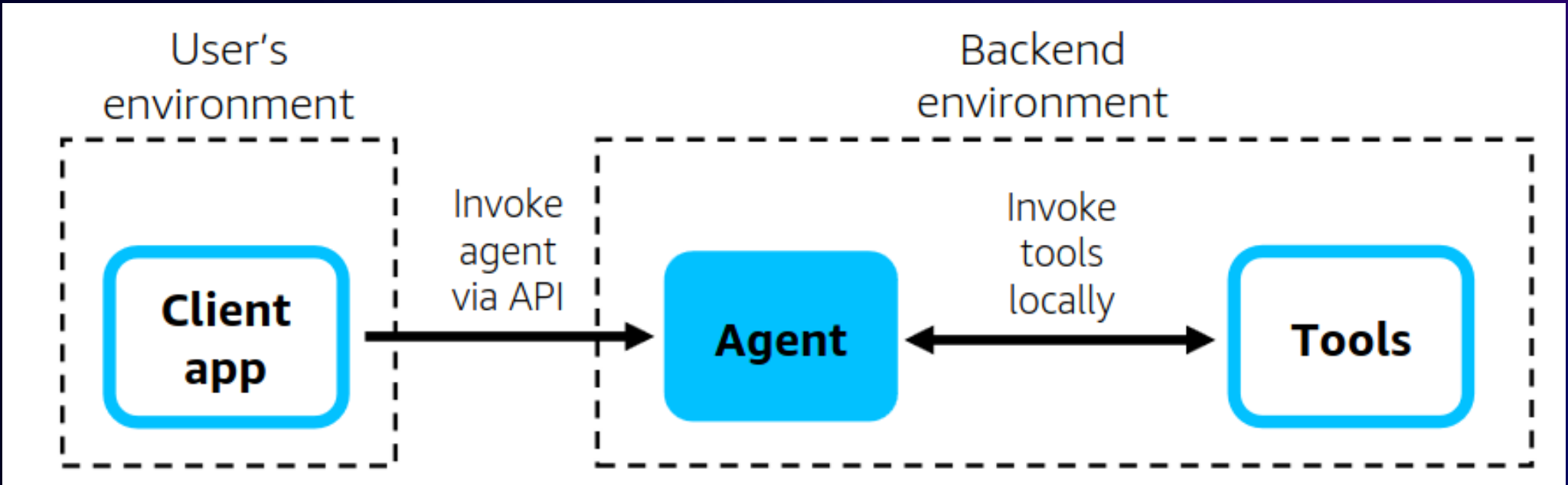
Metrics are also exposed into the traces

Strands Agents Observability and Evaluation

Strands integrates with different open-source tools to evaluate your agent and visualize metrics and traces on dashboards



Run agent behind an API



Strands Agents comes with examples and documented steps for deploying agents on: AWS Lambda, Amazon EC2, AWS Fargate, Amazon EKS, Amazon Bedrock AgentCore Runtime

Difference between Bedrock Agents and Strands Agents

Amazon Bedrock Agents

- For developers who want a fully-managed experience for building and deploying agents.
- Fast path to production, using API calls to specify the model, instructions, tools, configuration.
- Avoids overhead of managing hosting and scaling, or stitching together core agent capabilities.
- Comes with multi-agent collaboration, observability, code interpretation, memory management, guardrails and knowledge bases.

Strands Agents

- For developers who want deeper customization of agent behavior, including orchestration, and business logic.
- Works with any model provider, including proprietary LLM gateways.
- Agents can be deployed anywhere you would host a Python application.
- Open source community project maintained by AWS.

Get started with Strands Agents

Start your agentic AI journey today

Docs

Review the user guide, quick start, API documentation for best practices

SDK

Get started building agents and contribute to the open-source project on GitHub

Tools

Access pre-built tools to get started experimenting with agents

Samples

Discover sample agents in our samples repository

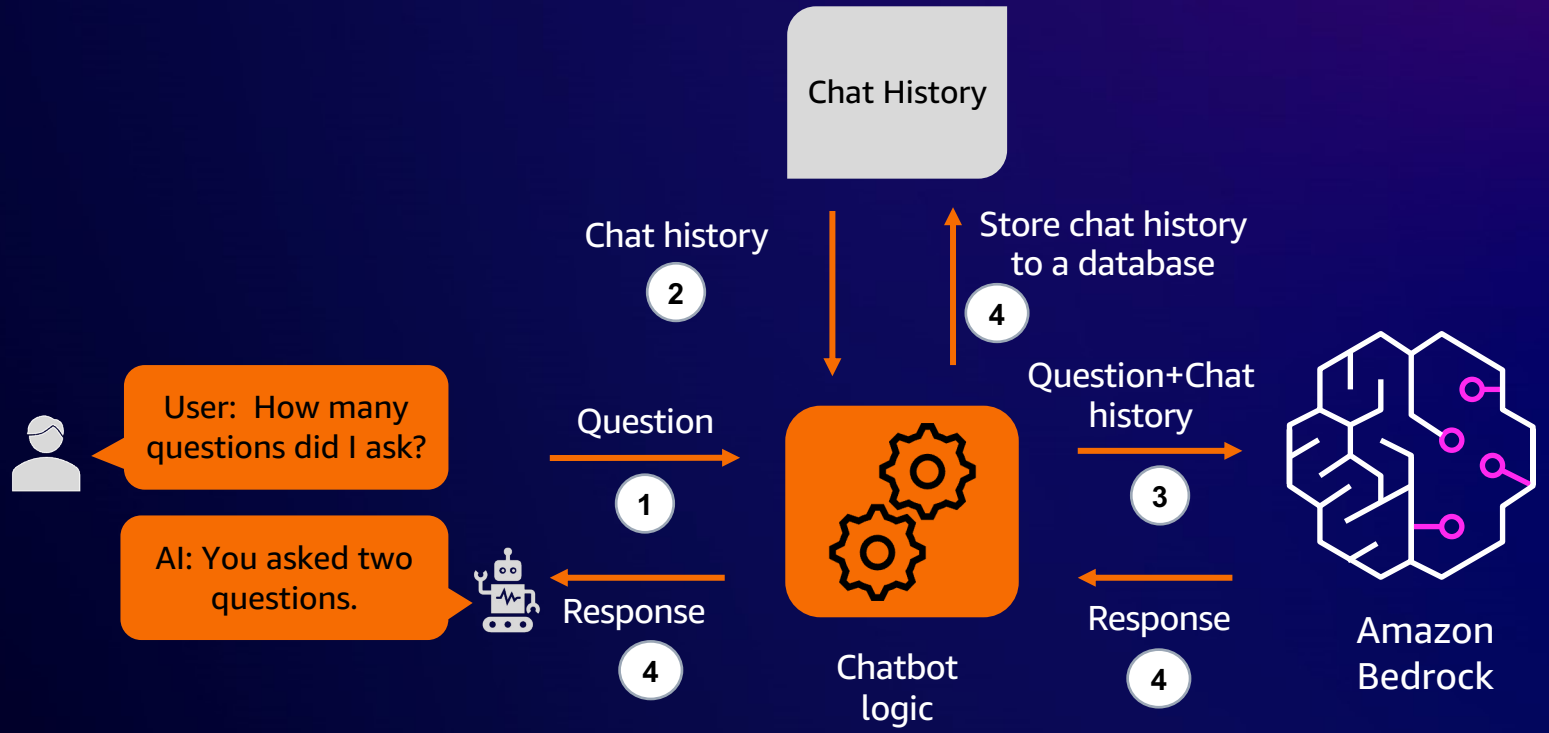
Notebook Exercises

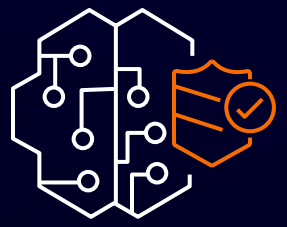
GENAI BOOTCAMP

Secure Chatbots with Bedrock Guardrails

Chatbot - Overview

1. User asks a question
2. Chat history is retrieved
3. Answer is generated using question and chat history
4. Answer is returned to the user and saved in chat history





Amazon Bedrock Guardrails

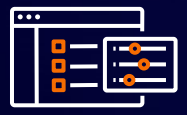
Apply safeguards customized to your generative AI application requirements and responsible AI policies

Why do you need additional controls?

THE COST OF NO GUARDRAILS: HIDDEN RISKS OF UNFILTERED AI INTERACTIONS

Why guardrails?

Customizations based on use cases and organizational policy



Safety and privacy controls for responsible AI

Consistent safeguards across FMs and applications

The cost without guardrails

Good FM training isn't enough

Dataset poisoning is persistent

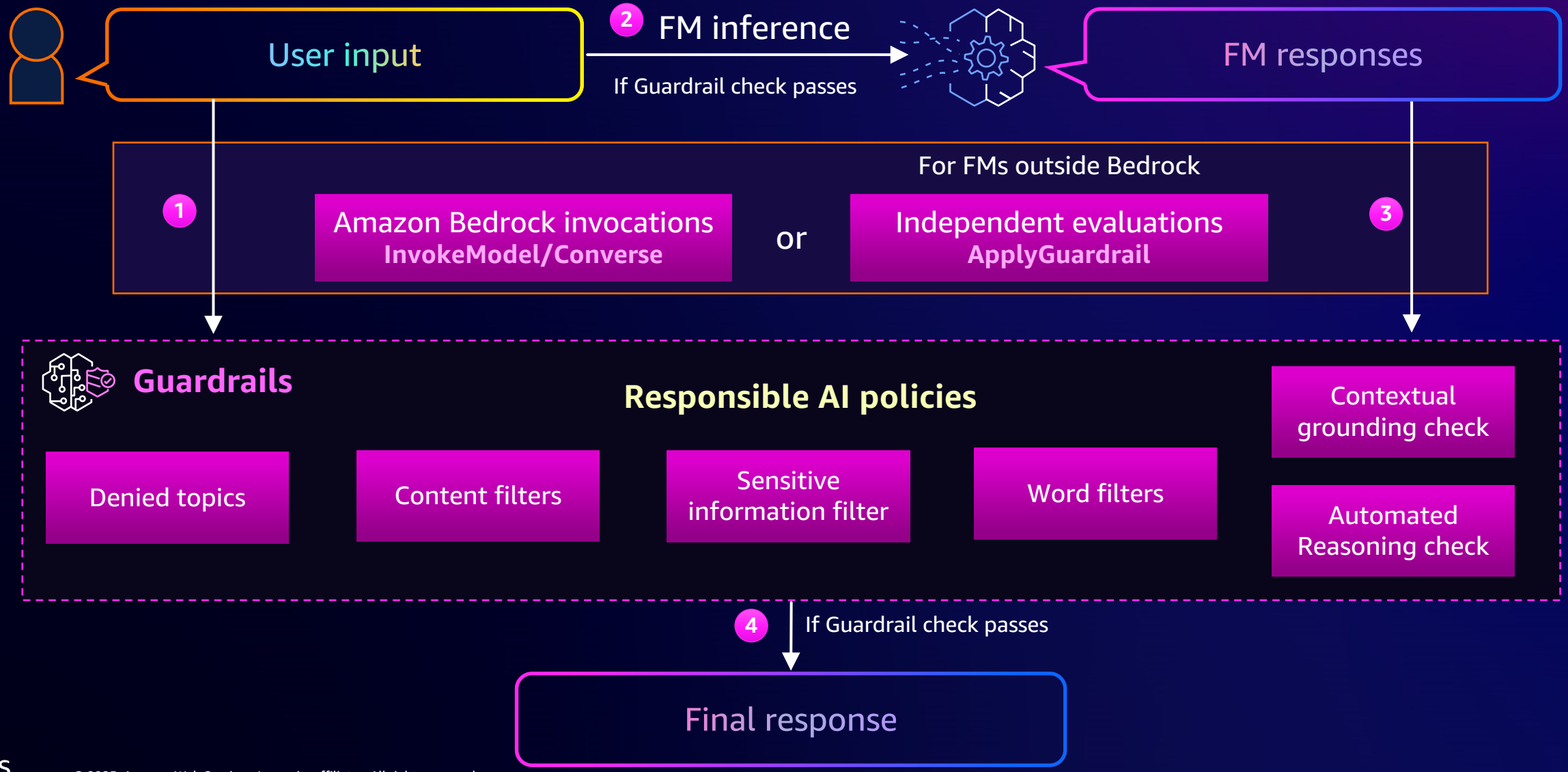
RAG doesn't stop hallucinations

Risk of exposing sensitive information

Good FM prompting isn't enough

Prompt attacks bypass and FM controls

How Amazon Bedrock Guardrails works

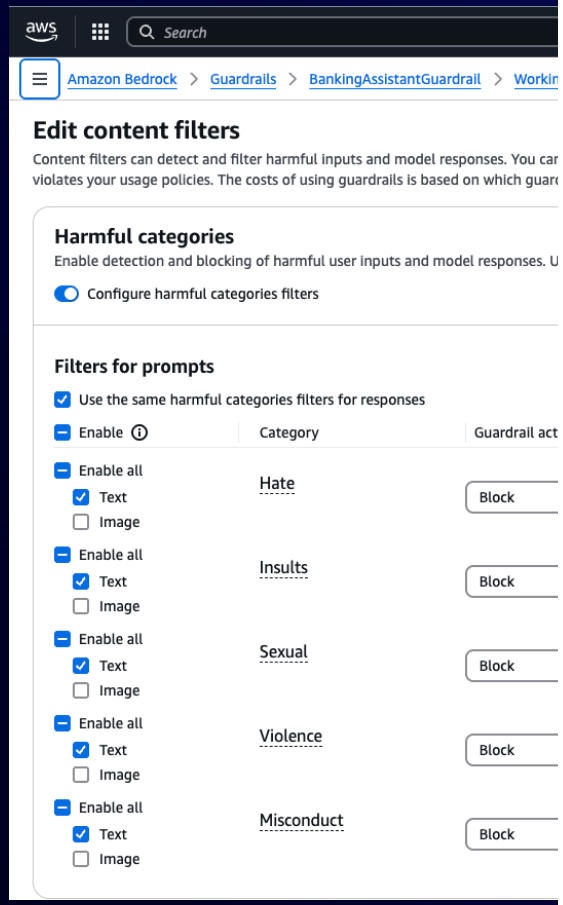


Deep dive into the components of a guardrail

Content filters

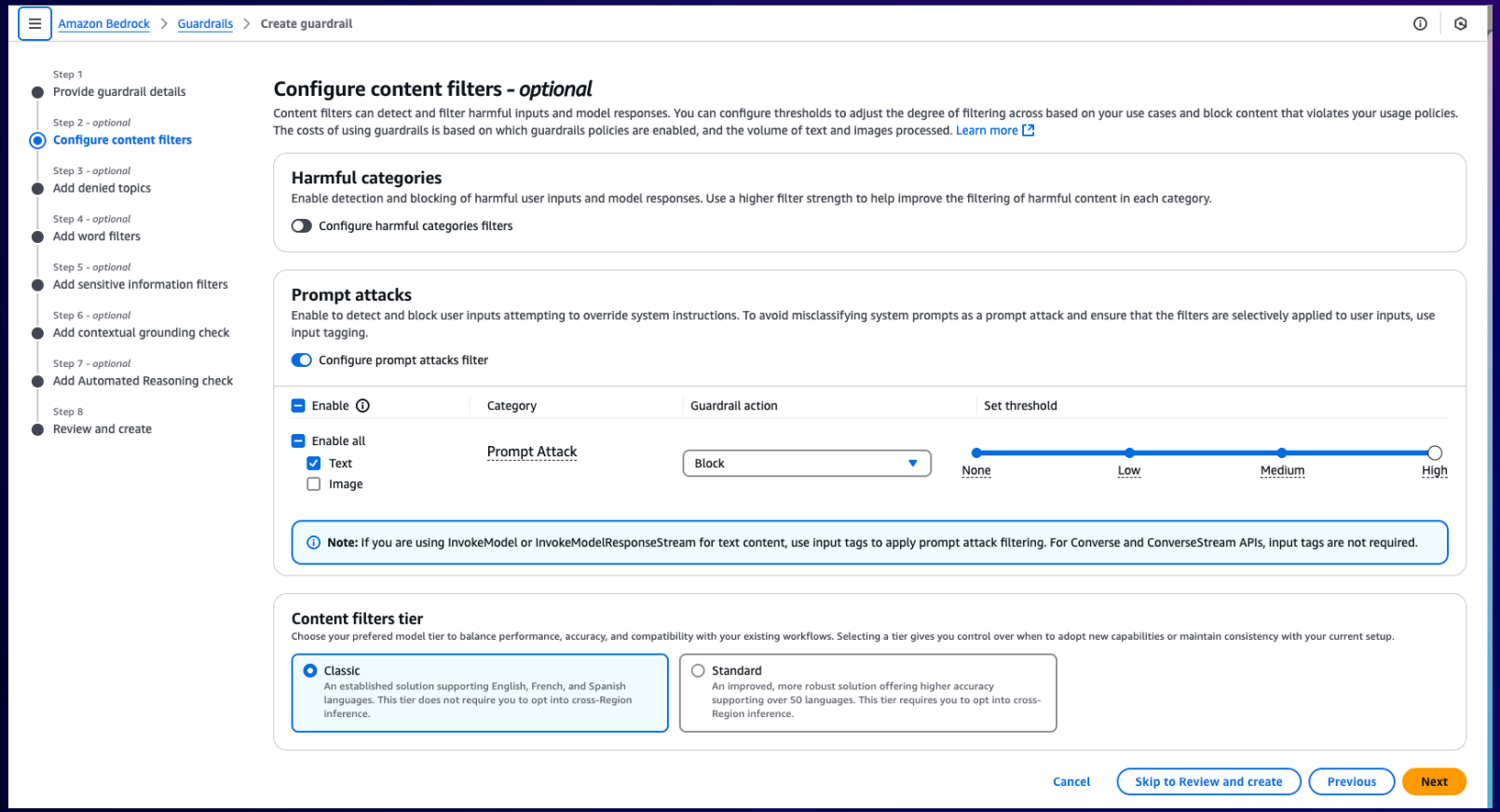
FILTER HARMFUL CONTENT ACROSS 6 CATEGORIES

- Hate
- Insults
- Sexual
- Violence
- Misconduct
- Prompt attack



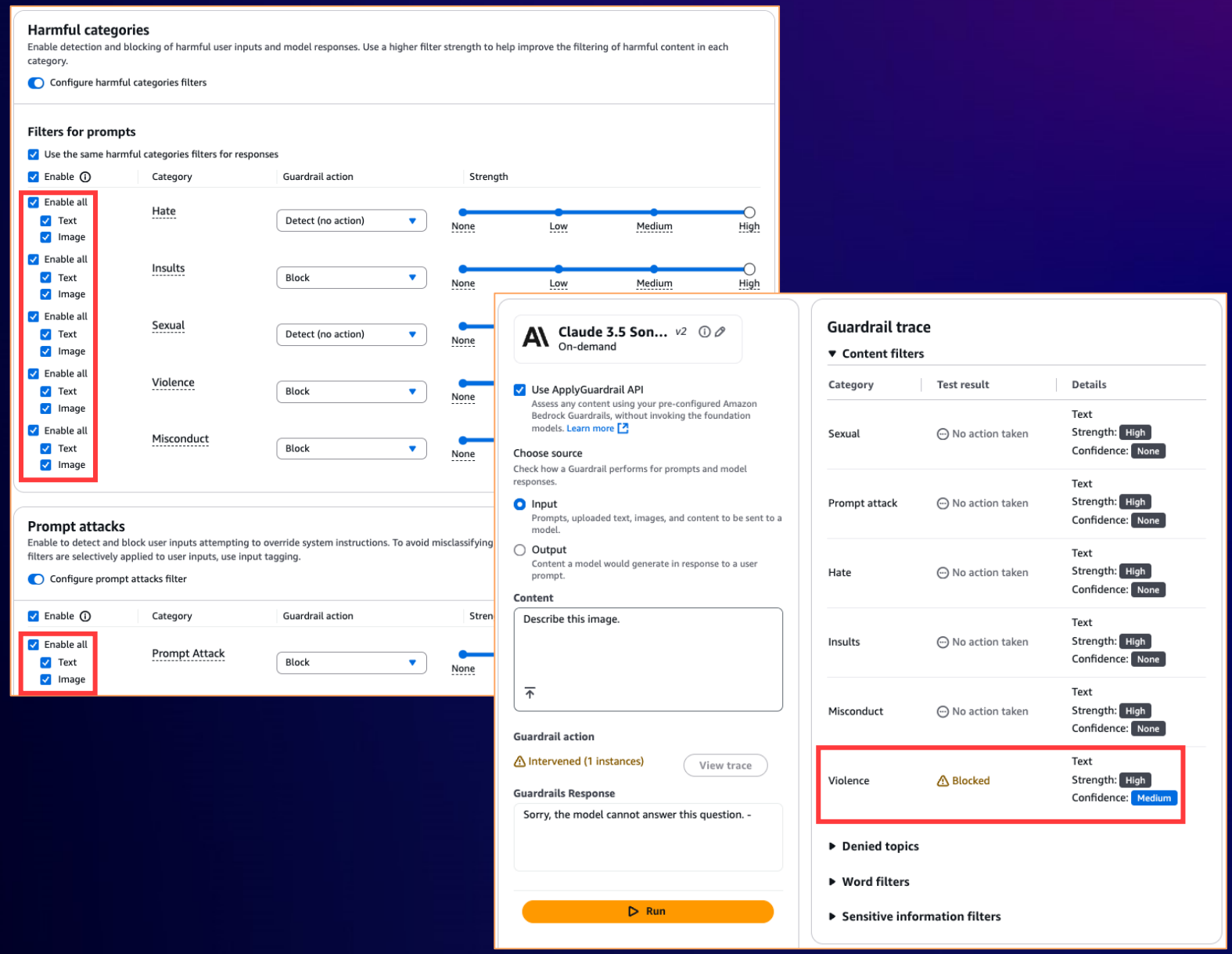
Prompt attacks detection

Similar to harmful categories, prompt attacks are detected based on classification confidence



Multi-Modal Toxicity Detection

- ✓ Content filters now supports toxic image detection, in addition to text
- ✓ Enhance user experiences and manage risk by detecting and filtering undesirable content across different media
- ✓ Content moderation for social media, e-commerce and advertising, financial fraud detection, educational and healthcare applications, among other applications
- ✓ Industry-leading, up to 88% harmful multimodal content detection rate

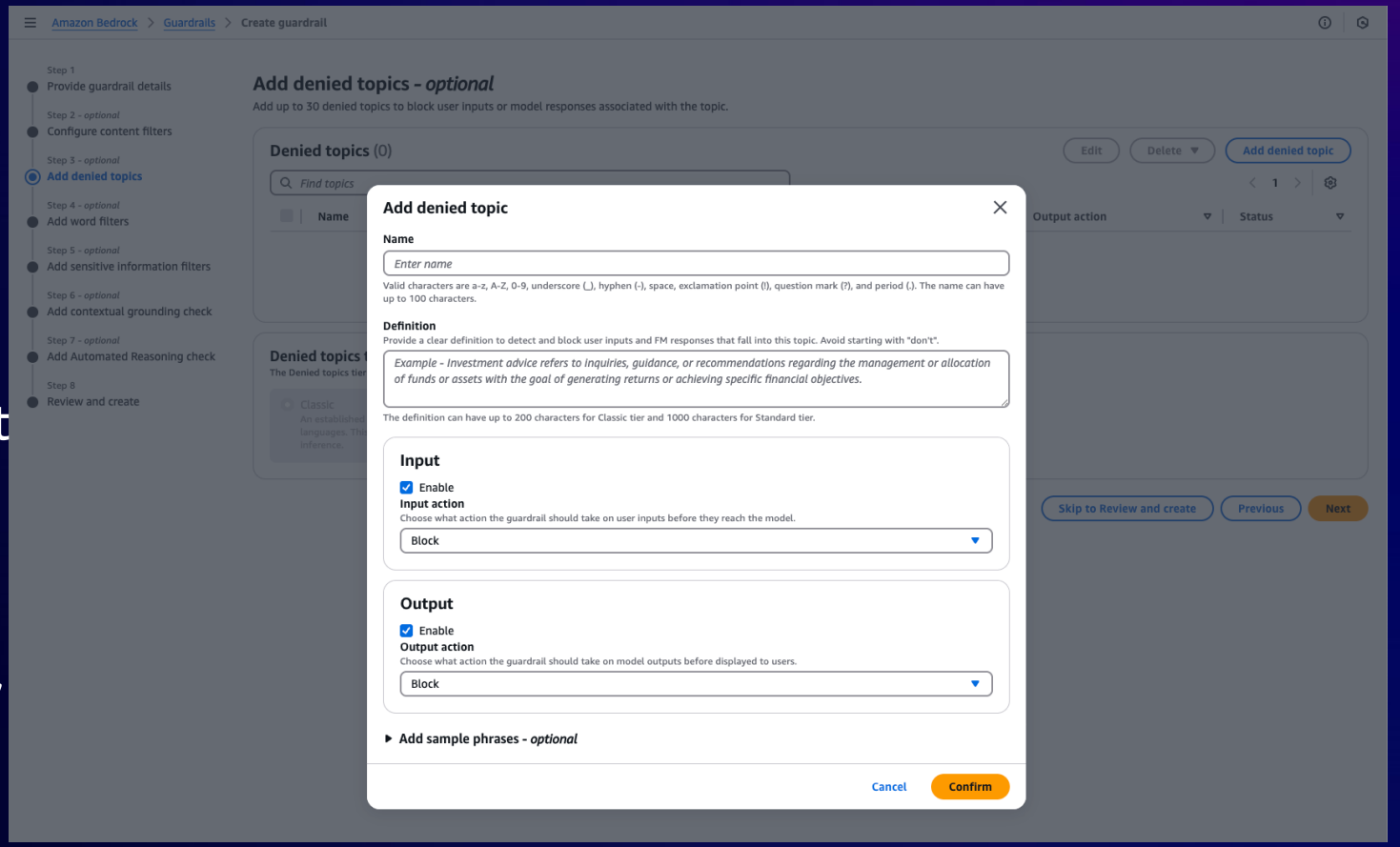


Denied topics

Topics are defined in simple language and compared against user queries/requests to determine similarity

Examples:

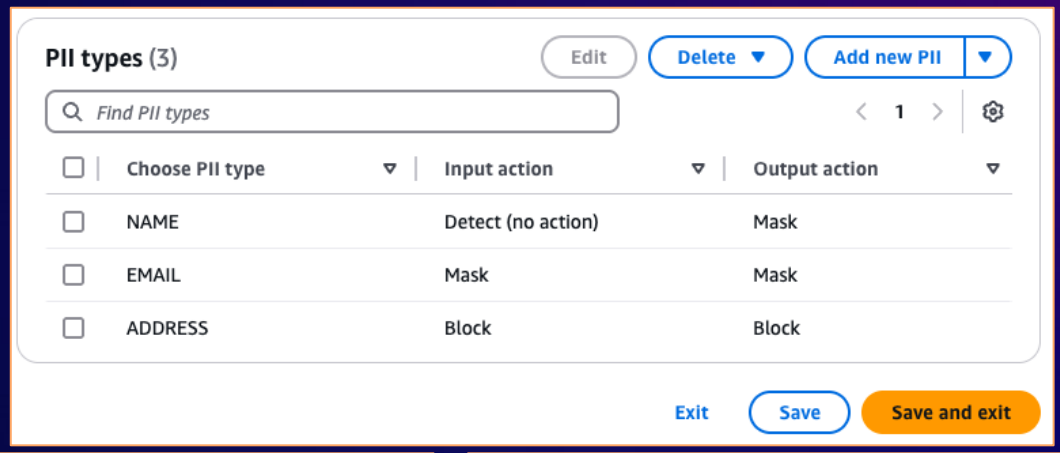
- **Substance use history—**past or present use of alcohol, tobacco, drugs, or medications outside the scope defined in the application process
- **Financial information—**income, assets, debts, credit score, or financial details not directly relevant to the insurance product applied for

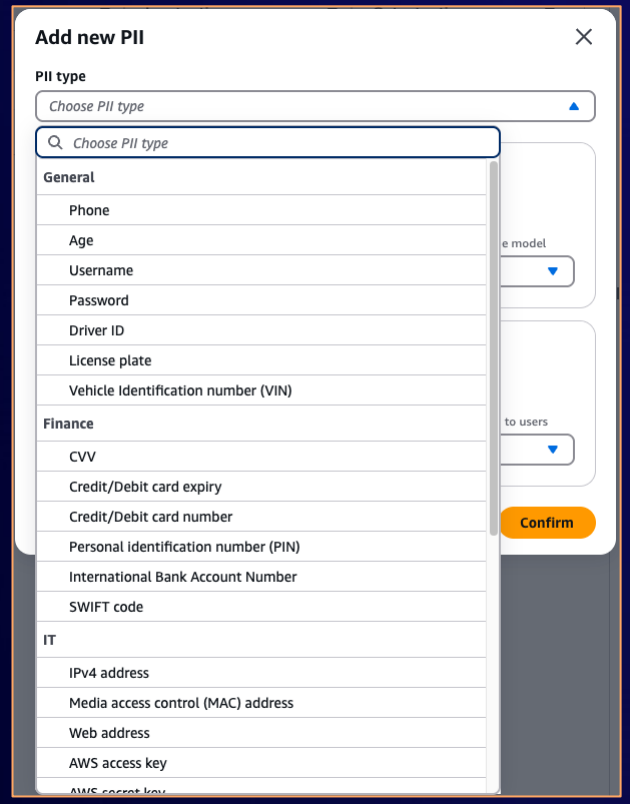


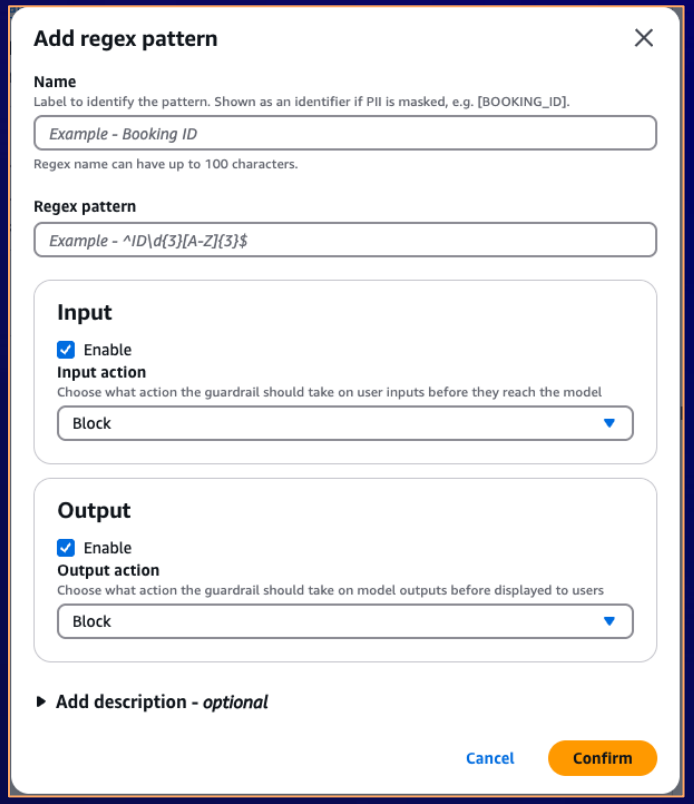
Sensitive information filter

PROTECT YOUR SENSITIVE DATA

- ✓ Detect, redact or filter PII in FM responses to protect user privacy
- ✓ Detect, redact or filter PII in user inputs
- ✓ Select from a variety of pre-configured PII types, based on application requirements
- ✓ Or define your own sensitive information, by using regular expressions (regex)
- ✓ Apply actions for prompts and/or responses independently



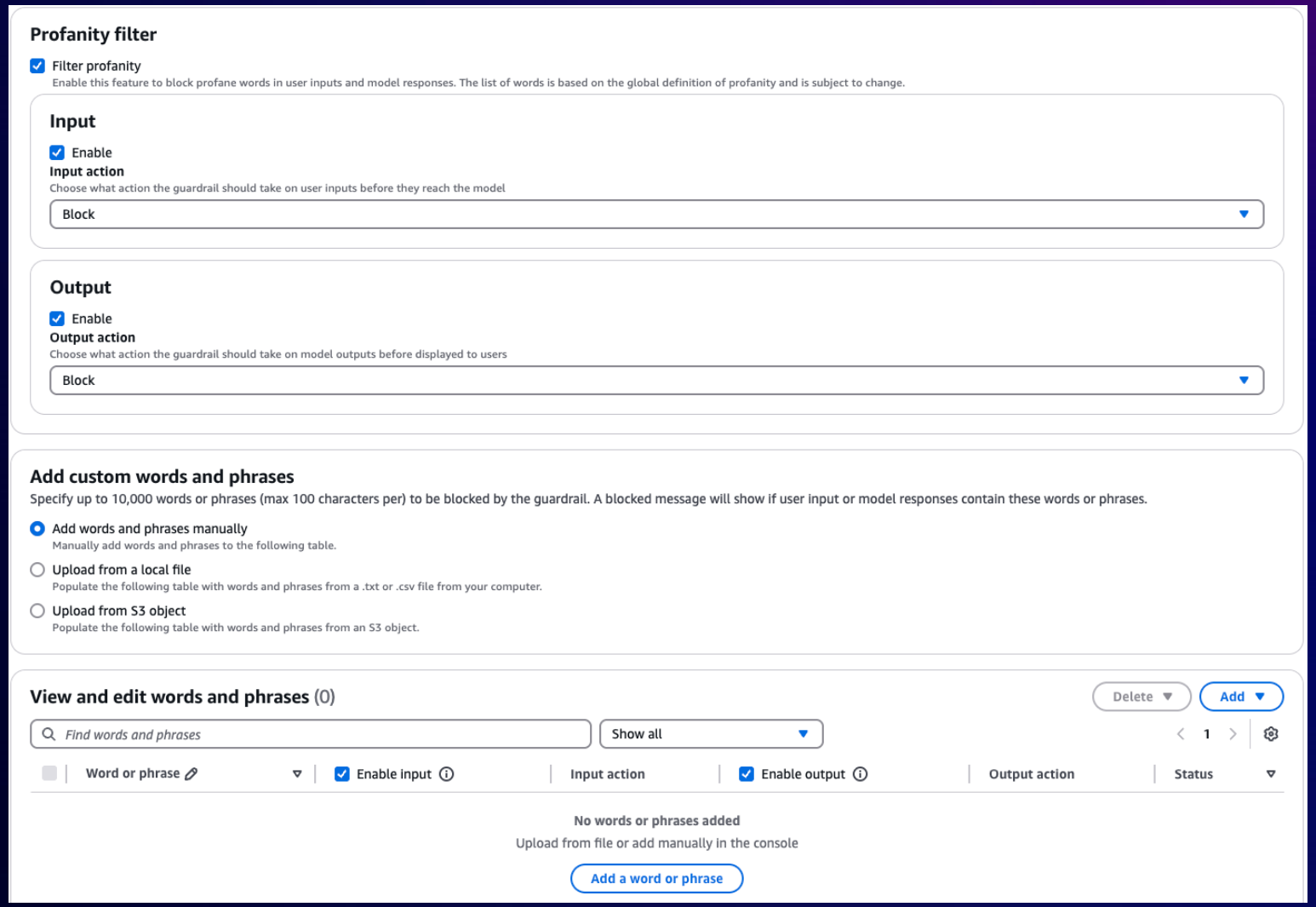




Word filters

BLOCK SPECIFIC WORDS OR PHRASES

- Filter profane words
- Define a set of custom words to block in user input and FM responses



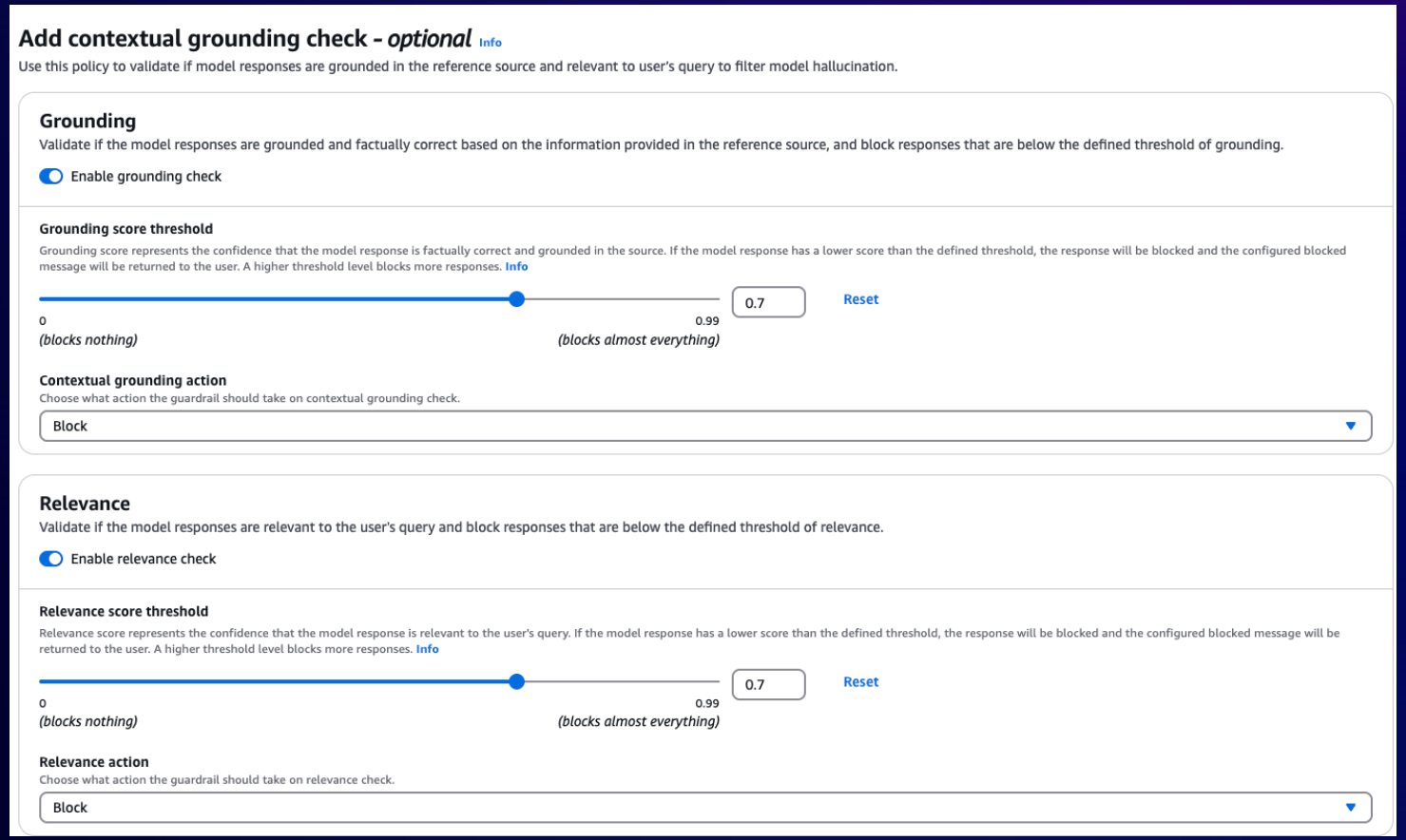
Contextual Grounding checks & relevence

REDUCE HALLUCINATION BY FILTERING UNGROUNDED AND IRRELEVANT RESPONSES

- Filter hallucinations in RAG and summarization applications
- Check response accuracy based on your enterprise data
- Check if the responses are relevant to user's query or instruction

Grounding threshold : Based on the information in your RAG response and the FM answer, Grounding evaluates the correctness of the answer

Relevance threshold: Evaluate the FM response and the user query. If the confidence that the response answers the query is below the threshold, a blocked message is returned to the user



Configure Guardrails' actions

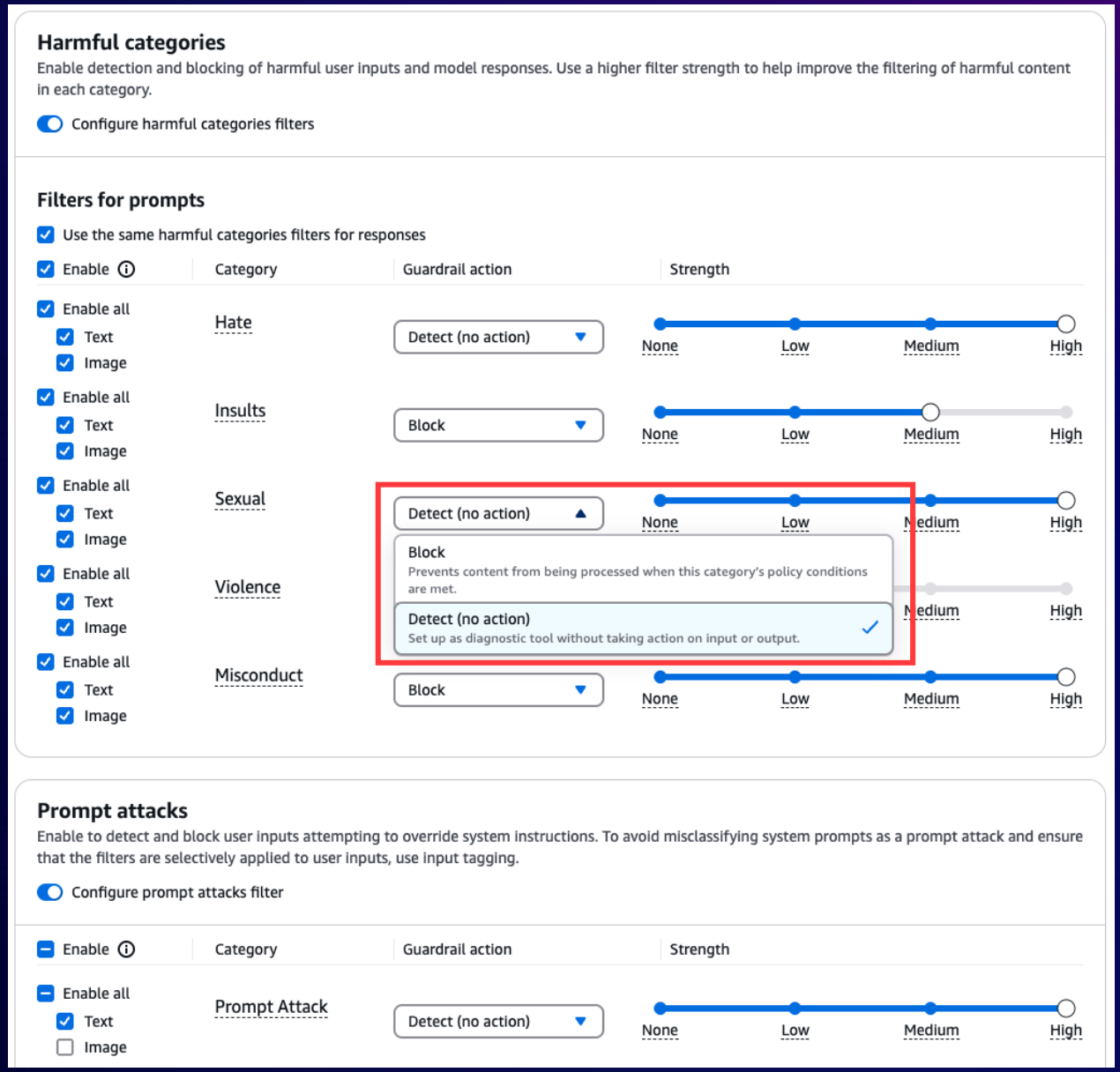
For each policy, select between:

- Block – For preventing content from being delivered when conditions are met

or

- Detect – For monitoring only, without intervention in the actual content flow

Available for all configurations, including content filters, denied topics, sensitive information filters, word filters and contextual grounding.

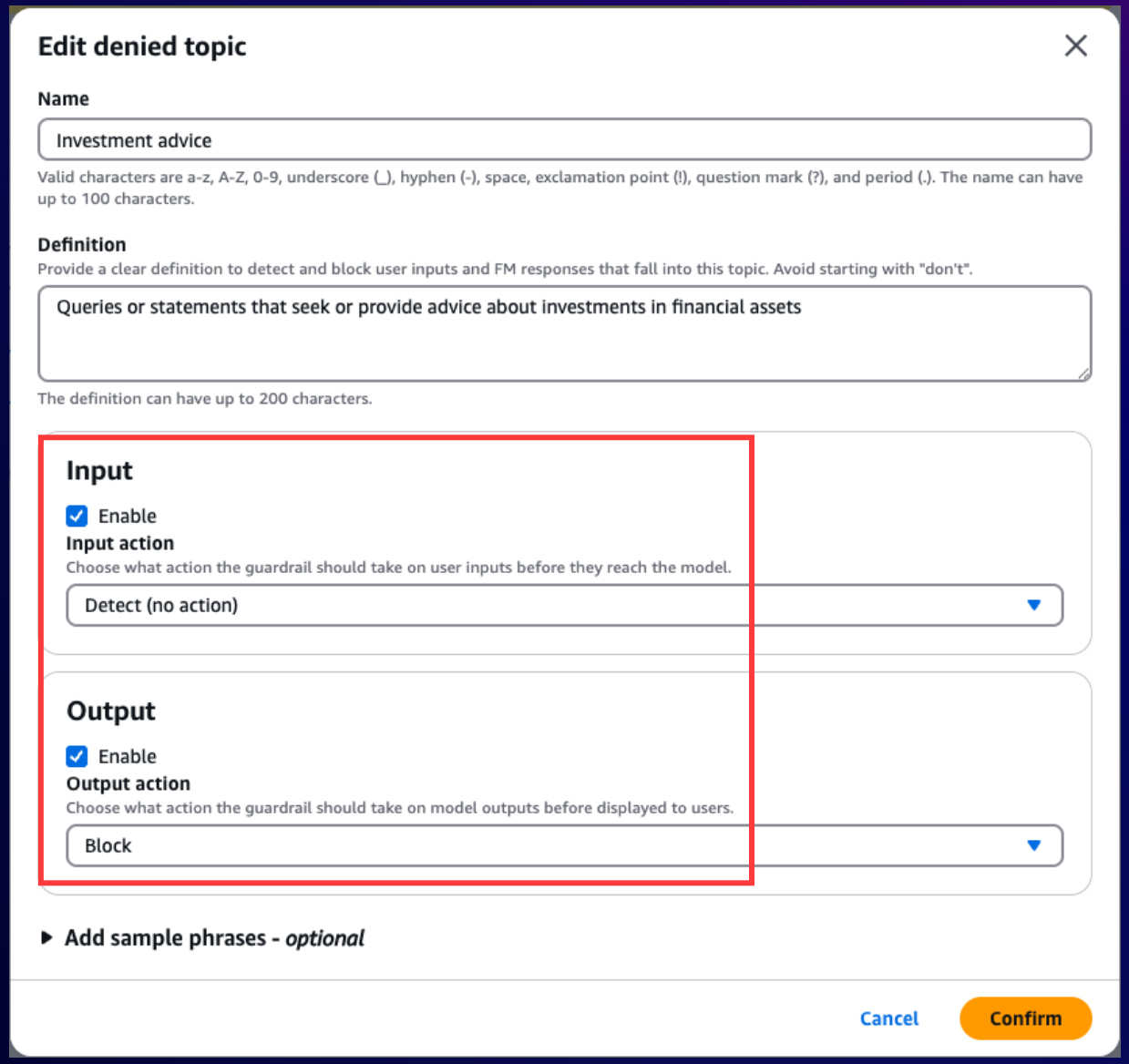


Configure Guardrails' directions

Decide independent configurations and actions for each direction of the flow:

- Input – For evaluations on user's inputs
- Output – For evaluations on models' outputs

Available for content filters, denied topics, sensitive information filters, and word filters.

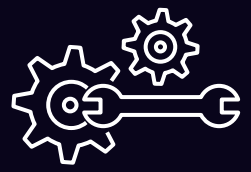


Integration with existing systems & applications

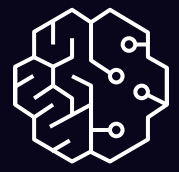
SEAMLESS ADOPTION WITHIN YOUR CURRENT INFRASTRUCTURE

Seamless API integration

Easy incorporation into current workflows and applications



separated api to allow use independent of first, third-, or external-party FM use

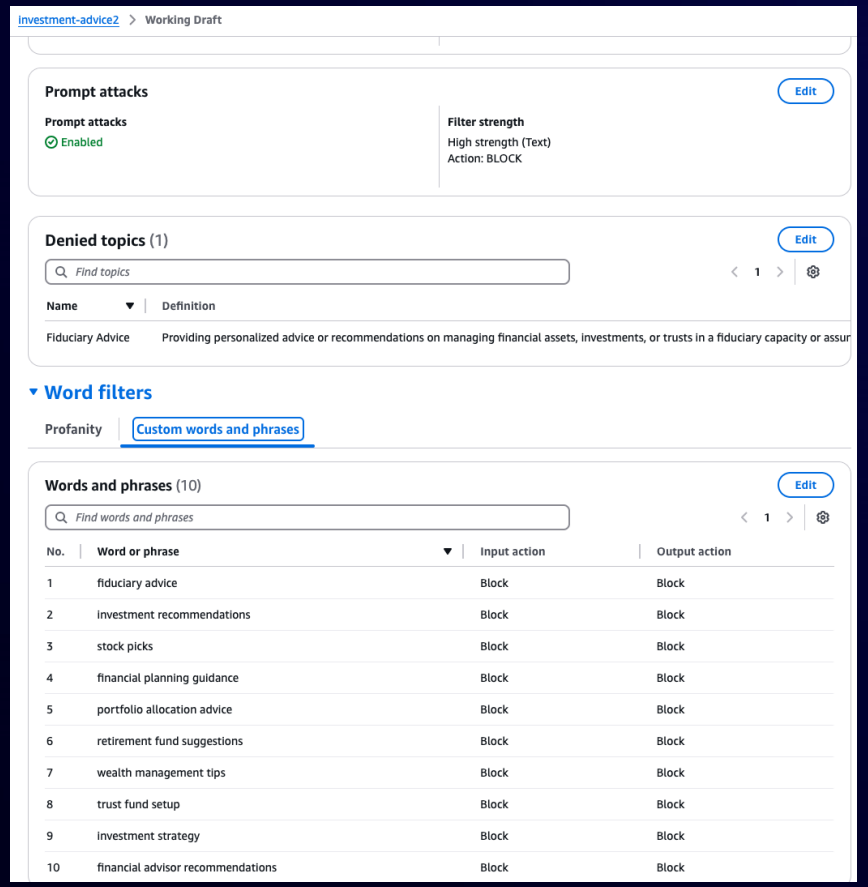


Guardrails scale to enterprise applications with parallel execution

Tailored guardrails for different use cases and departments

JSON responses allow easy integration with existing observability tools

Strands Agents with Bedrock Guardrails




```python
# Create a Bedrock model with guardrail configuration
bedrock_model = BedrockModel(
    model_id="us.amazon.nova-premier-v1:0",
    guardrail_id=guardrail_id,
    guardrail_version=guardrail_version,
    boto_session=boto3.Session(region_name="us-west-2"),
    guardrail_trace="enabled"
)

# Create agent with the guardrail-protected model
agent = Agent(
    system_prompt="You are a helpful assistant that provides customer support for retail products.",
    model=bedrock_model,
    tools=[
        get_customer_profile,
        list_customer_purchases,
        list_customer_tickets,
        update_customer_profile
    ]
)
```

User: This is my account number 923987 can you tell me what stocks to buy based on this info

Agent: I apologize, but I am not able to provide fiduciary advice. Additionally, it seems that you may have included some sensitive personal or financial information in your request. For your privacy and security, please modify your input and try again without including any personal, financial, or restricted details. ⚠️ GUARDRAIL INTERVENED!

Code sample
https://github.com/strands-agents/samples/tree/main/01-getting-started/06-guardrail-integration

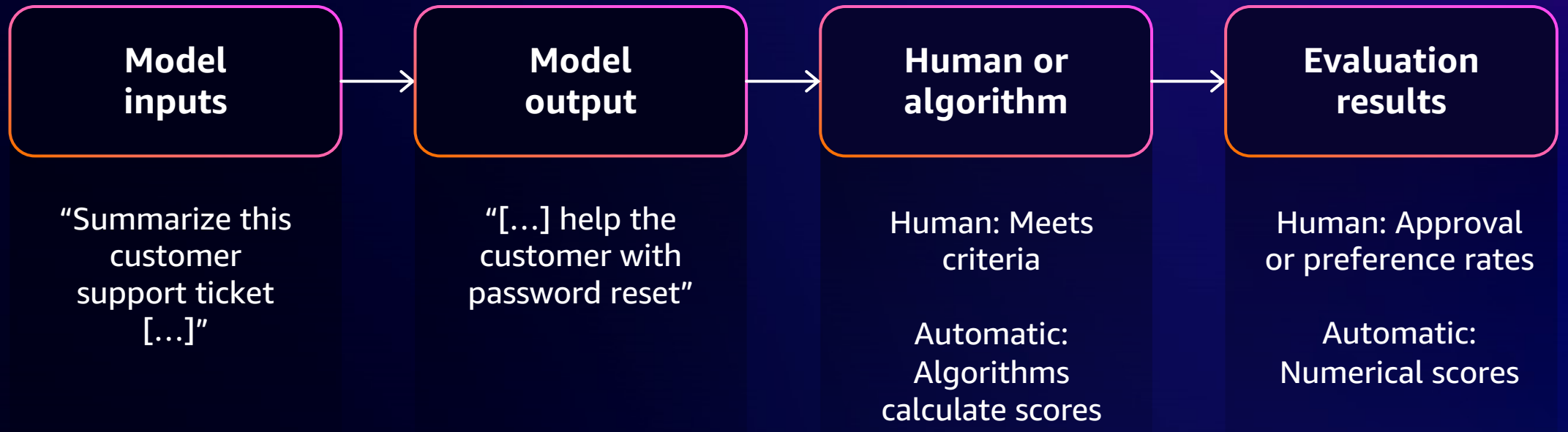
Notebook Exercises

GENAI BOOTCAMP

Model + RAG + Agent Evaluation

What is model evaluation?

1. Quality



What is model evaluation?

QUALITY

TRADEOFF

COST

LATENCY

Selecting an LLM: Multiple priorities

Cost > Precision > Speed

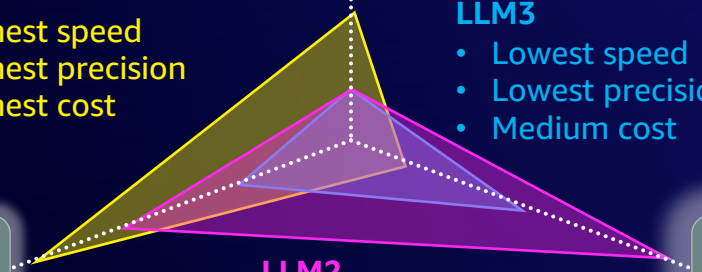
Model	Speed
LLM1	2/5
LLM2	1/5
LLM3	1/5

Speed

Lowest priority

LLM1
- Highest speed
- Highest precision
- Highest cost

LLM3
- Lowest speed
- Lowest precision
- Medium cost



!! Medium priority

Precision

Model	Evaluation Score
LLM1	5/5
LLM2	4/5
LLM3	3/5

LLM2
- Lowest speed
- Medium precision
- Lowest cost

Winner: LLM2

Cost

Highest priority !!!

Model	Cost
LLM1	$$$$
LLM2	$$
LLM3	$$$

Why is foundation model evaluation important?

Make quality, cost, and latency tradeoffs

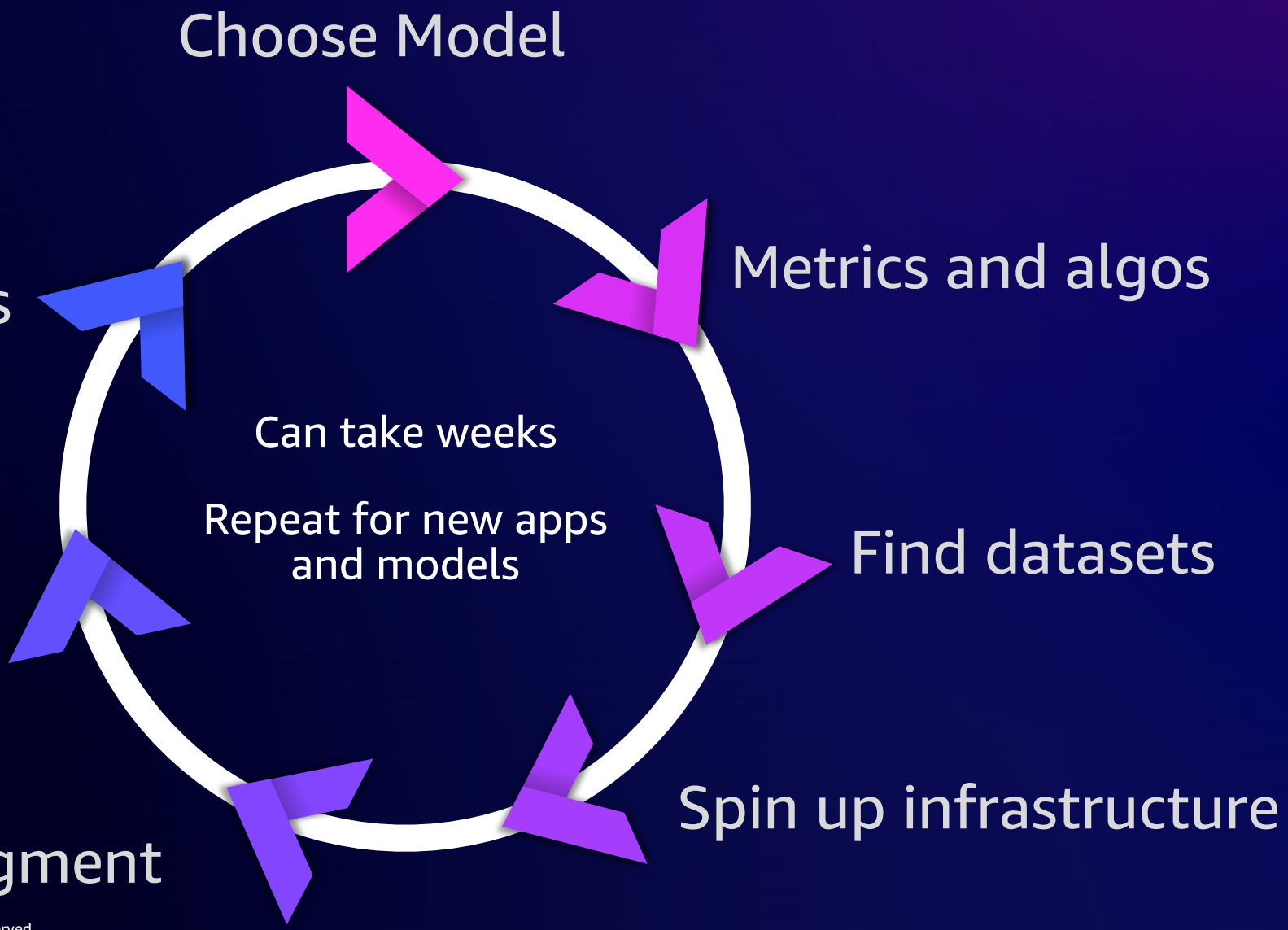
Align to your company’s style and brand voice

Evaluate for your specific use cases

Evaluate with your company’s data

Monitor biases, safety, and trust

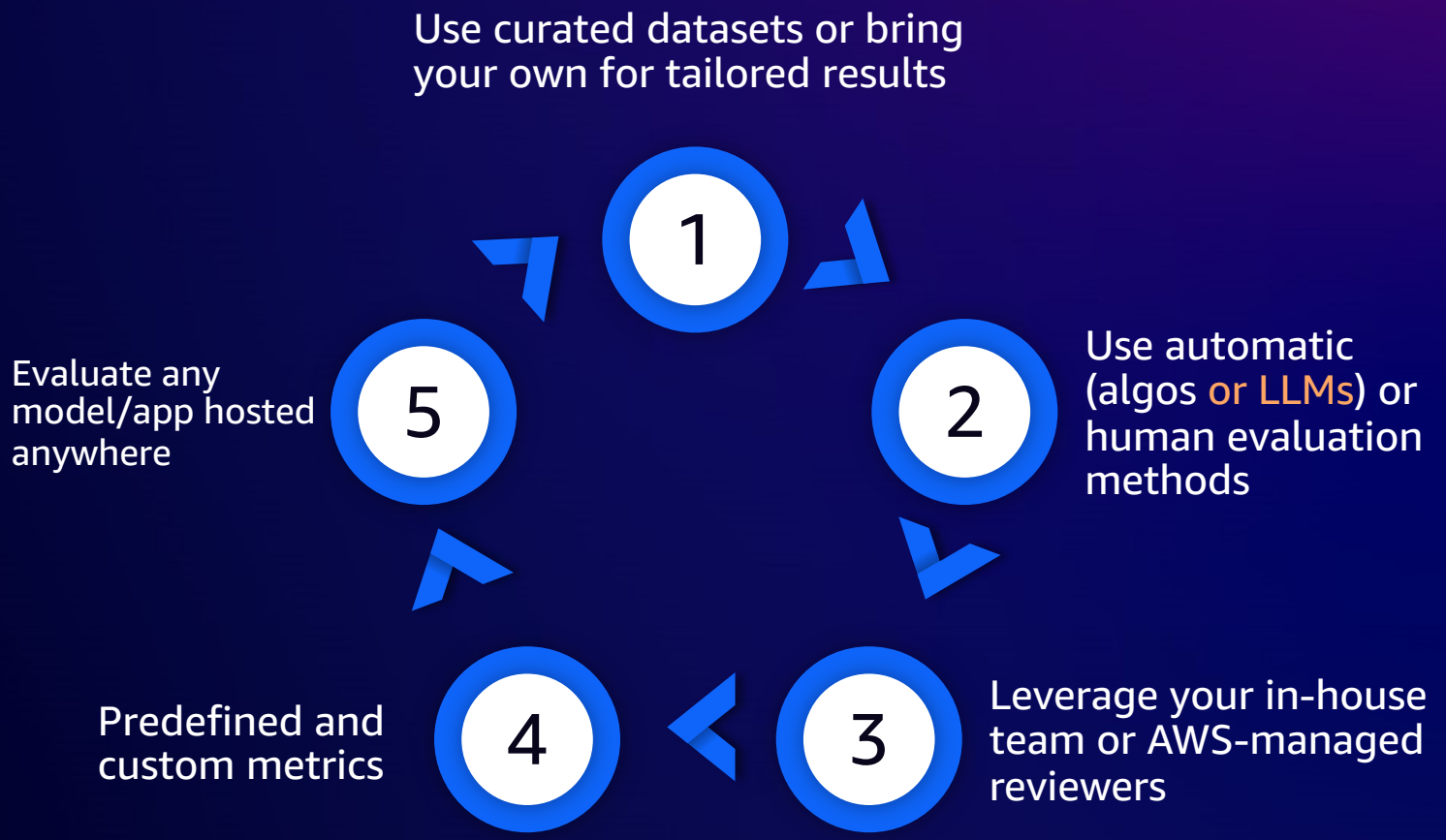
Model evaluation lifecycle and challenges



Amazon Bedrock model evaluation

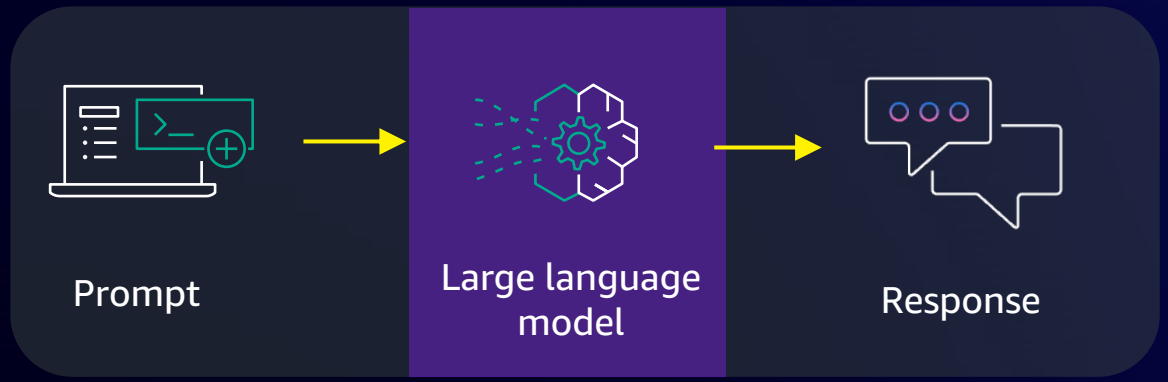
Evaluate, compare, and select the best foundation model for your use case

Public API
Evaluate custom models
Evaluate distilled models
Evaluate imported models
Evaluate prompt routers
Any model hosted outside Bedrock
Use an LLM-as-a-judge

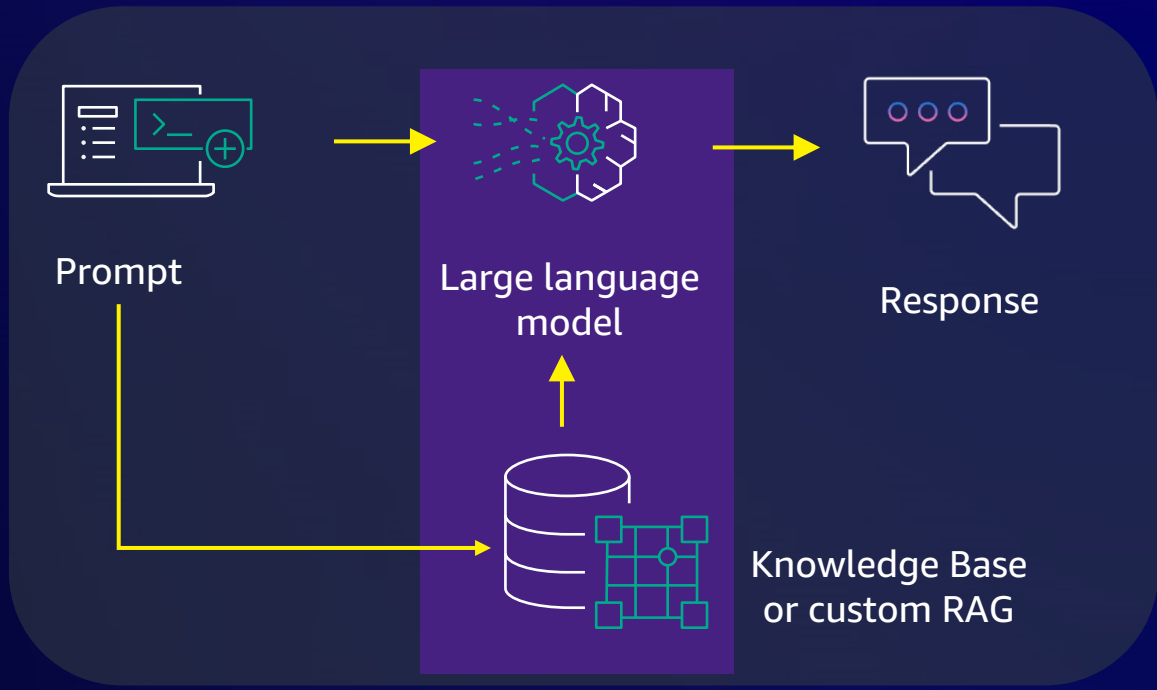


Evaluation Features - Bedrock

GenAI Model Evaluation



RAG Evaluation



Bedrock LLM evaluation options

Programmatic evaluation

Accuracy | Robustness | Toxicity

LLM-as-a-judge

Correctness | Completeness | Helpfulness

Relevance | Coherence | Readability

Human evaluation

Creativity | Style | Tone

Accuracy | Consistency | Brand voice

Algorithms

BERTScore | Classification accuracy
F1 | Real-world knowledge score

LLM reasoning

Multistep reasoning | Correlation with expert human evaluators

Rating methods

Thumbs up/down | 5-point Likert scales
Binary choice buttons | Ordinal ranking

How to set up LLM-as-a-judge model evaluations

Choose generator model

Select inference source – Bedrock models or Bring your own inference responses

Choose metrics or create your own custom metrics

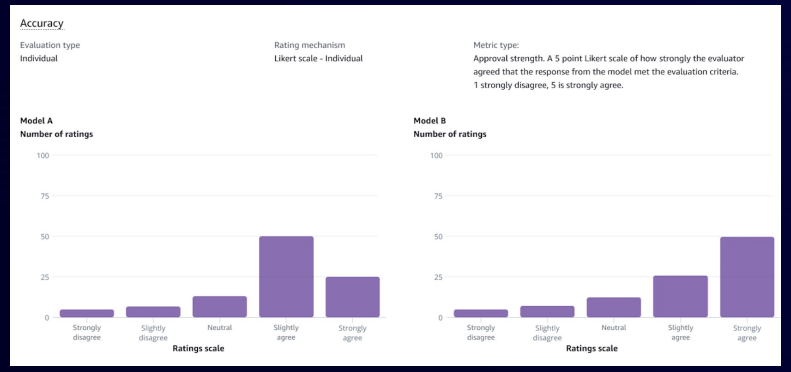
Choose curated or bring your own datasets

Inference and scoring

View results

Get results in a few clicks

Human evaluation reports

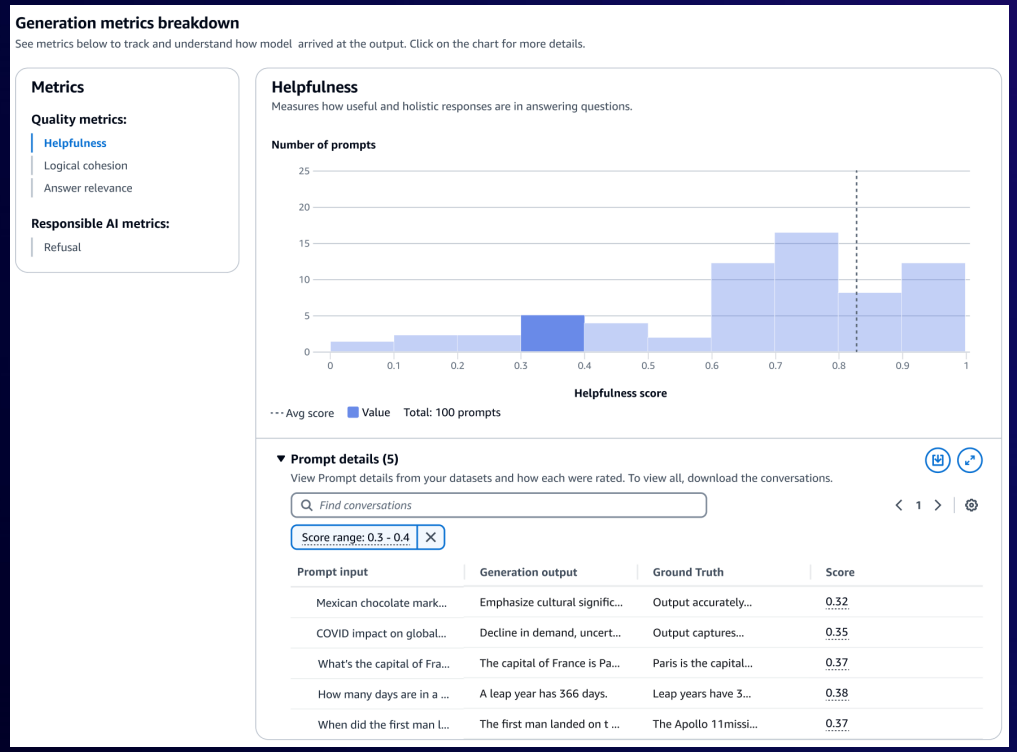


Track ratings from your team

See distributions visually

Simple metric explanations

LLM-as-a-judge reports

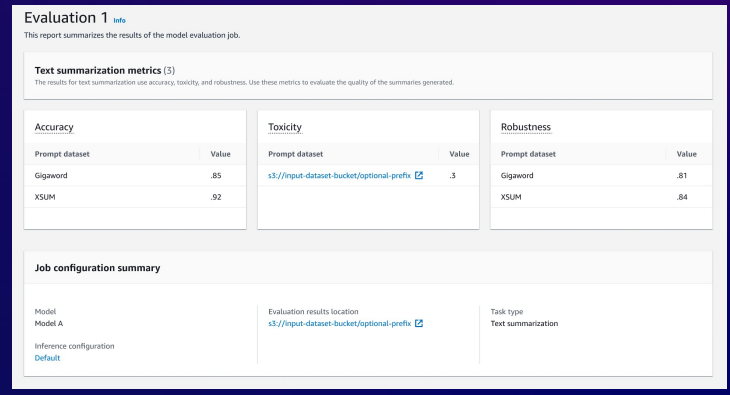


Simple to read scores

See distributions visually

See ratings explanations

Automatic evaluation reports



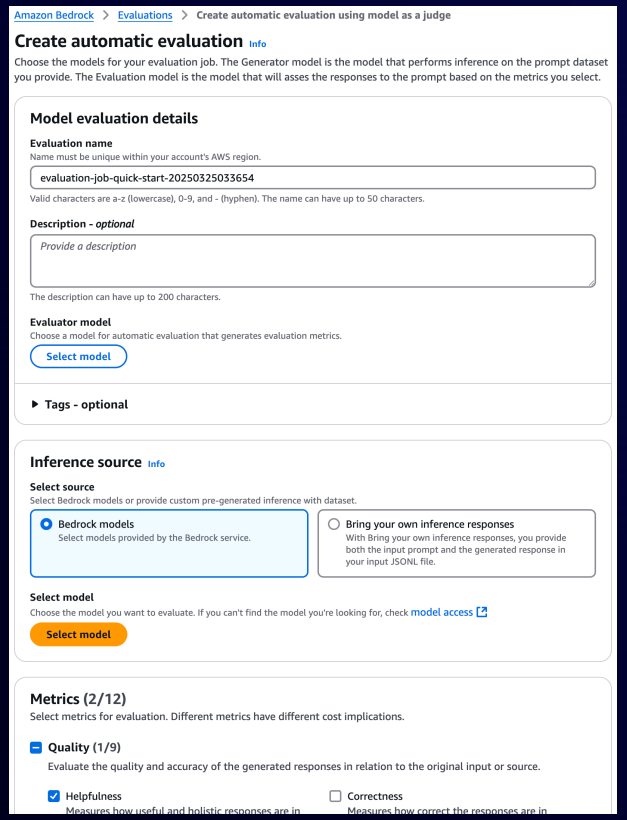
Simple-to-read scores

Curated per task type

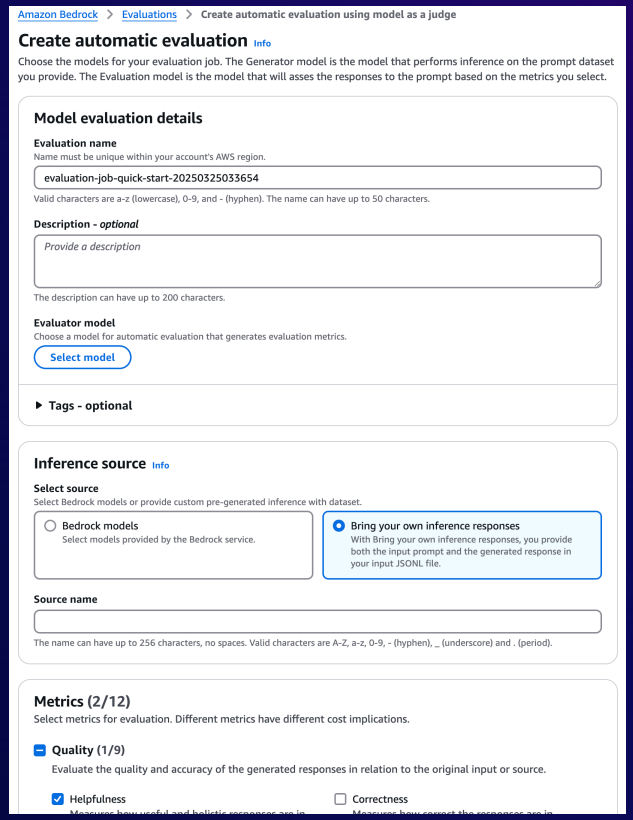
Popular algorithms utilized

Bring your own inference responses – model eval

Bedrock Model



Any model anywhere (or app)



- Evaluate any model or app hosted anywhere. No longer limited to Bedrock model calls for model eval
- Bring your responses in your input dataset (JSONL), skip the Bedrock model call (additional input JSONL fields supported)
- **Pro tip:** Not limited to models. If you only want to eval the response, it can be from anything, including whole apps

Amazon Bedrock RAG Evaluation

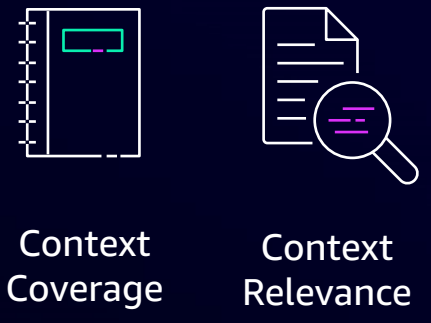
Evaluate your full RAG stack to optimize your application, including any application hosted anywhere

1. Bring your own datasets for tailored results
2. Evaluate retrieval alone or retrieval + generation with a choice of LLM-as-a-judge
3. Built-in metrics for quality and responsible AI, create custom metrics, compatible with Amazon Bedrock Guardrails
4. Compare across multiple evaluation jobs
5. Evaluate any RAG system hosted anywhere

aws

Choice of RAG evaluation metrics

Retrieval



Context Coverage

Context Relevance

Retrieval + Generation

Correctness

Completeness

Helpfulness

Citation precision

Citation coverage

Logical coherence

Faithfulness

Responsible AI

Harmfulness

Stereotyping

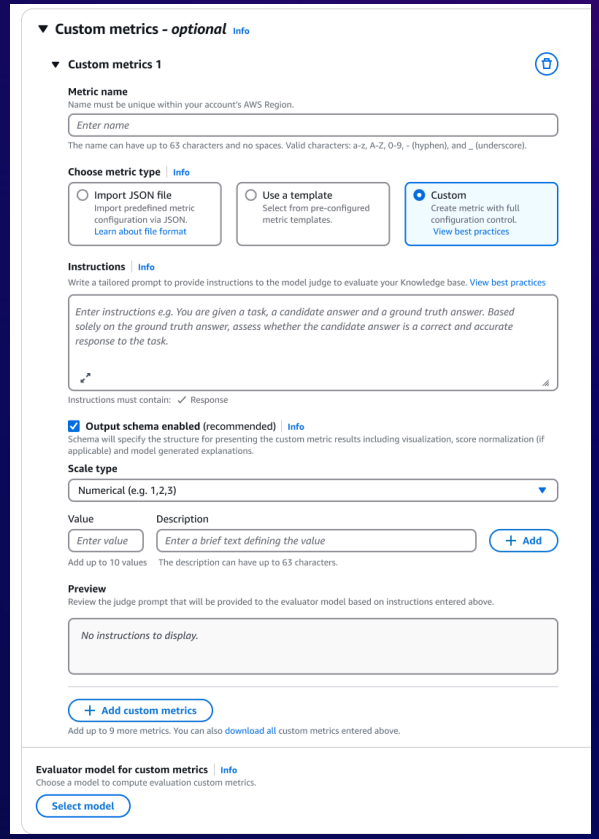
Refusal

Custom metrics

Type metric rubric (templates available)

Define grading scale

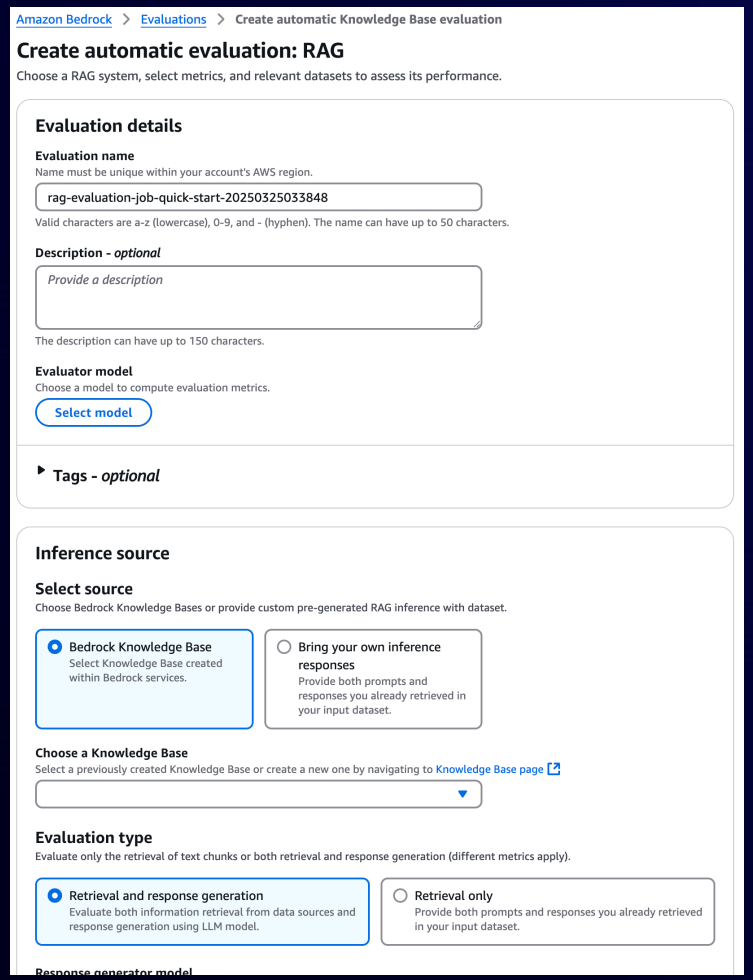
See preview of the full judge prompt



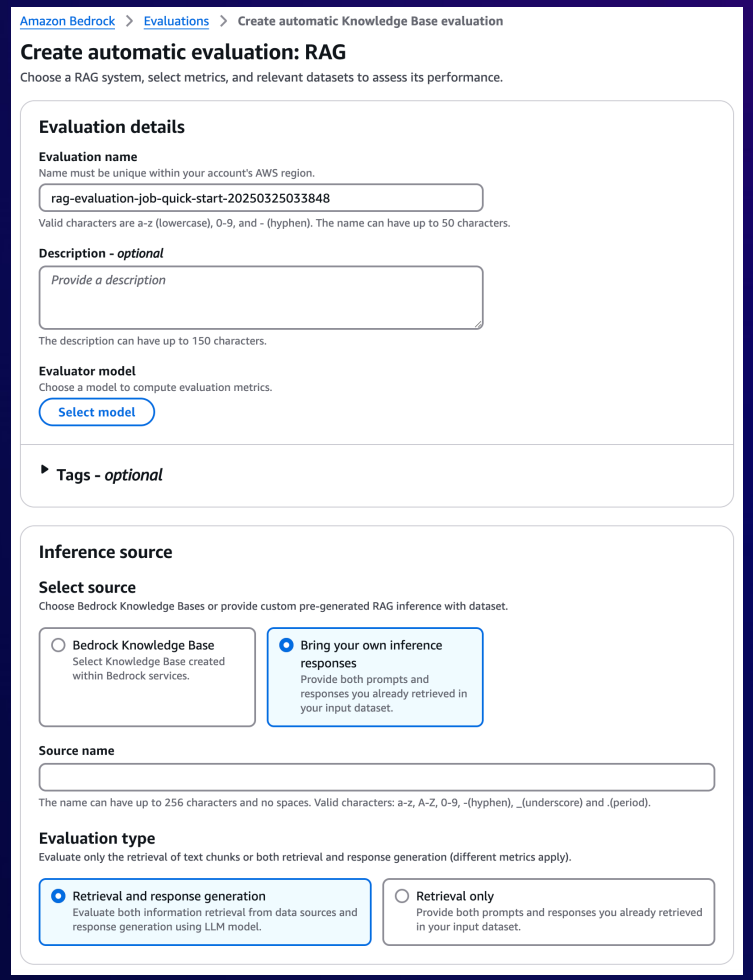
- Define your own metric for LLM-as-a-judge in model and RAG evaluation
- Variables for data injection from dataset and responses
- Save and reuse metrics later

Bring your own inference responses – RAG eval

Bedrock KB



Your own RAG system



- Evaluate any RAG system hosted anywhere. Not limited to Bedrock Knowledge Bases
- Bring your responses and retrieved passages in your input dataset (JSONL), skip the Bedrock KB call (additional JSONL fields in input dataset)

How RAG evaluation works

Choose evaluator model

Select inference source – Bedrock Knowledge Base or Bring your own inference responses

Choose to evaluate retrieval only or retrieve + generate

Choose your generator model

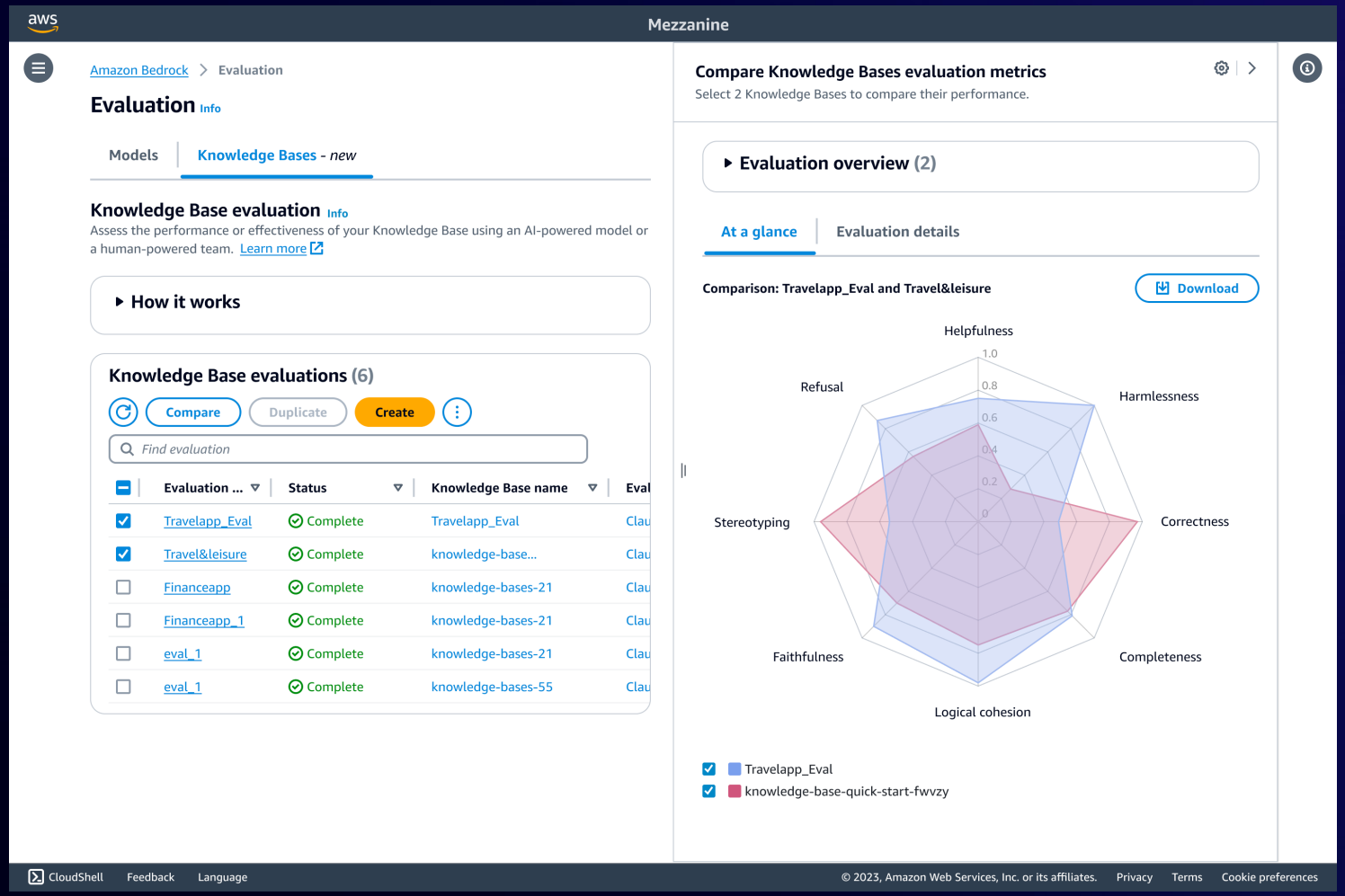
Choose your metrics or create your own custom metrics

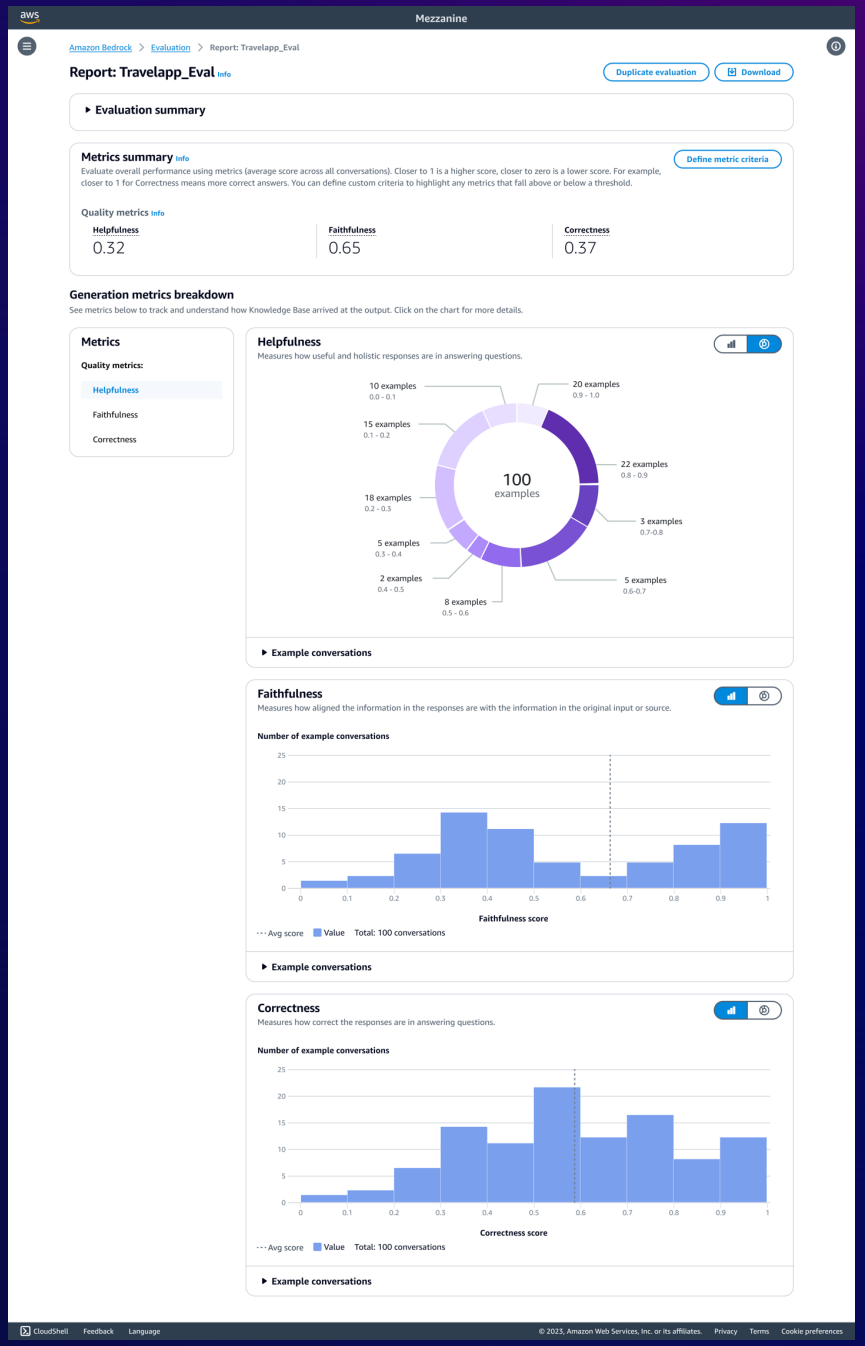
Upload your prompt dataset

Inference and evaluation

View results

RAG Evaluation Reports





Agent Evaluation
Why it matters for production?

Agents can take many paths to a correct result

Success isn’t just about the outputs

Common LLM evaluation methods are not sufficient

Recommendation 1 Step-by-Step evaluation

Agents components can fail independently (router/skills/tools)

-Assess router performance on tool selection

-Add skill specific evaluation(RAG, Code, API)

-Use both deterministic and LLM-as-a-judge approaches

Recommendation 2 Track path and convergence

Agents can get stuck in loops, or take inefficient paths, this impacts cost, latency and user experience

-Use an iteration counter and track step sequences

- Manually inspect traces during implementation

-Monitor for loops, unnecessary returns to router and redundancy

-Use both deterministic and LLM-as-a-judge approaches

Recommendation 3 Implement test case strategy

Test as many query variations as possible, include edge-cases, off-topic queries and parameter overlaps

- Evolve the cases with production data and discovered failures

-Focus on path coverage and not on volume

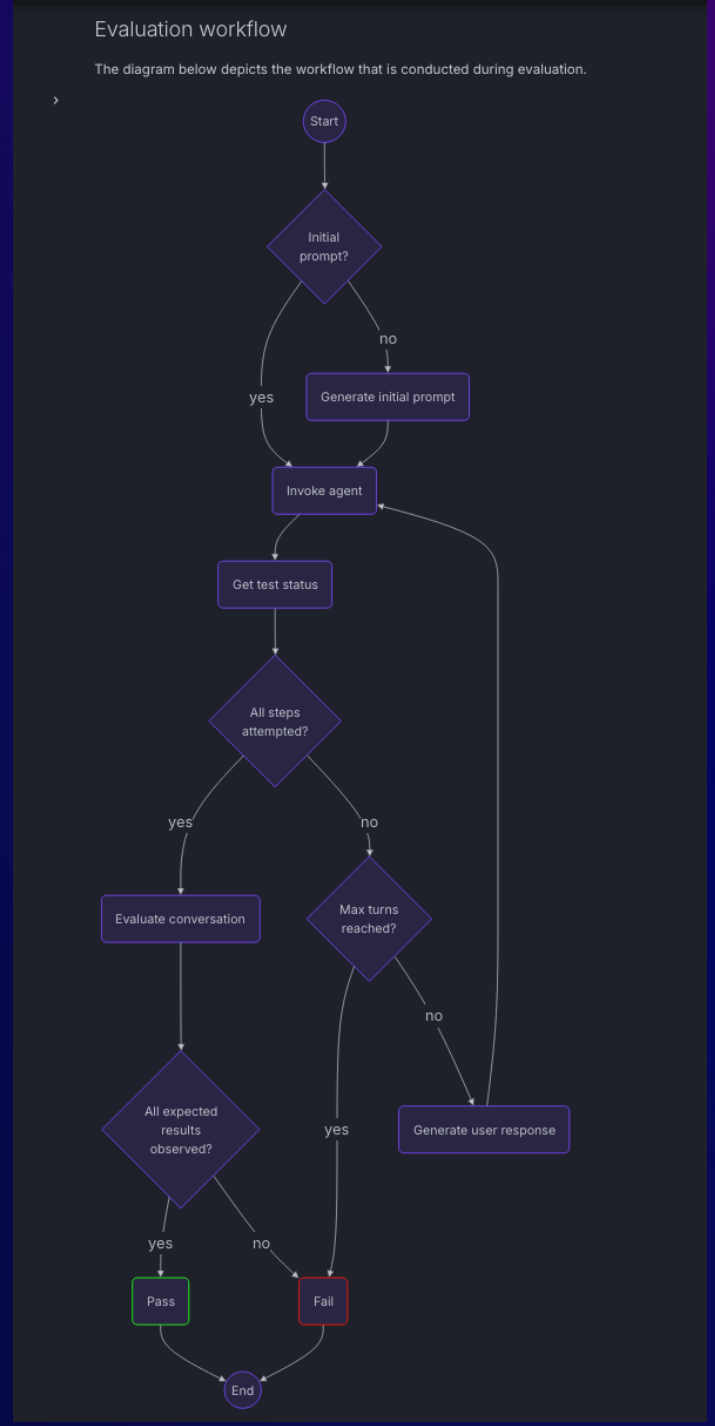
-Use both deterministic and LLM-as-a-judge approaches

Evaluation approaches

- Deterministic: anything that can be coded or calculated
- LLM-as-a-Judge(LLMaaJ): Qualitive assessment – using stress-tested prompts
- Hybrid: A combination of both for comprehensive coverage
- AgentCore Eval

Agent evaluation sample

https://awslabs.github.io/agent-evaluation/evaluators/



Open source options for evaluation

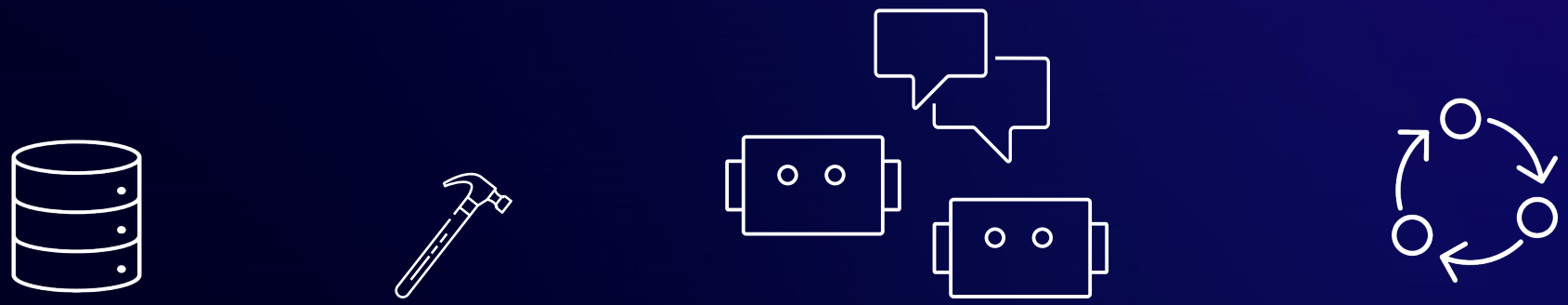
- Available templates and guidance to run agent evaluation
- Easy to integrate with Bedrock and AgentCore observability

Welcome to the final day of the Learning Week

Production ready agents with: **Bedrock AgentCore**

Implementing an AI Agent is complex

But open source frameworks make it easy to build agents and create proof of concepts (PoCs)



Context

Tools and tools execution

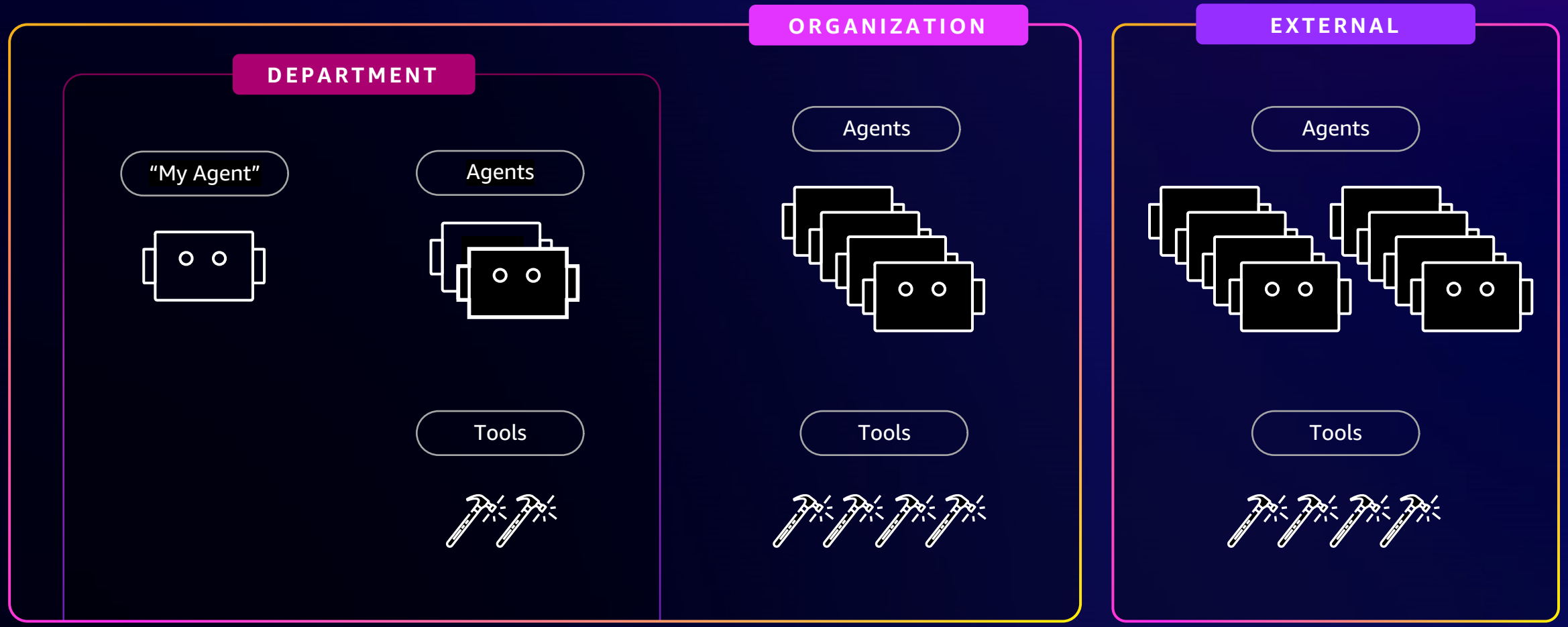
Multi-agent collaboration

Orchestrating between actions

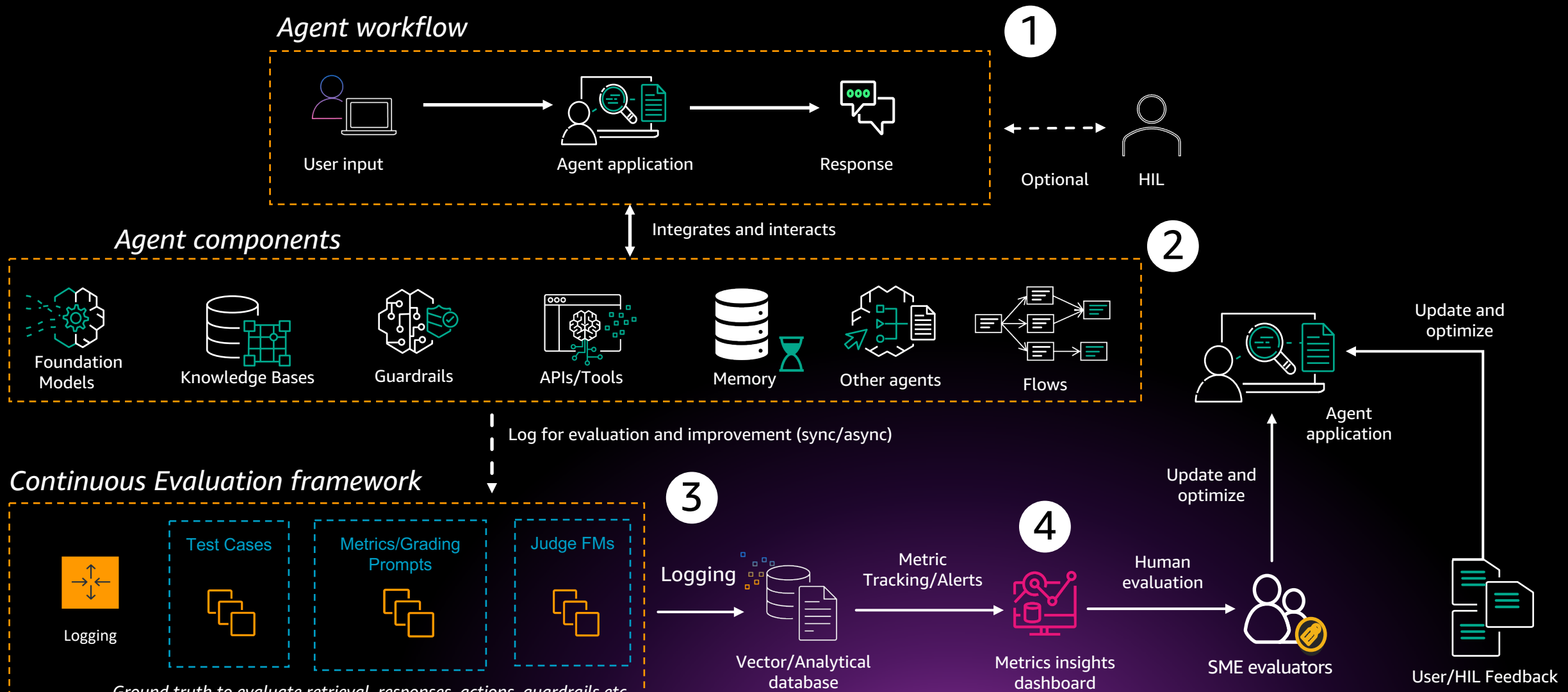
Functionality available

Strands Agents LangChain LangGraph crewai

Building an agent is the start, production scale is the goal



What does an agentic system look like?



Scale requires operational excellence

Excitement and potential

POC

Challenges on the path to production

Performance

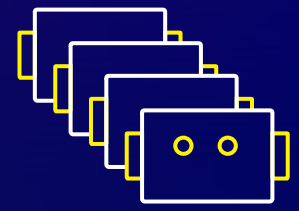
Scalability

Security

Context

Governance

Meaningful business value



AI production agents

Fundamentals for agentic AI Scale

SECURITY & TRUST

Ensure agents can act safely across system boundaries, protect data, ensure compliance

RELIABILITY & SCALABILITY

Support autonomous actions reliably at enterprise scale

IDENTITY

Authenticate and authorize all actors – agent and users

OBSERVABILITY

Track and monitor agent behaviors and decisions

DATA FOR CONTEXT

Feed agents with rich, contextually aware knowledge

INTEGRATIONS

Connect seamlessly across systems and tools

MEMORY

Agents need to maintain context across interactions

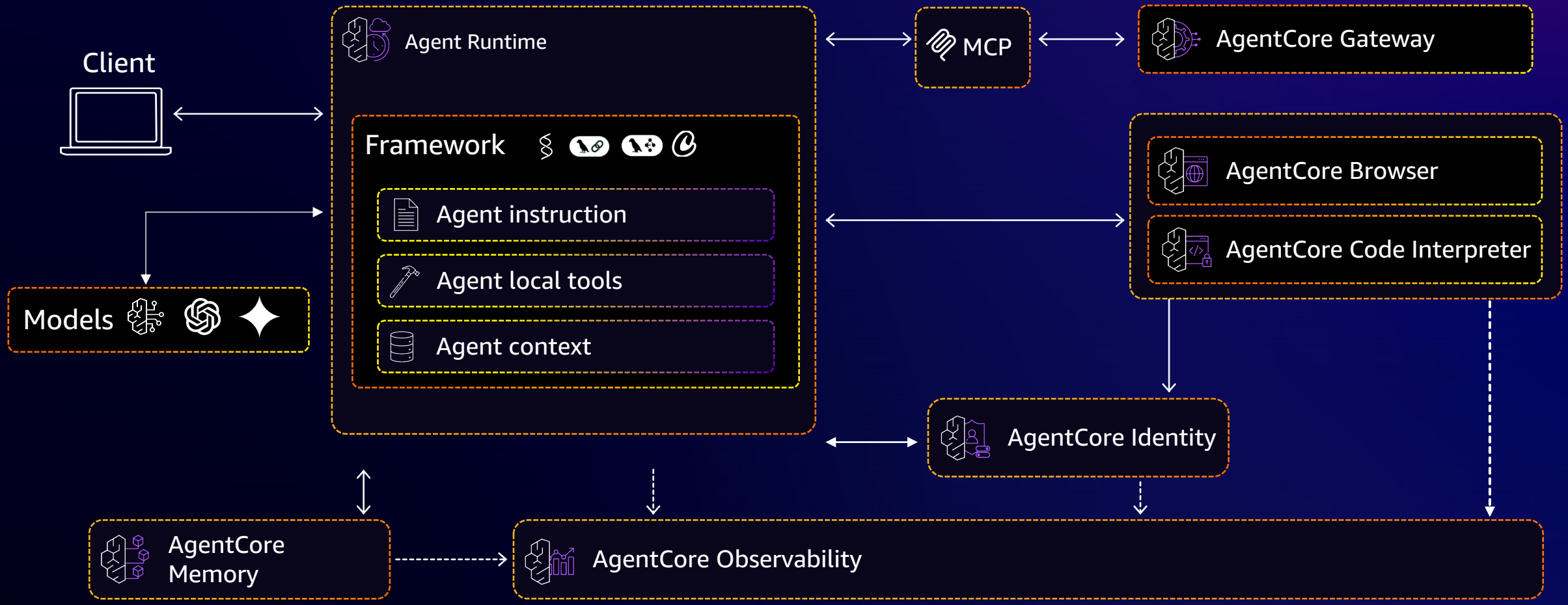
BROWSER ACTIONS

Agents need to navigate websites to perform actions

EXECUTE CODE

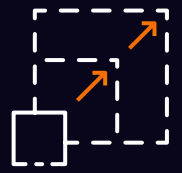
Agents need to execute code and run applications on user's behalf

AgentCore capabilities enabling agents at scale



AgentCore Runtime

Scale from real-time to multi-hour workloads



- Scale from real-time to multi-hour workloads with low latency and industry-leading extended runtime (up to 8 hours)
- Supports payloads across modalities

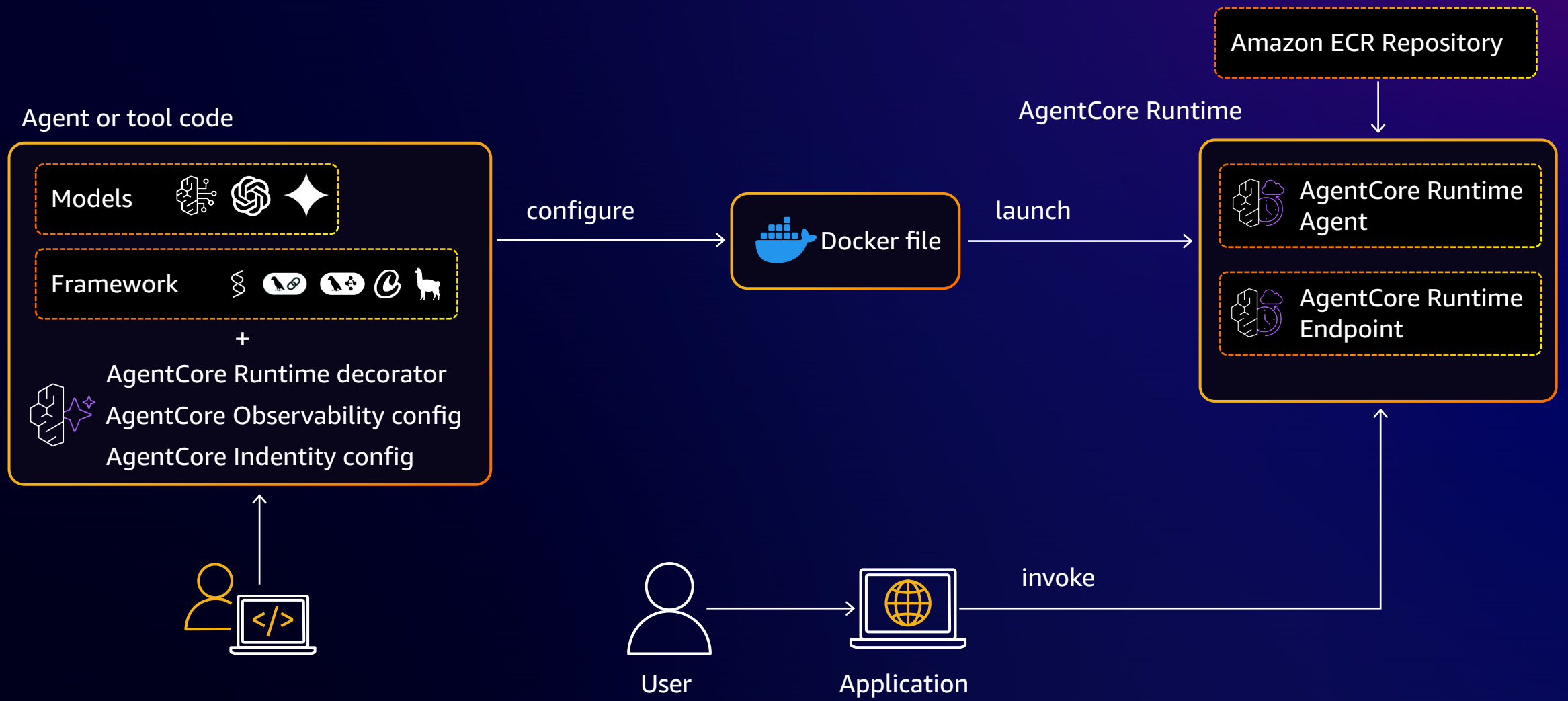
Accelerate time to market

- Deploy any AI agent using common open-source frameworks
- Deploy from POC to production in a few lines of code

Secure workload with enterprise-grade isolation

- True session isolation to protect your data
- Integrates with existing identity providers

AgentCore Runtime



AgentCore Gateway

Simplified tool development & integration

- Turn APIs, Lambda functions, and existing services into MCP-compatible tools
- Access thousands of tools through a standardized interface

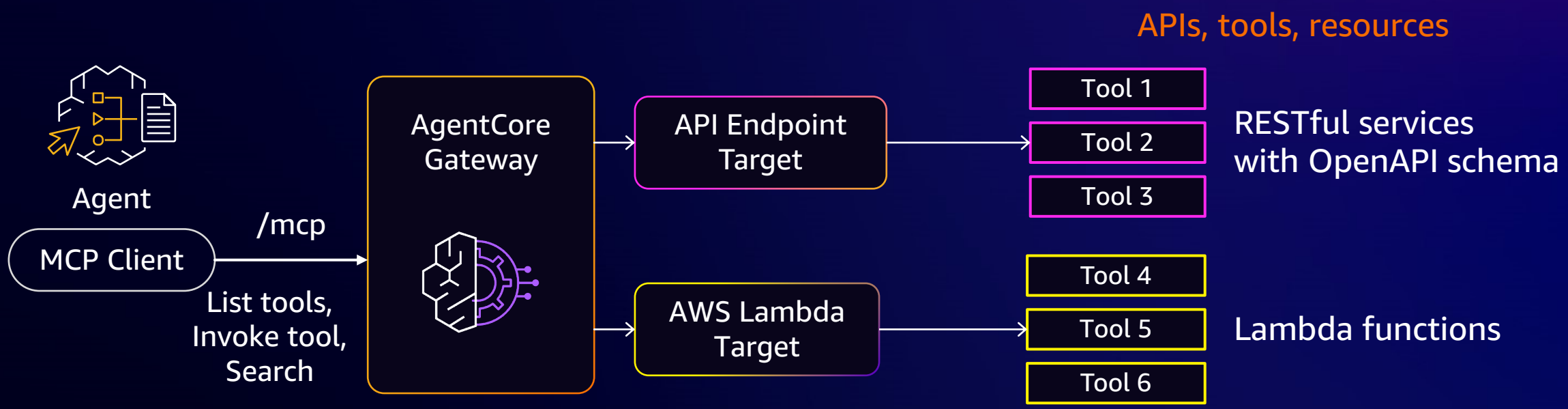
Secure and unified access

- Discover and use tools through a single, secure endpoint
- Combine multiple tools sources into one unified interface

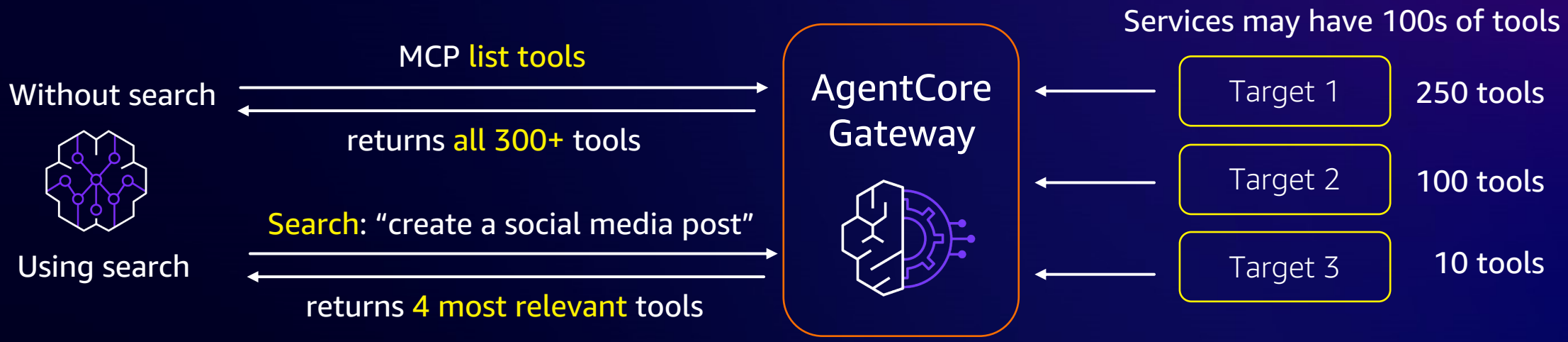
Intelligent tool discovery

- Enable agents to find the right tools with context aware discovery
- Curated tool collections with granular permissions

AgentCore Gateway



AgentCore Gateway semantic search



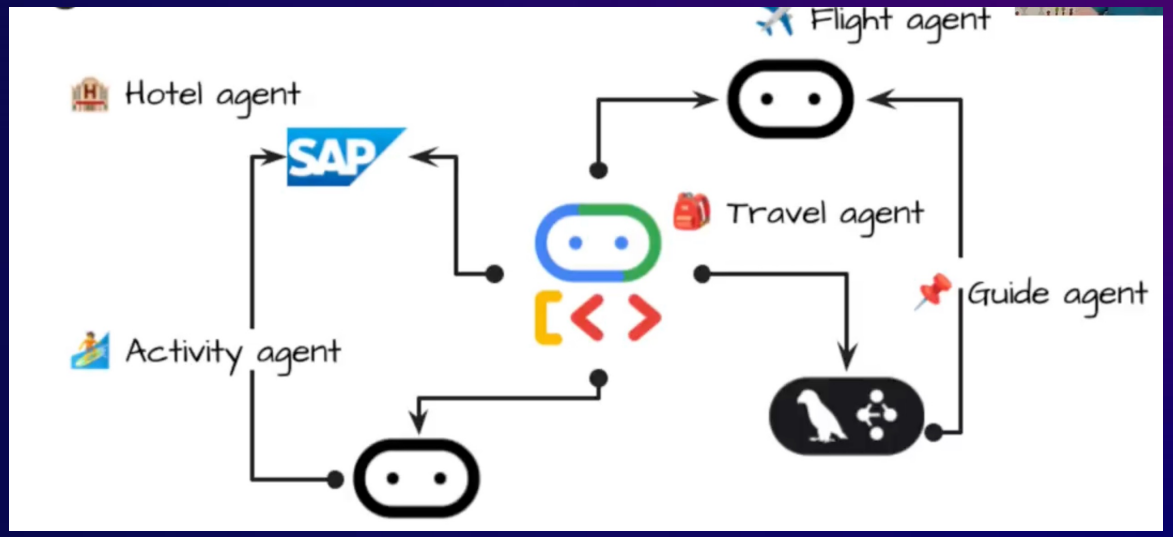
Benefits

- AgentCore Gateway automatically indexes tools, and gives serverless semantic search
- Reduces context passed to the agent’s LLM, improving accuracy, speed, and cost
- Lets agent focus on tools relevant for a given task

A2A

A2A Protocol

A2A is an open protocol that provides a standard way for agents to collaborate with each other, regardless of the underlying framework or vendor.



Key Design Principles

- **Embrace agentic capabilities**: Collaborate in natural, unstructured modalities, even when agents don't share memory, tools and context.
- **Build on existing standards:** Easy integration with existing IT stacks businesses use as the protocol is built on top of existing, popular standards including HTTP, SSE, JSON-RPC
- **Secure by default**: Supports enterprise-grade authentication and authorization, with parity to OpenAPI's authentication schemes at launch.
- **Support for long-running tasks:** Support for long-running tasks like deep research that may take hours and or even days when humans are in the loop. A2A can provide real-time feedback, notifications, and state updates to its users.
- **Modality agnostic:** Support for audio and video streaming

A2A Protocol

HOW IT WORKS

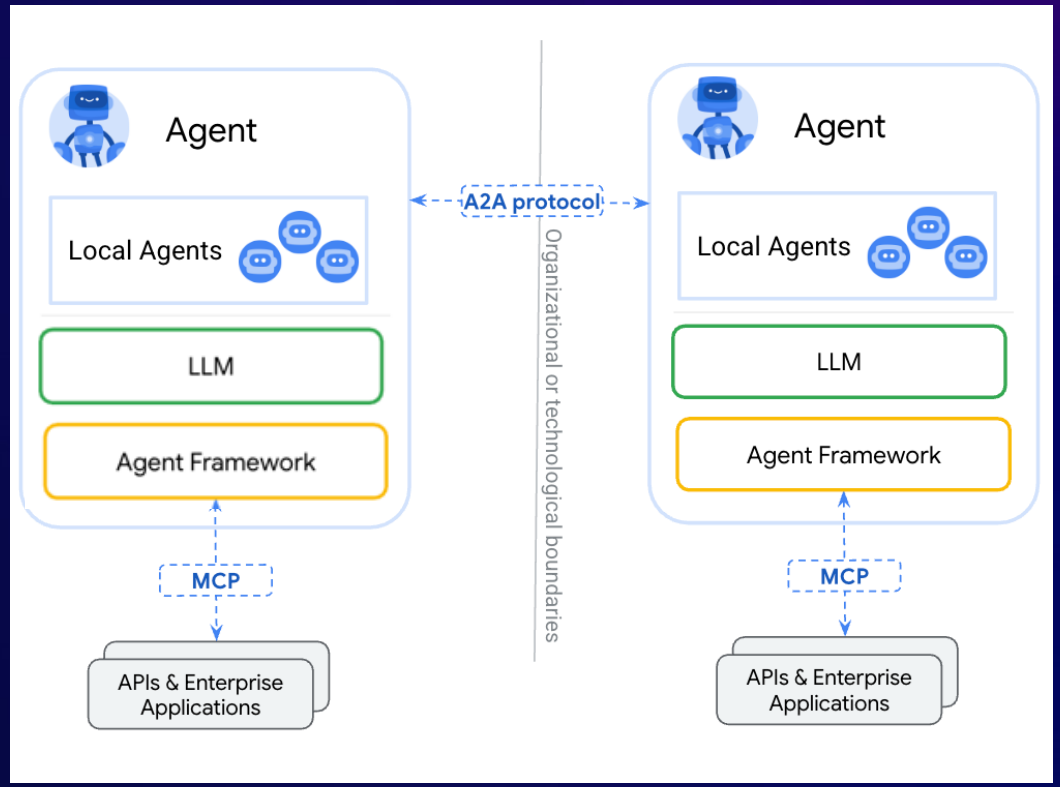
A2A protocol follows a client-server paradigm

Discovery: Agents can advertise their capabilities using an “Agent Card” in JSON format, allowing the client agent to identify the best agent that can perform a task and leverage A2A to communicate with the remote agent.

Task management: Client agent is responsible for formulating and communicating “**tasks”**, while the remote agent is responsible for acting on those tasks.

Collaboration: Agents can send each other messages to communicate context, replies, artifacts, or user instructions (long running tasks).

User Ex Negotiation: Each message includes “parts,” which is a fully formed piece of content, like a generated image. Each part has a specified content type, allowing client and remote agents to negotiate the correct format needed and explicitly include negotiations of the user’s UI capabilities–e.g., iframes, video, web forms, and more.



AgentCore Memory

Simplify memory management

- Abstracts memory infrastructure
- Scales automatically with serverless architecture
- Automatically stores and manages agent context across sessions

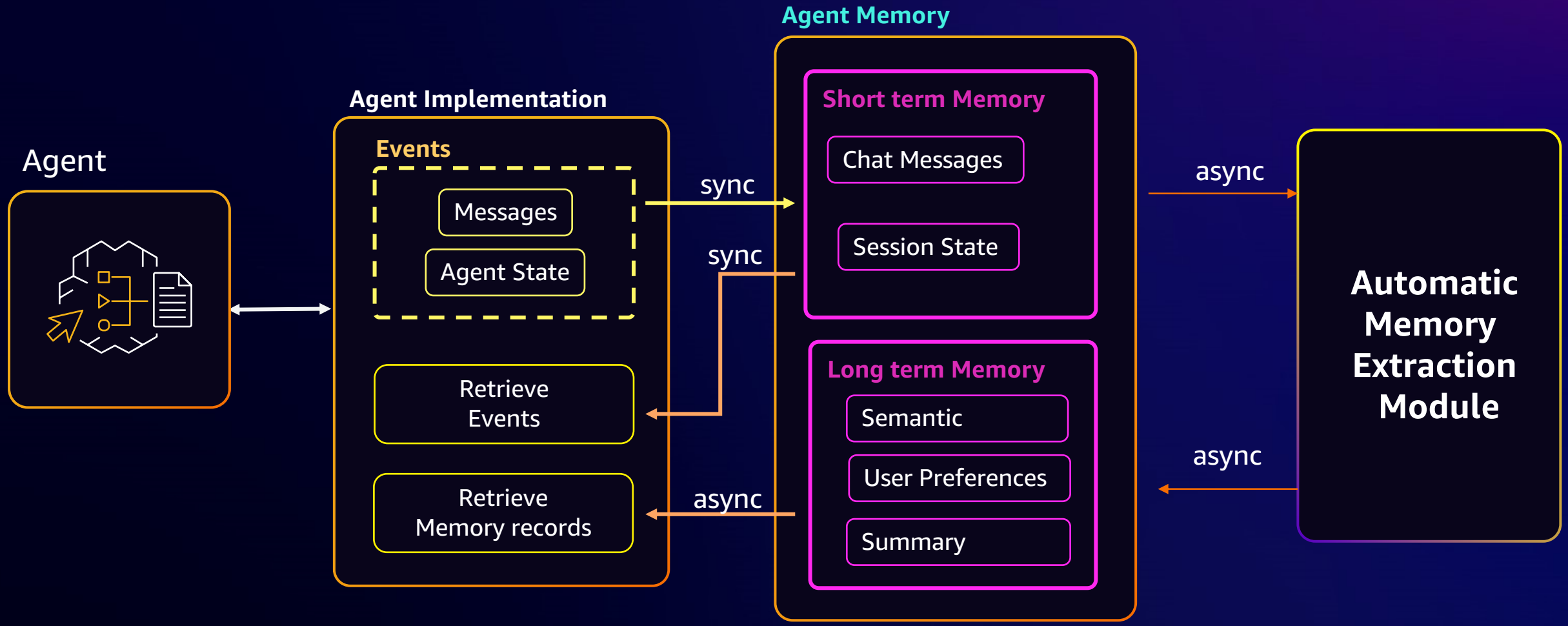
Enterprise-grade

- Complete data privacy with dedicated storage for each customer
- Enterprise security with encryption and regional data storage options

Deep customization

- Define memory patterns based on your use case
- Configure extraction rules
- Choose models and customize prompts for memory extraction

AgentCore Memory



AgentCore Browser

Serverless browser infrastructure

- Low latency browser sessions
- Auto-scales from 0 to hundreds of concurrent sessions

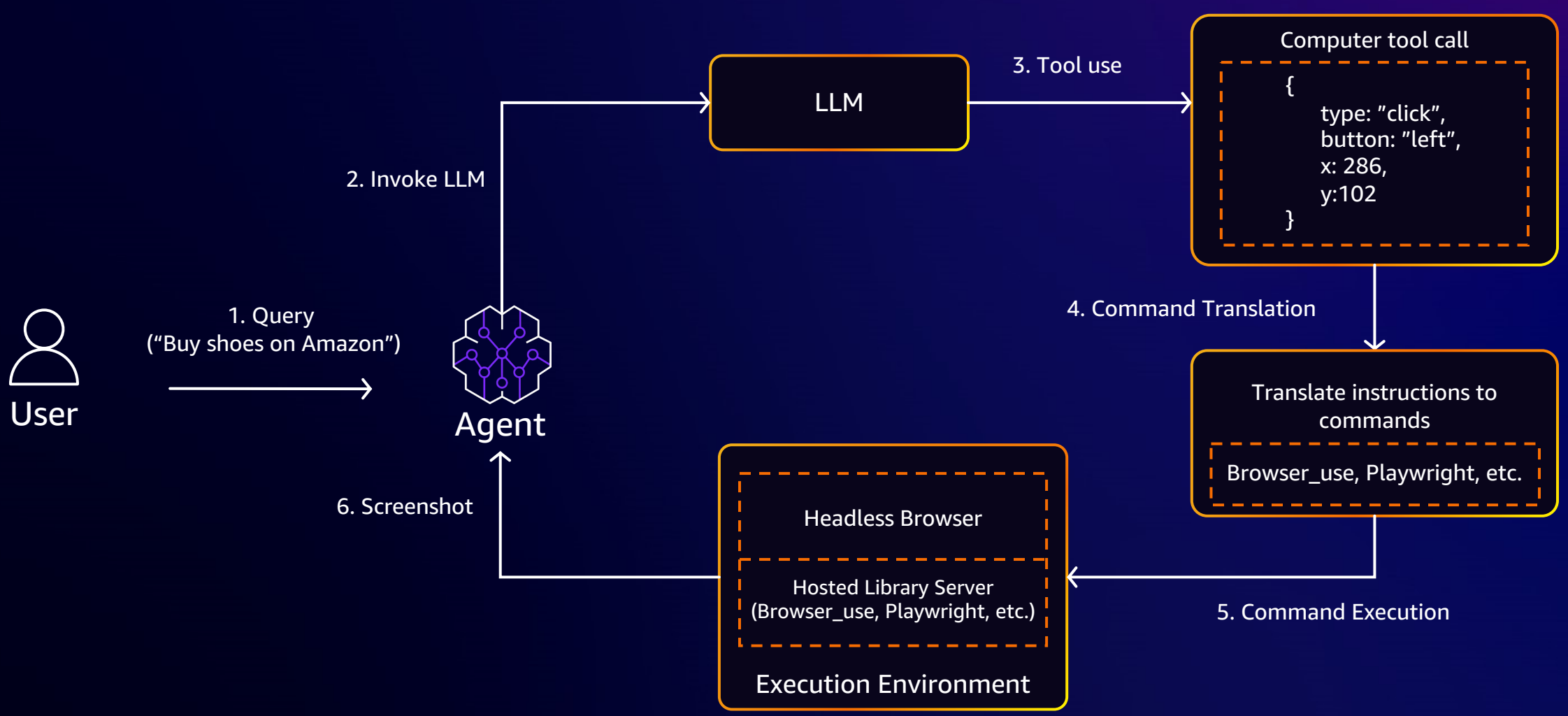
Enterprise-grade security

- Session isolated compute with VM-level isolation per user
- Secure credential handling

Enterprise observability

- Live streaming URLs for real-time monitoring
- Session replays for debugging
- Extensive logging of all browser commands to CloudTrail

AgentCore Browser



AgentCore Code Interpreter

Execute code securely

- Execute complex workflows and data analysis in isolated sandbox environments
- Access internal data sources securely without exposing sensitive data

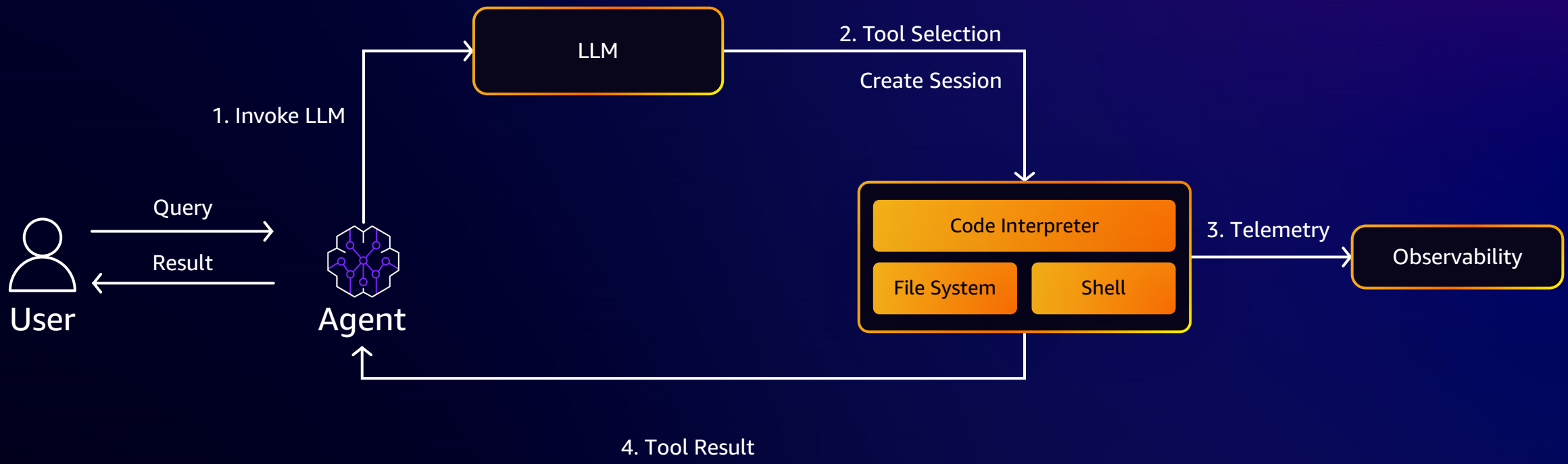
Large-scale data processing

- Process gigabyte-scale datasets efficiently using Amazon S3 integration, without API limitations

Ease of use

- Quick start with pre-built execution runtimes for JavaScript, TypeScript, and Python with common libraries pre-installed

AgentCore Code Interpreter



AgentCore Identity

Secure, delegated access for AI agents

- Enables AI agents to securely access AWS resources and third-party tools such as GitHub, Google, Salesforce and Slack
- Robust access controls with just-enough access and secure permissions delegation

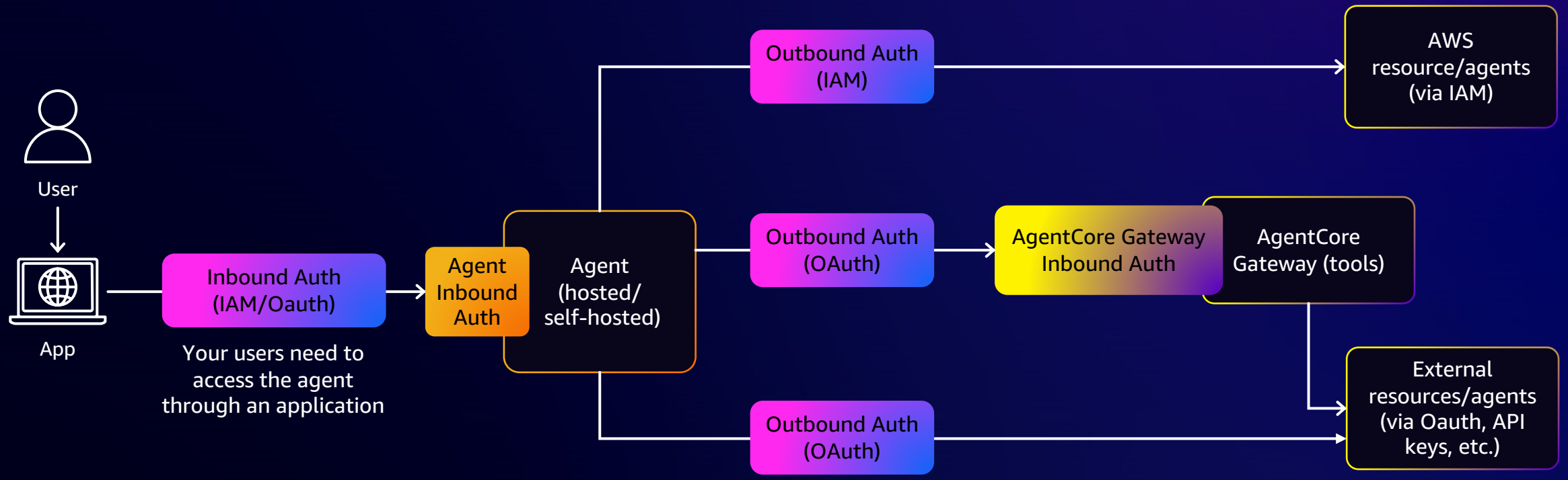
Build streamlined AI agent experiences

- Minimizes consent fatigue with a secure token vault
- Streamlines authentication flows

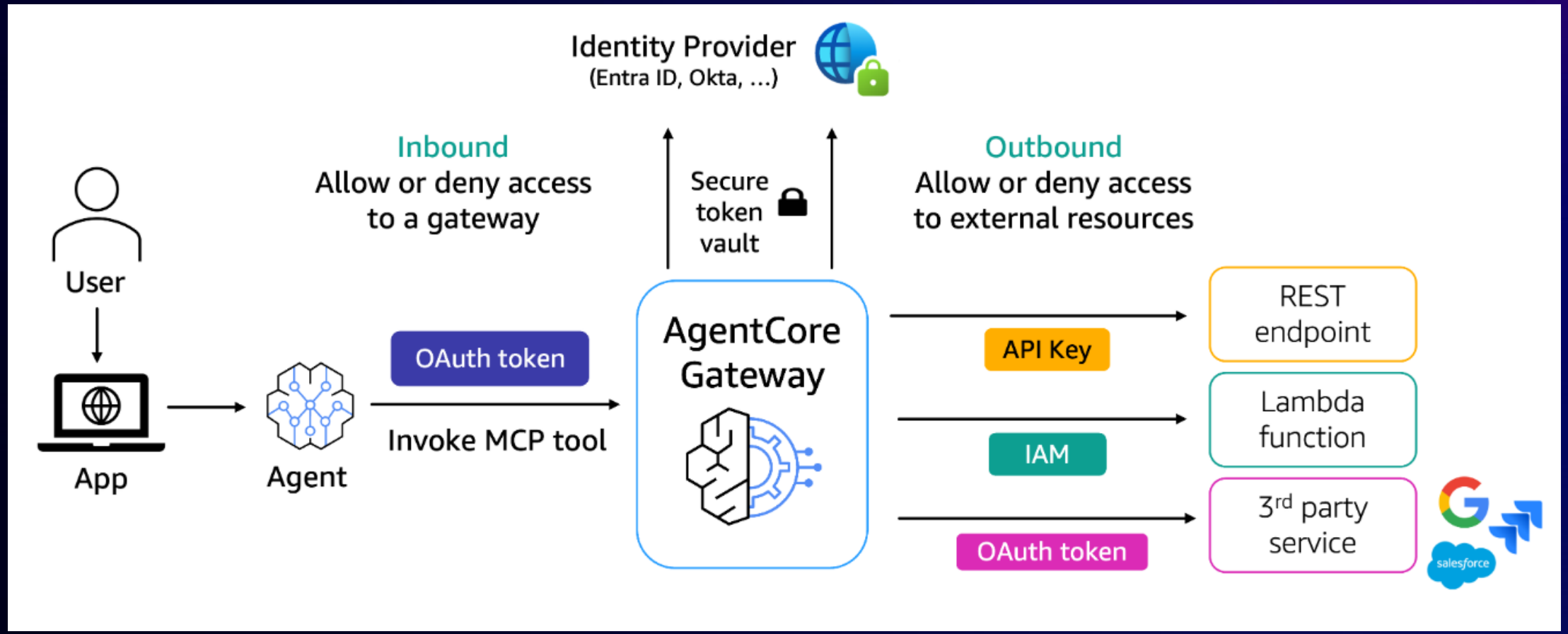
Accelerated AI agent development

- Preserves existing identity systems such as Okta, Azure AD, or Amazon Cognito
- Lowers custom development efforts without need for migrating users or rebuilding authentication flows

AgentCore Identity



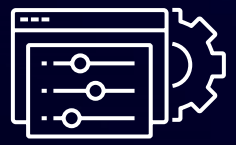
AgentCore Gateway & Identity



PREVIEW

Policy in AgentCore: Complete control over agent actions

Control over agent actions



- Intercept every tool call before execution to ensure adherence to defined policy.
- Control agent access, actions across APIs, Lambda functions, and third-party services.

Frictionless policy enforcement

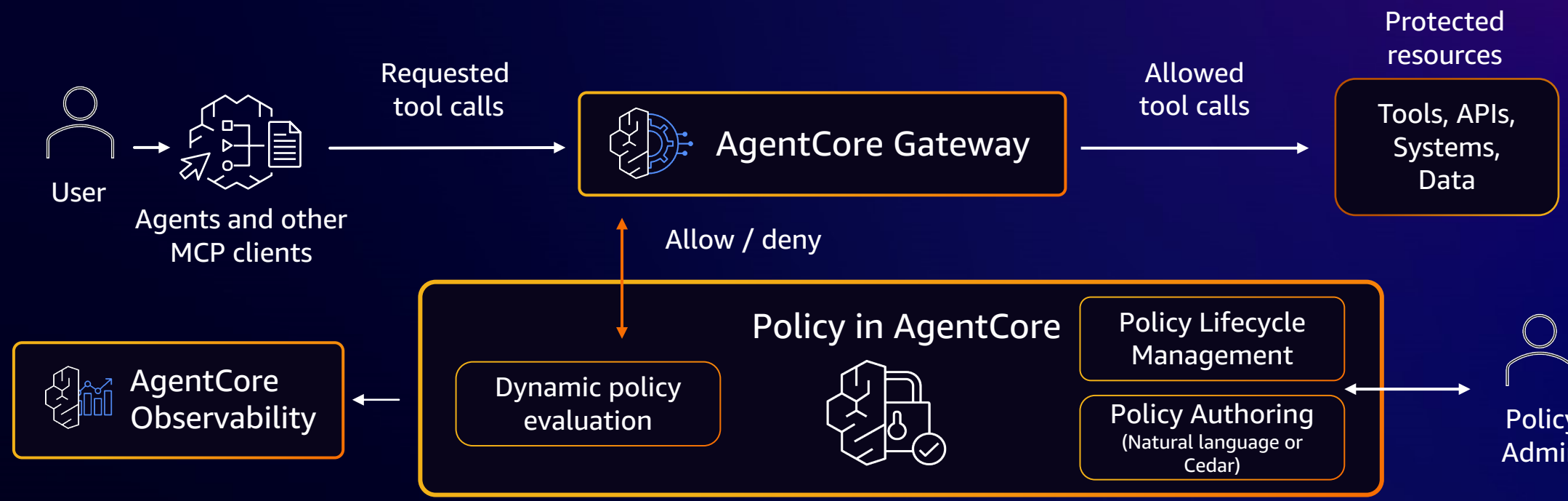
- Process thousands of requests per second with millisecond policy evaluation.
- Support granular access controls with secure VPC and PrivateLink integration.

Easy policy creation and management

- Create and manage policies using natural language with automatic Cedar conversion.
- Deploy consistent policies across your organization with centralized control.

Policy in AgentCore

PREVIEW



Keep agents in bounds

Act autonomously, but stay within boundaries and compliance

Instant and consistent

Policies evaluated in milliseconds, without slowing agents

Verifiably correct

Built on years of automated reasoning, using Cedar

AgentCore Hands-on

Observability
Why it matters?

1. Agents follow non-deterministic control flows powered by model's reasoning
2. Root cause analysis in complex sequence of chained calls
3. Assessing system health holistically for operational performance
4. Detecting performance degradation

Do I need both?

Short answer: YES!

Observability: What is happening? – Reactive

Evaluation: How well is it happening? - Proactive

Pillars of Observability

Logs

“The input prompt and generated outputs by the user were…”

Metrics

“The prompt to response latency has increased by 35%.”

Traces

“What is the distribution of processing time across a RAG architecture?”

A layered approach to instrumenting observability

Basic observability

Layer 1: Component-level metrics (e.g. invocations errors, latency, resource utilization)

Layer 2: RAG, agent, or chain traces

Layer 3: Advanced metrics & analysis (e.g. guardrails interventions or embedding drift)

Layer 4: End-user feedback (e.g. thumbs up/down, text feedback)

Deeper observability

Best practices to consider for this layers

1/ Multi-system tracing: across multi-agents, MCPs and tools

2/ Session-level tracking: monitor entire interactions and task sequences

3/ Path analysis(Obs+Eval): using tracing and step-by-step execution sequences

+Continous Improvement

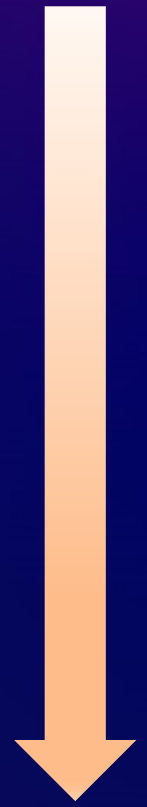
A layered approach to instrumenting observability

Layer 1: Component-level metrics (e.g. invocations errors, latency, resource utilization)

Layer 2: RAG, agent, or chain traces

Layer 3: Advanced metrics & analysis (e.g. guardrails interventions or embedding drift)

Layer 4: End-user feedback (e.g. thumbs up/down, text feedback)



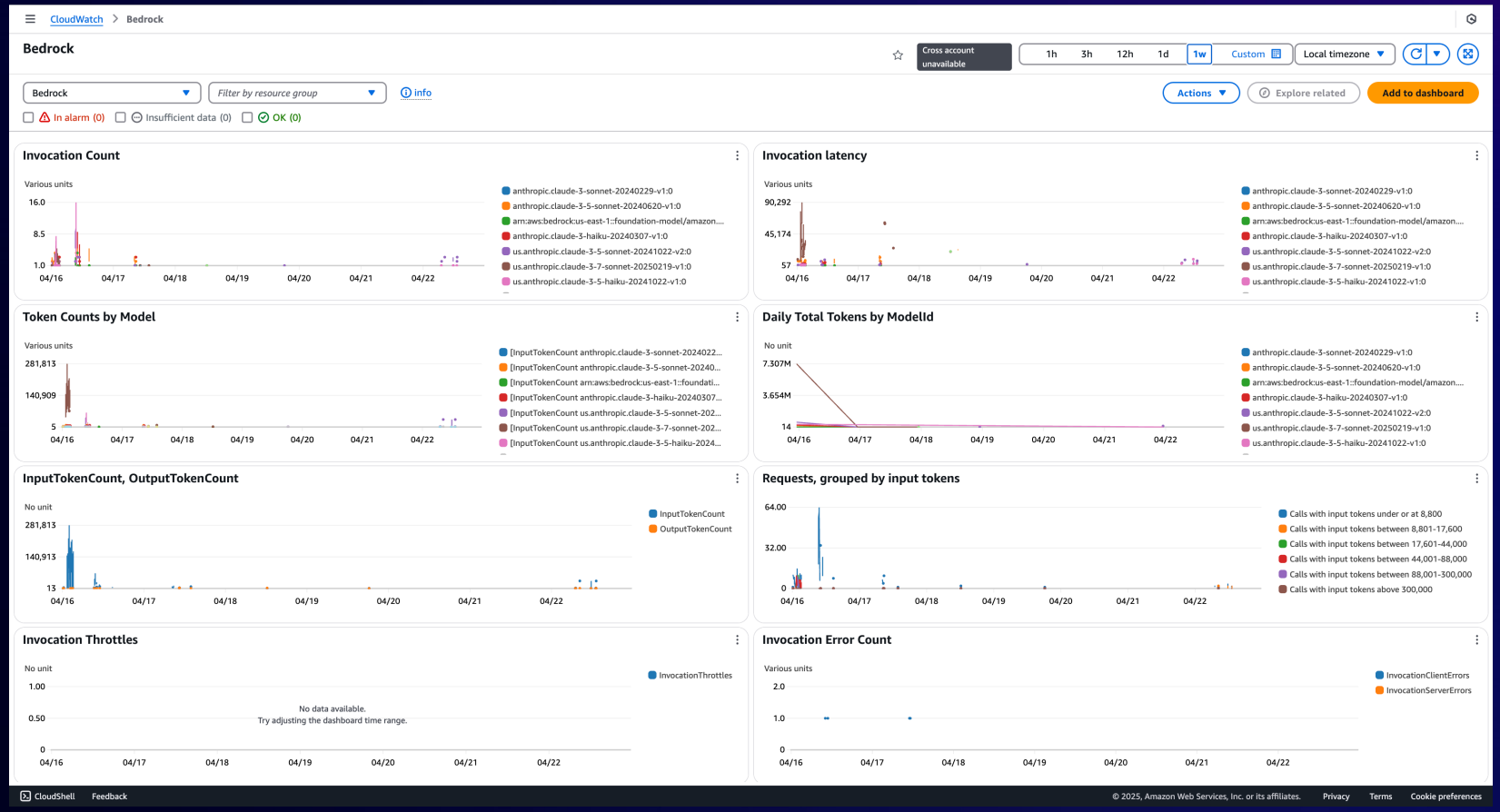
Observability features in Amazon Bedrock

BUILDING BLOCKS FOR SCALABLE MONITORING

Metrics

Logs

Traces



AgentCore Observability

Maintain quality and trust

- Comprehensive end-to-end visibility into agent behavior
- Accelerated debugging and quality audits
- Quickly detect issues and assess performance trends

Integrate with 3P observability tools

- Integration with a wide range of monitoring and observability tools, including CloudWatch
- Flexibility to leverage your existing observability stack

AgentCore Observability

Agent deployed on AgentCore Runtime

Agent memory using AgentCore Memory

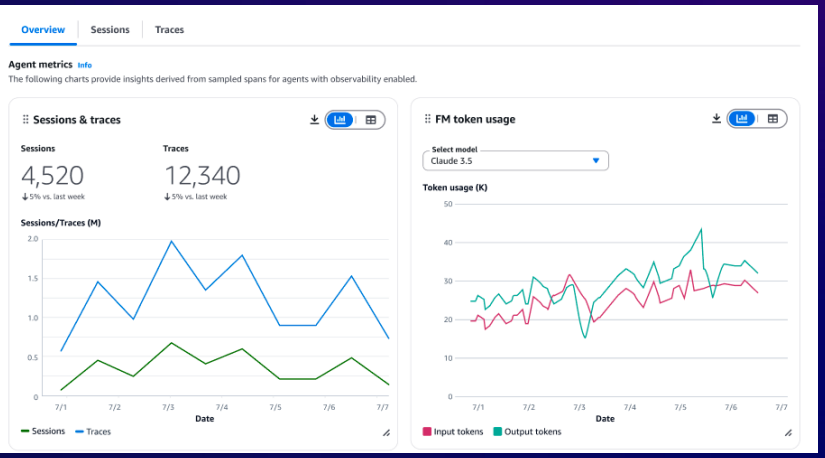
Tools from AgentCore Gateway

...

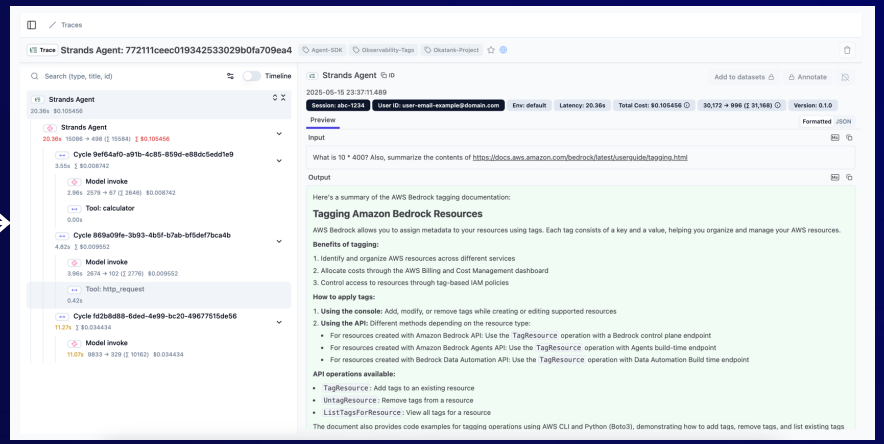
Amazon Bedrock AgentCore Observability

OTEL logs

AgentCore Observability dashboards

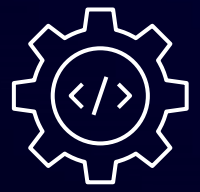


Third-party observability dashboards



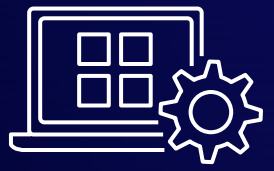
AgentCore Evaluations: Improve agent quality based on real-world performance

Real-time quality intelligence



- Sample and score live interactions with 13 built-in evaluators
- Gain actionable insights from real-world user behavior to continuously improve agent performance.
- On-demand and online evaluations

Custom business scoring



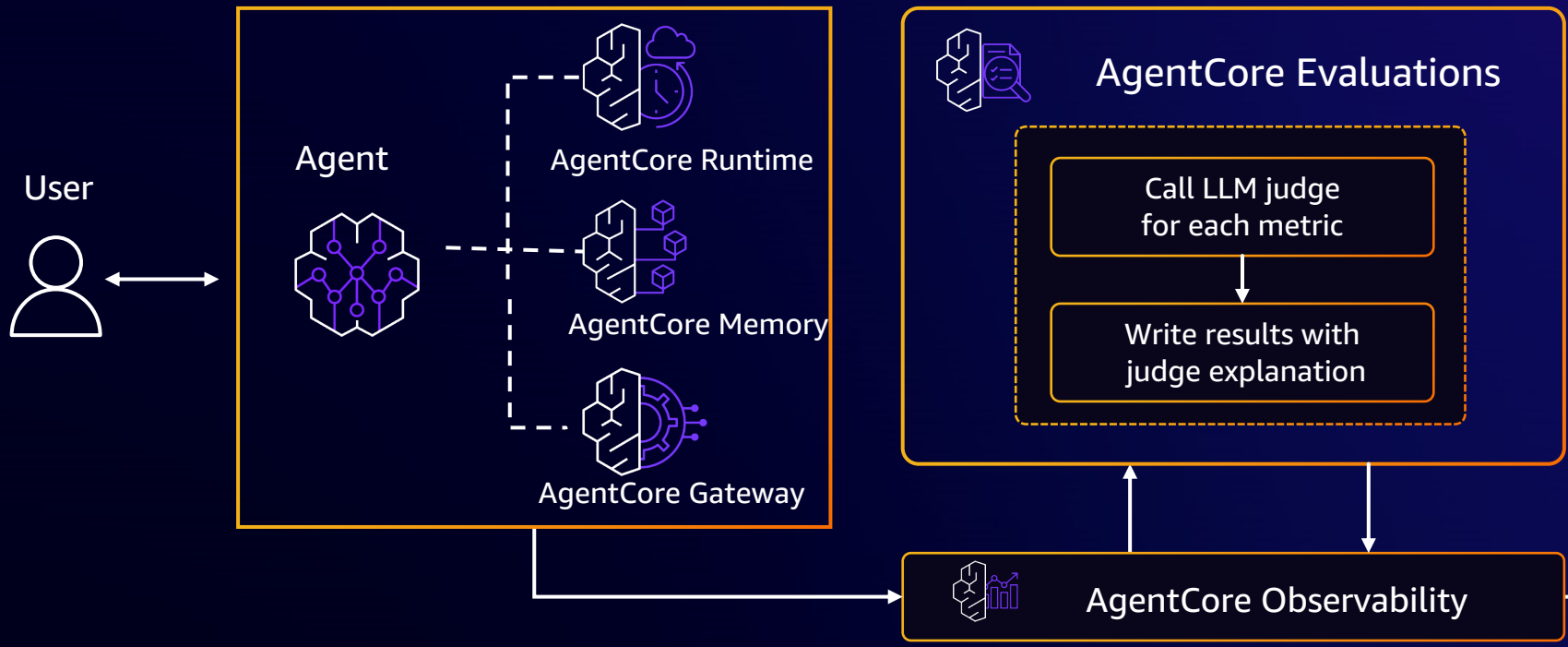
- Build tailored quality assessments for specific use cases using custom evaluators.
- See various levels of granularity (entire trajectory, final response, individual step).

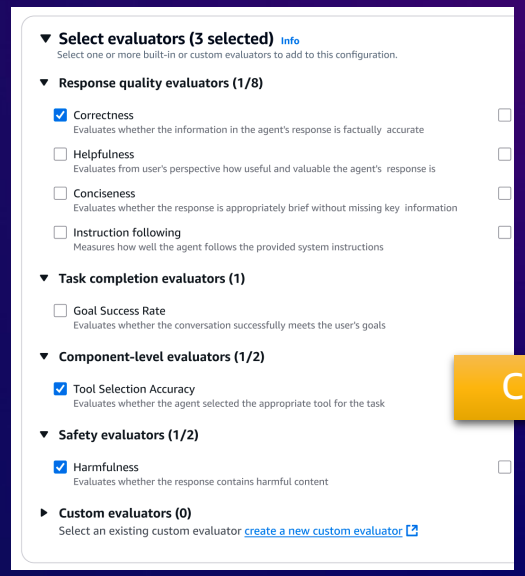
No infrastructure overhead

- Out-of-the-box evaluation from logs available in Amazon CloudWatch.
- Eliminate months of effort required to build infrastructure and manage operational complexities.

AgentCore Evaluations

PREVIEW





Configure

On-demand

Agent owner

Monitor and assess

Two evaluation modes for different needs

Agent's evaluation metrics depends on the dimension we want to optimize for

Online Evaluation

- Monitor live agent interactions
- Sample traces continuously
- Detect quality degradation

Use cases

- Catch silent failures
- Track quality trends
- Monitor after deployments
- Ensure ongoing reliability

Production Monitoring

On-Demand Evaluation

- Test changes before deployment
- Run evaluation tests in CI/CD
- Regression testing for builds
- Gate deployments on quality

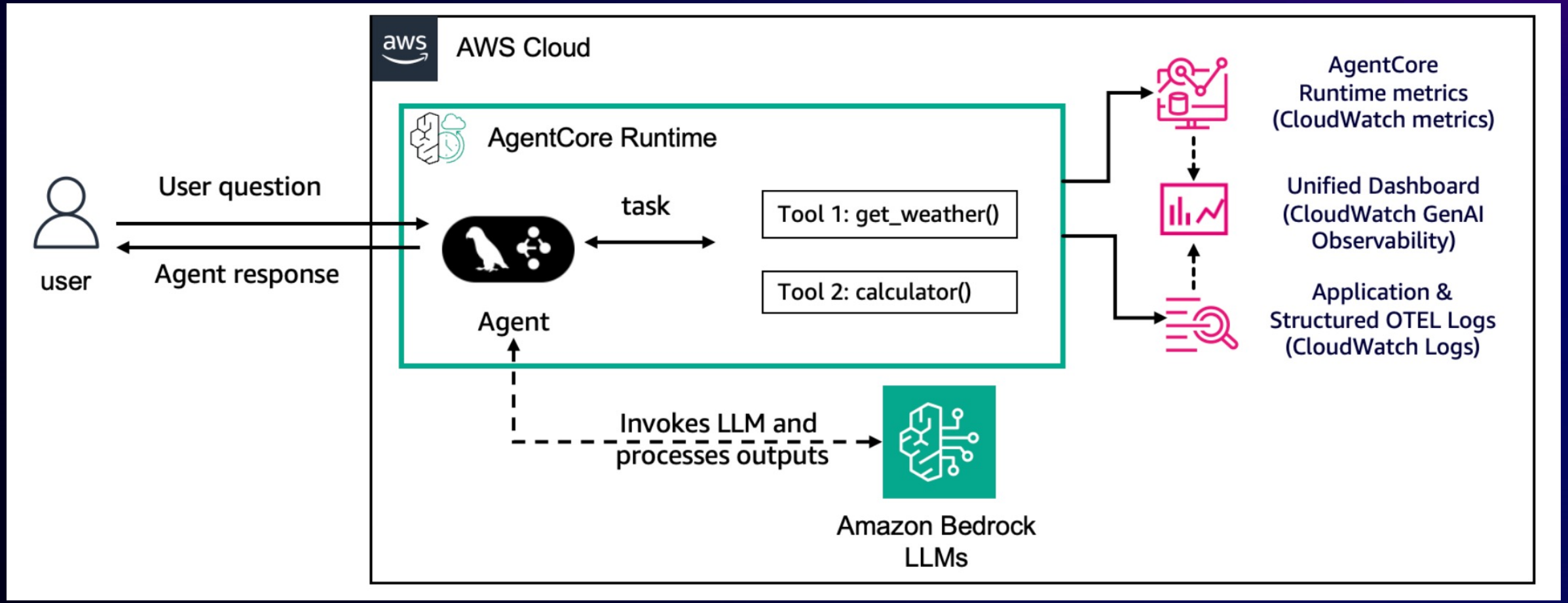
Use cases

- Validate prompt changes
- Compare model performance
- Prevent quality regressions
- Automate quality gates

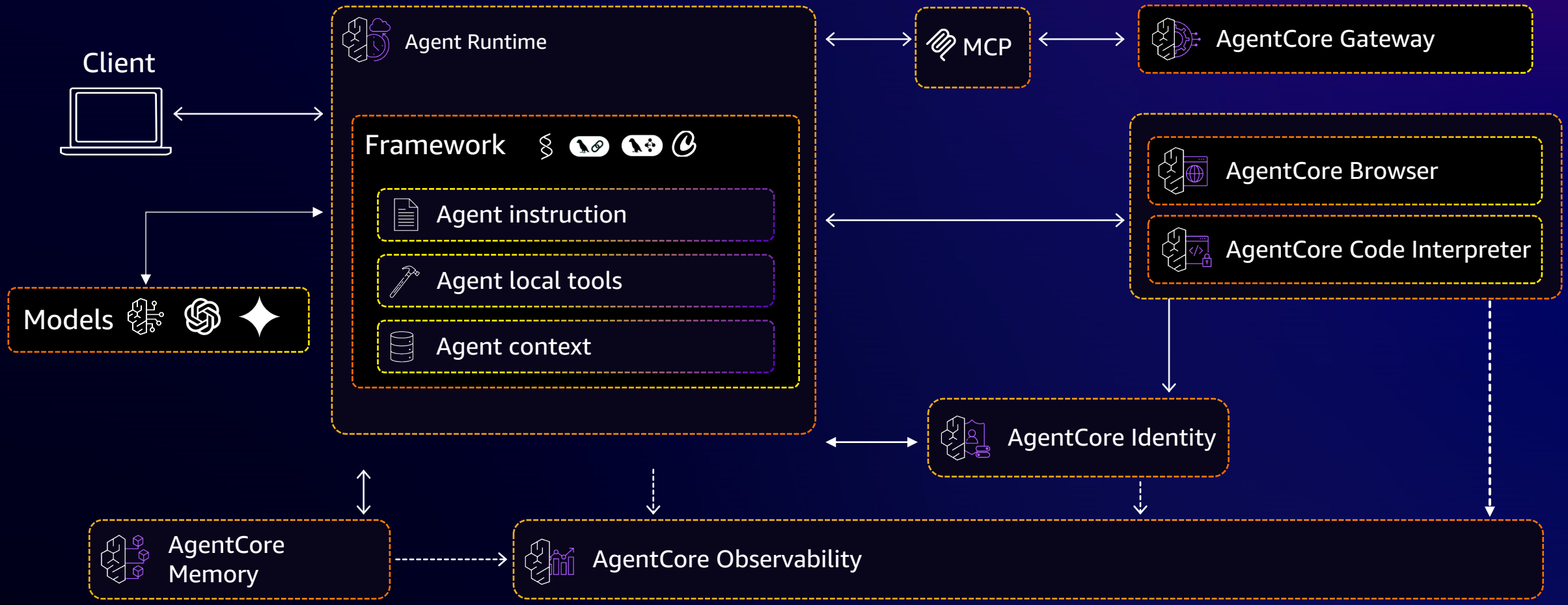
Development and CI/CD

Architectures Diagram

AgentCore Observability with agents deployed on AgentCore Runtime



AgentCore capabilities enabling agents at scale



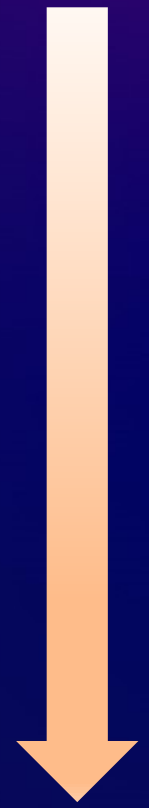
A layered approach to instrumenting observability

Layer 1: Component-level metrics (e.g. invocations errors, latency, resource utilization)

Layer 2: RAG, agent, or chain traces

Layer 3: Advanced metrics & analysis (e.g. guardrails interventions or embedding drift)

Layer 4: End-user feedback (e.g. thumbs up/down, text feedback)



Layer 3: Advanced metrics and analysis

The following table des

Runtime metrics

Metric name
Invocations
InvocationLatency
InvocationClientErro
InvocationServerErro
InvocationThrottles
TextUnitCount
InvocationsIntervene

Dimension name	Dimension values	Available for the following metrics
Operation	ApplyGuardrail	• Invocations • InvocationLatency • InvocationClientErrors • InvocationServerErrors • InvocationThrottles • InvocationsIntervened • TextUnitCount
GuardrailContentSource	• Input • Output	• Invocations • InvocationLatency • InvocationClientErrors • InvocationServerErrors • InvocationThrottles • InvocationsIntervened • TextUnitCount
GuardrailPolicyType	• ContentPolicy • TopicPolicy • WordPolicy • SensitiveInformationPolicy • ContextualGroundingPolicy	• InvocationsIntervened • TextUnitCount
GuardrailArn, GuardrailVersion	• Guardrail Arn • Guardrail Version number or DRAFT	• Invocations • InvocationLatency • InvocationClientErrors • InvocationServerErrors • InvocationThrottles • InvocationsIntervened • TextUnitCount

CloudWatch Metrics.

operation
e errors
er-side errors
ed
ails policies
tervened

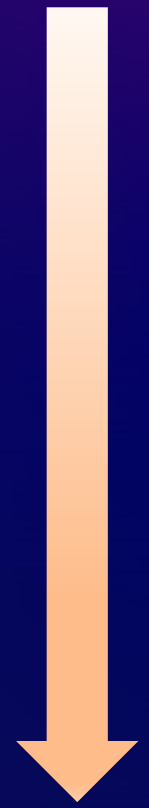
A layered approach to instrumenting observability

Layer 1: Component-level metrics (e.g. invocations errors, latency, resource utilization)

Layer 2: RAG, agent, or chain traces

Layer 3: Advanced metrics & analysis (e.g. guardrails interventions or embedding drift)

Layer 4: End-user feedback (e.g. thumbs up/down, text feedback)

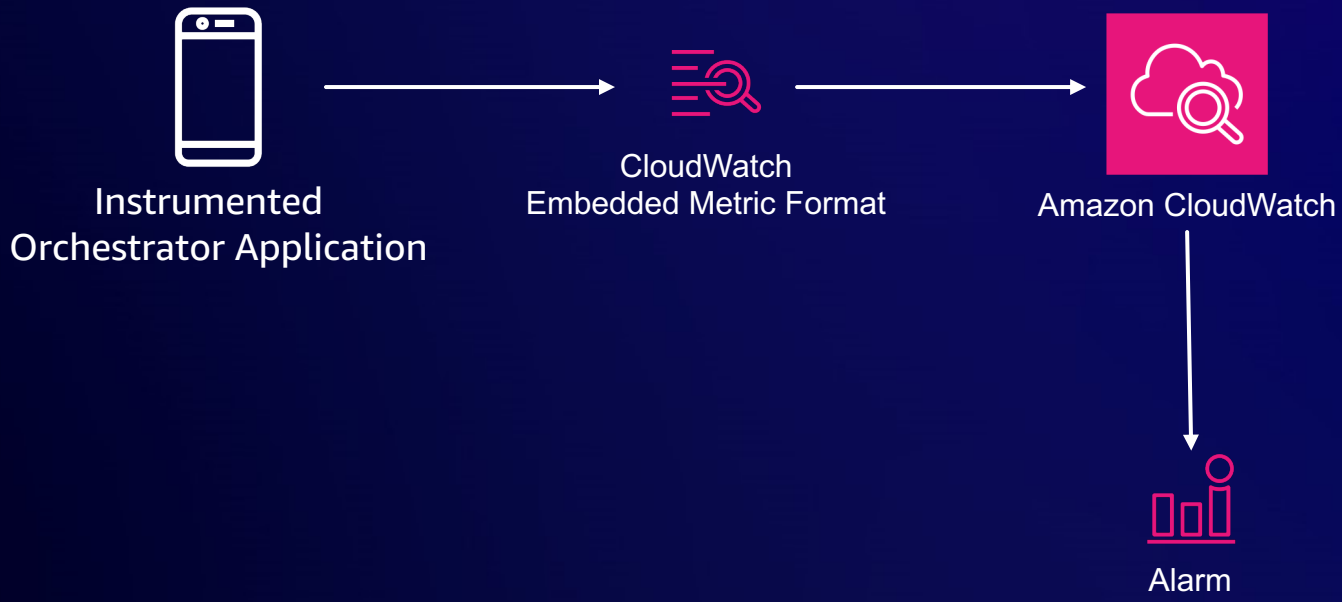


Layer 4: End-user feedback

PROVIDE EARLY INDICATIONS OF POTENTIAL ISSUES

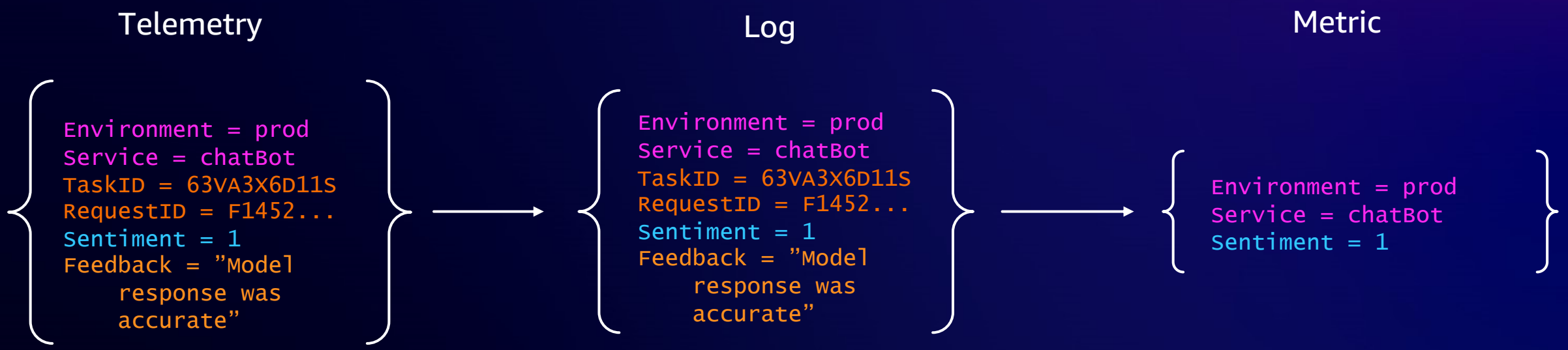
We can leverage CloudWatch Embedded Metric Format (EMF) to instruct CloudWatch Logs to automatically extract metrics values.

This allows us to correlate a thumbs-up/thumbs-down response with written feedback and optionally alarm on negative user experience



CloudWatch Embedded Metric Format (EMF)

CORRELATE YOUR LOGS AND METRICS



High cardinality
Dimensions
Metrics
Properties

CloudWatch Embedded Metric Format (EMF)

```python
def submit_feedback():

    customMetricJson = {
        "_aws": {
          "Timestamp": int(time.time())*1000,
          "CloudWatchMetrics": [
            {
              "Namespace": "ChatBot-Sentiment",
              "Dimensions": [["modelId"]],
              "Metrics": [
                {
                  "Name": "sentinment",
                  "Unit": "Count",
                  "StorageResolution": 60
                }
              ]
            }
          ]
        },
        "modelId": model_id,
        "sentinment": thumbFeedback,
        "feedback": txtFeedback
    }
```

```python
cwLogEvent = json.dumps(customMetricJson)

    logsClient = boto3.client('logs')
    response = logsClient.put_log_events(
        logGroupName='/chatbot/feedback',
        logStreamName='default',
        logEvents=[
            {
                'timestamp': int(time.time())*1000,
                'message': cwLogEvent
            }
        ]
    )
```

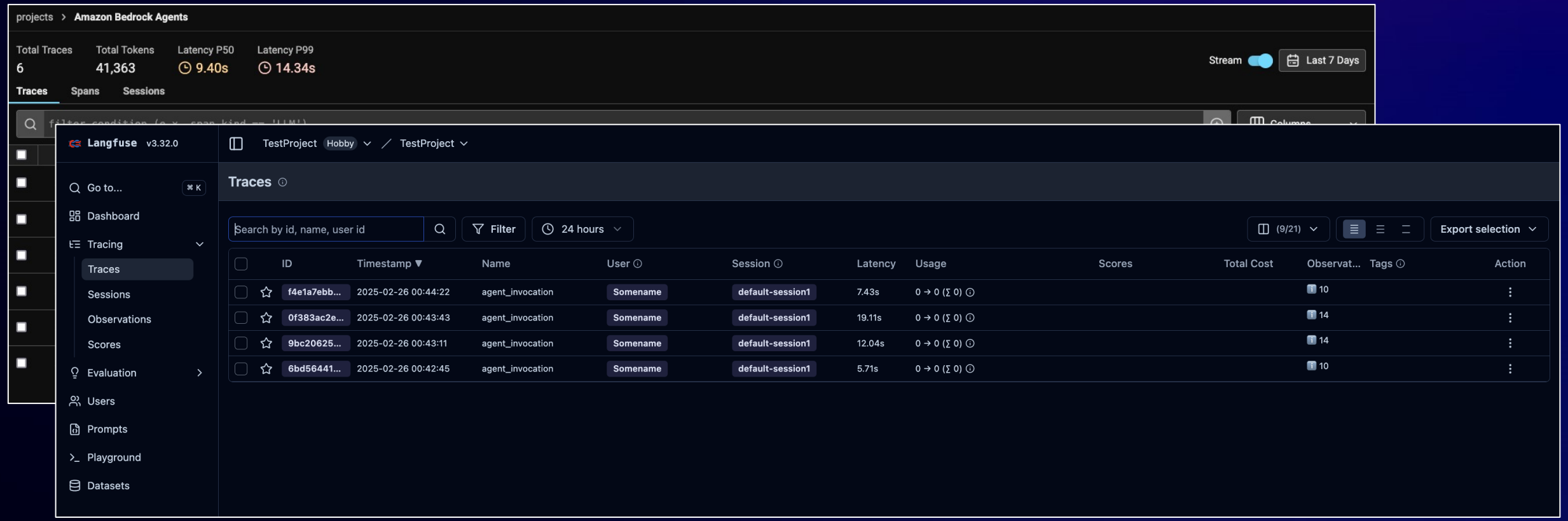
Amazon Bedrock OpenTelemetry Instrumentation

- Decorator implementation
- Can integrate with any observability provider
- 2500+ lines of Python code
- Single click CFN template for Langfuse
- Instrumentation for invoke and Converse API
- Amazon Bedrock Agents
- Amazon Bedrock knowledge Bases

Monitoring Bedrock Agents apps

GETTING VALUE FROM YOUR TRACES AND LOGS USING OPEN-SOURCE SOLUTIONS

Get started now, understanding agent behavior using Phoenix and/or LangFuse:
https://github.com/awslabs/amazon-bedrock-agent-samples/tree/main/examples/agent_observability



Integrations

Discovery Questions for Your Teams

Current State

Are we defaulting to agents when a decision tree would suffice?

Scaling Strategy

What is our path from frontier models to cost-optimized inference at scale?

Engineering Investment

How much engineering effort is allocated to evaluation vs. feature development?

Safety Posture

Do we have deterministic guardrails on both input AND output?

Organizational Readiness

Is there executive ownership of AI safety and trust?

Red Flags and Anti-Patterns

Agent-First Thinking

"We will use an agent for everything." This ignores the decision hierarchy.

Eval-Later Mindset

"We will figure out testing once it is in production." Recipe for rework.

Single-Model Dependency

No plan for cost optimization at scale.

Safety as Afterthought

"We will add guardrails when we need them."

Summary

1. Use AgentCore observability as a starting point

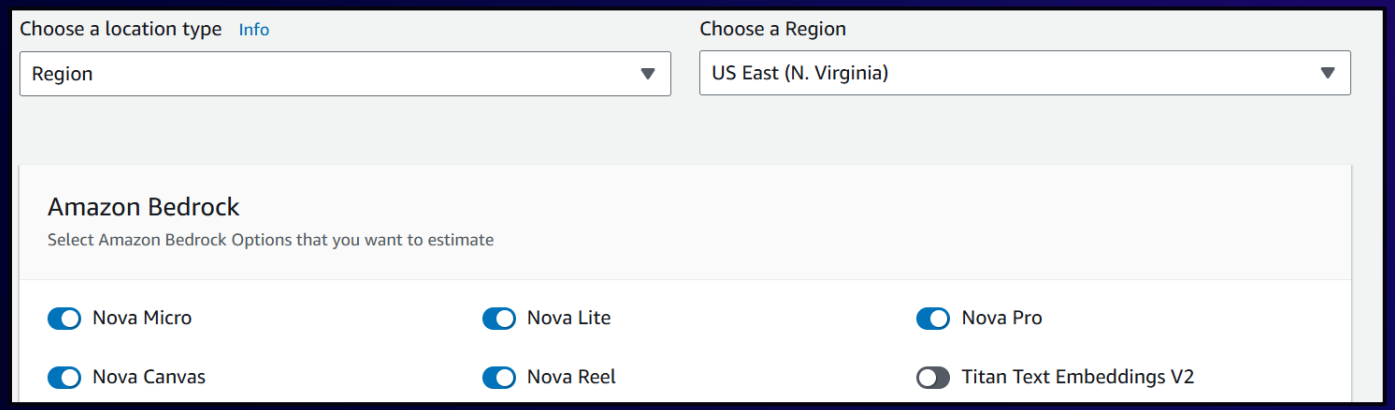
2. Introduce advanced metrics such as guardrail interventions and embedding drift detection

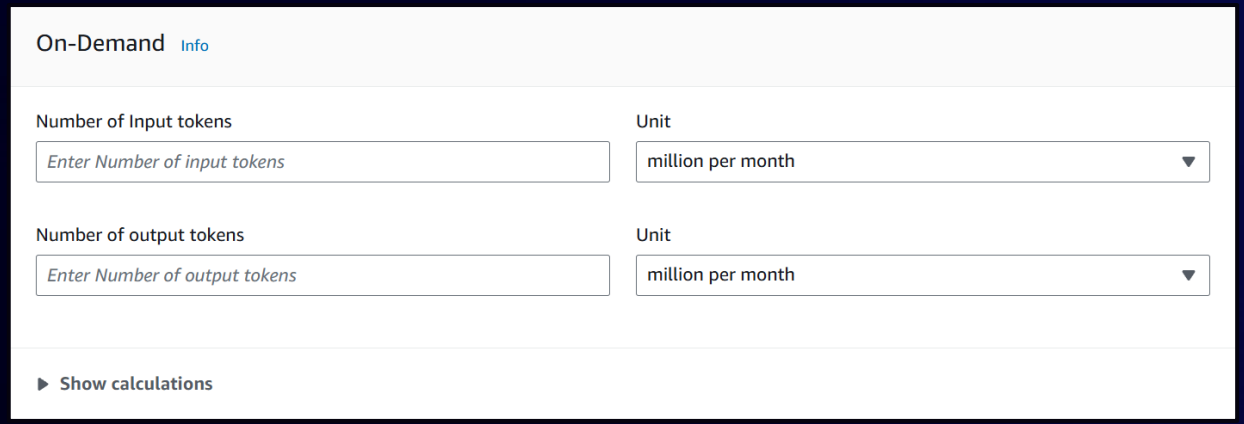
3. Finally, incorporate end user feedback for quality assessment and malfunction alerts, completing the observability loop

AgentCore Hands-on

Cost performance

Leverage the AWS Pricing Calculator





Amazon Bedrock offers several features to balance cost, latency and accuracy

Amazon Bedrock Model Distillation

Utilize smaller, more cost-effective models

Maximize distilled model performance with proprietary data synthesis

Distilled models are up to 500% faster and up to **75% less expensive** than original models, with **less than 2% accuracy loss** for use cases like RAG

Amazon Bedrock Intelligent Prompt Routing

Automatically route prompts to different foundation models to optimize response quality and lower costs

Provides a single endpoint to efficiently route prompts

Uses advanced prompt matching to meet cost and latency thresholds

Reduce costs by **up to 30%** without compromising on accuracy

Amazon Bedrock support for prompt caching

Cache repetitive context in prompts across multiple API calls

Securely cache entire prompts

Enhance accuracy through longer, detailed prompts

Reduce costs by **up to 90**% and latency **by up to 85%** for supported models

Amazon Bedrock inference consumption options

ON DEMAND

Pay-as-you-go, no commitment

- Pricing based on input and output token count for LLMs
- Great for prototyping, POCs, and small workloads with more relaxed requirements for throughput and latency
- Requests per minute (RPM) and tokens per minute (TPM) limits enforced
- Includes support for cross-region inference

PROVISIONED THROUGHPUT

Provision sufficient throughput to meet your application's performance requirements

- Reserve throughput (input/output tokens per minute) at a fixed cost
- Flexible commitment term of 1 month or 6 months
- Hourly PT (2MU), no commit, across different models
- Pay hourly rate, discounted for extended commit
- Great for production workloads or inference on custom models

BATCH INFERENCE

Use Amazon Bedrock to process prompts in batch to get responses for model evaluation, experimentation, and offline processing.

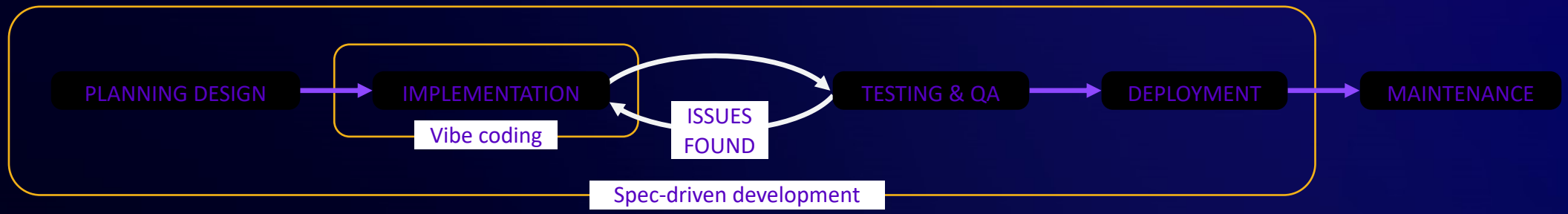
- Amazon Bedrock offers select foundation models (FMs) from leading AI providers like Anthropic, Meta, Mistral AI, and Amazon for batch inference at 50% of on-demand inference pricing
- Efficiently run model inference on large volumes of data
- Avoids throttling when running large jobs

© 2025, Amazon Web Services, Inc. or its affiliates. All rights reserved.

What would a development experience look like if it could take full advantage of working with AI agents?

Solution: Spec-driven development

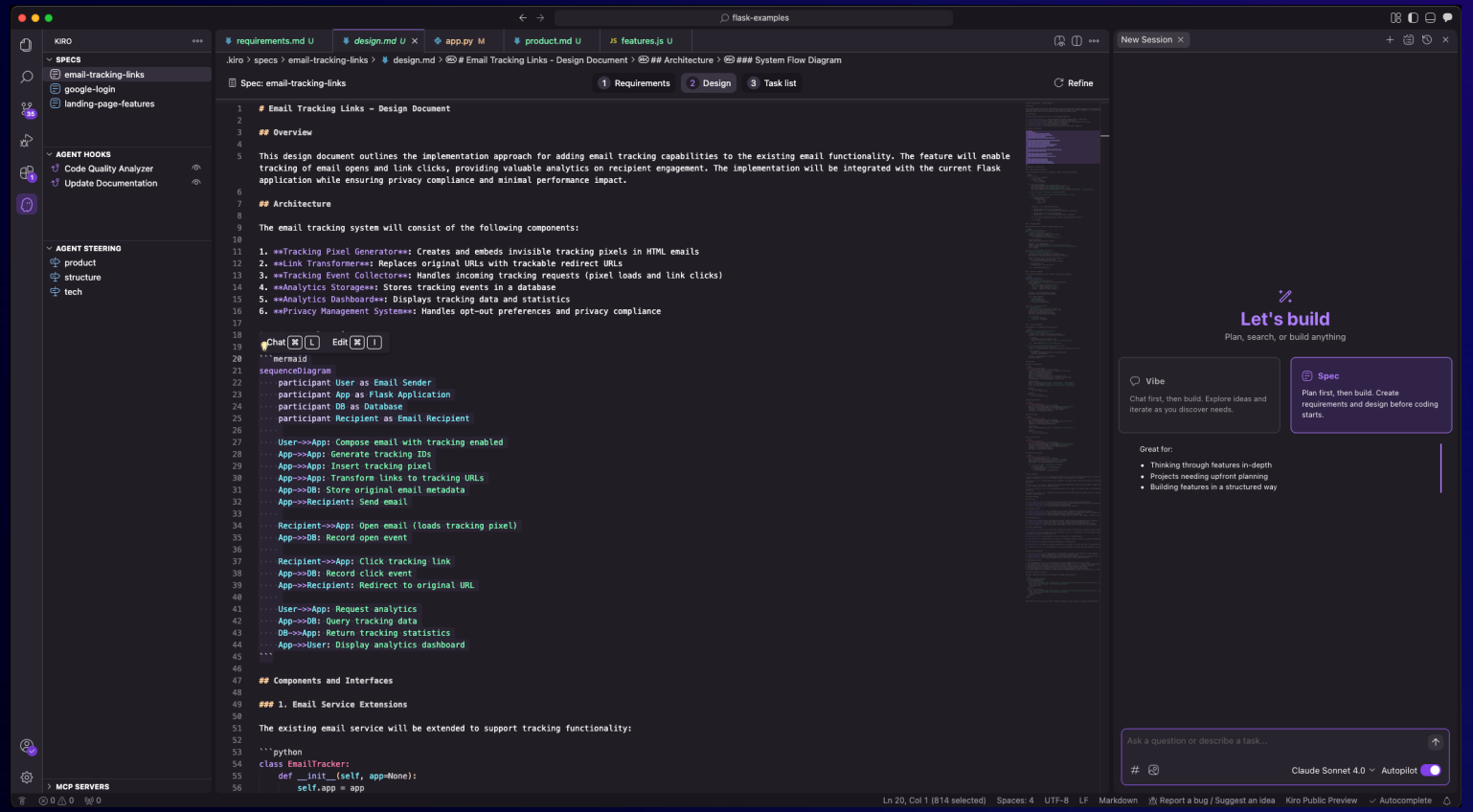
Vibe coding ships the prototype, traditional SDLC practices ship the product



Kiro helps developers and engineering teams
ship high quality software with AI agents

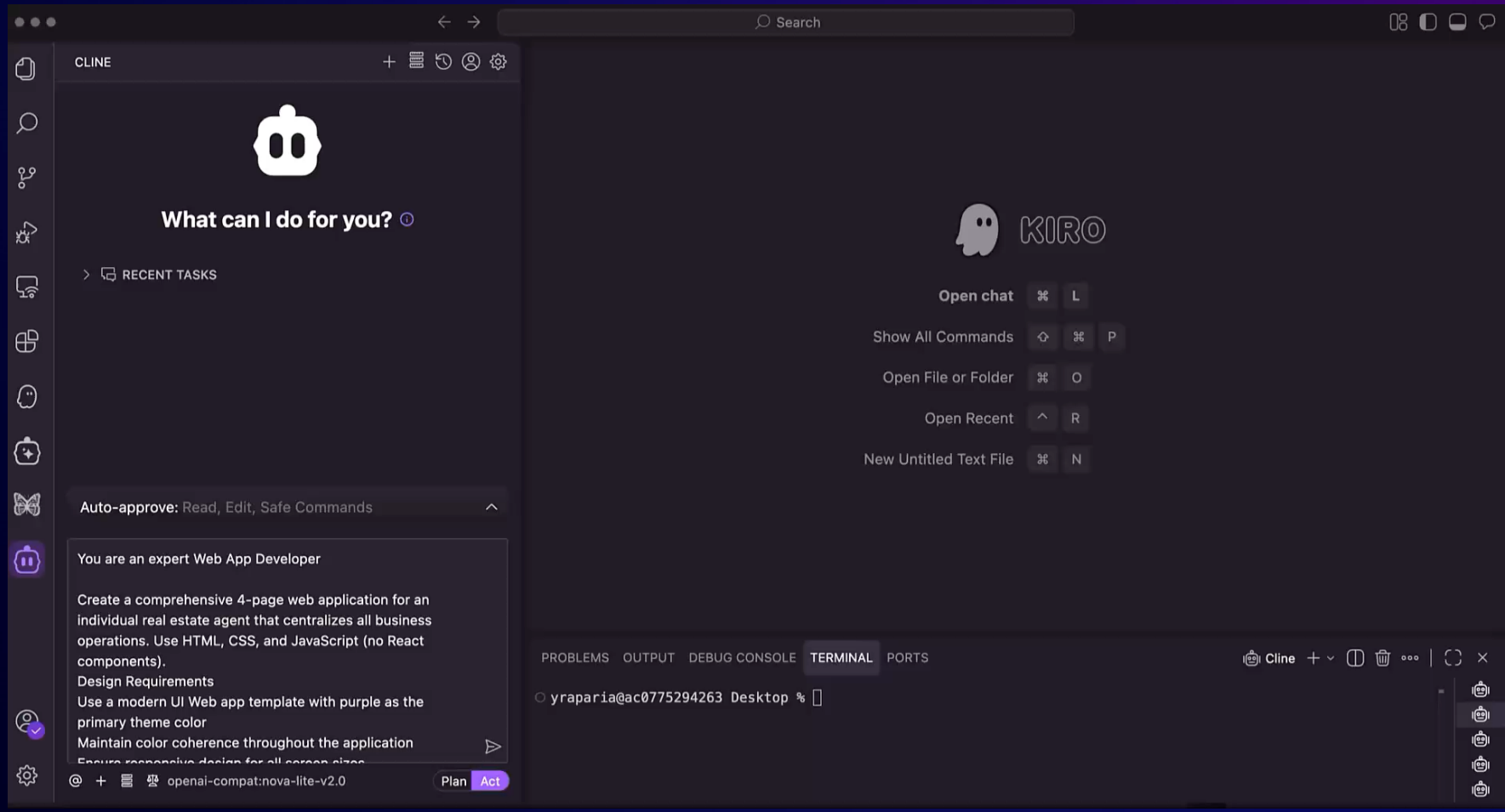
KIRO

The AI IDE for prototype to production



Kiro helps you do your best work by bringing structure to AI coding with spec-driven development.

Watch Nova 2 create a functioning web app in one shot using Cline extension in Kiro



SageMaker AI - New capabilities, at a glance

Serverless RFT and SFT	*Advanced tuning techniques without configuring infrastructure*
Flexible deployment	*1-click deployment to SageMaker or Bedrock, and exportable model weights saved to S3*
Synthetic data generation	*Agent-guided SDG for save time sourcing training data*
Deep evaluation	*LLM-as-a-Judge, custom scorer, and public benchmarks, including built-in visualizations*
Serverless MLFlow	*Advanced experiment tracking, fully integrated and serverless*
Optimized hyperparams	*Built on pre-optimized recipes based on pairing of model, technique, and instance*

Closing remarks

Thank you!