Architecture Core LLM Platform

CACIB Gen AI

AWS Professional Services
15th May 2024

Agenda

1. Introduction & Context
2. Logical component of a Gen AI platform
3. Requirements for wave 1 applications go to production
4. Mapping CACIB use cases with Gen AI platform
5. Design Principles
6. Gen AI Core Platform Blueprint
7. Use cases architecture proposal

Introduction & Context

Starting April 22nd, the program aims to deploy four use cases in production by September, leveraging a scalable and innovative Generative AI Core Platform on AWS.

Four discovery workshops have been conducted to capture the specific requirements of the use cases. Additional focused workshops are planned (FinOps, DevOps, Security) to delve deeper into the remaining open areas to enhance the global architecture design, ensuring that synergies and opportunities for mutualization are identified.

Through this presentation, we are aiming at aligning all the key stakeholders around :

- The definition of the Gen AI Core Platform Lot 1 for September
- Architecture principles to be validated and the associated action plan

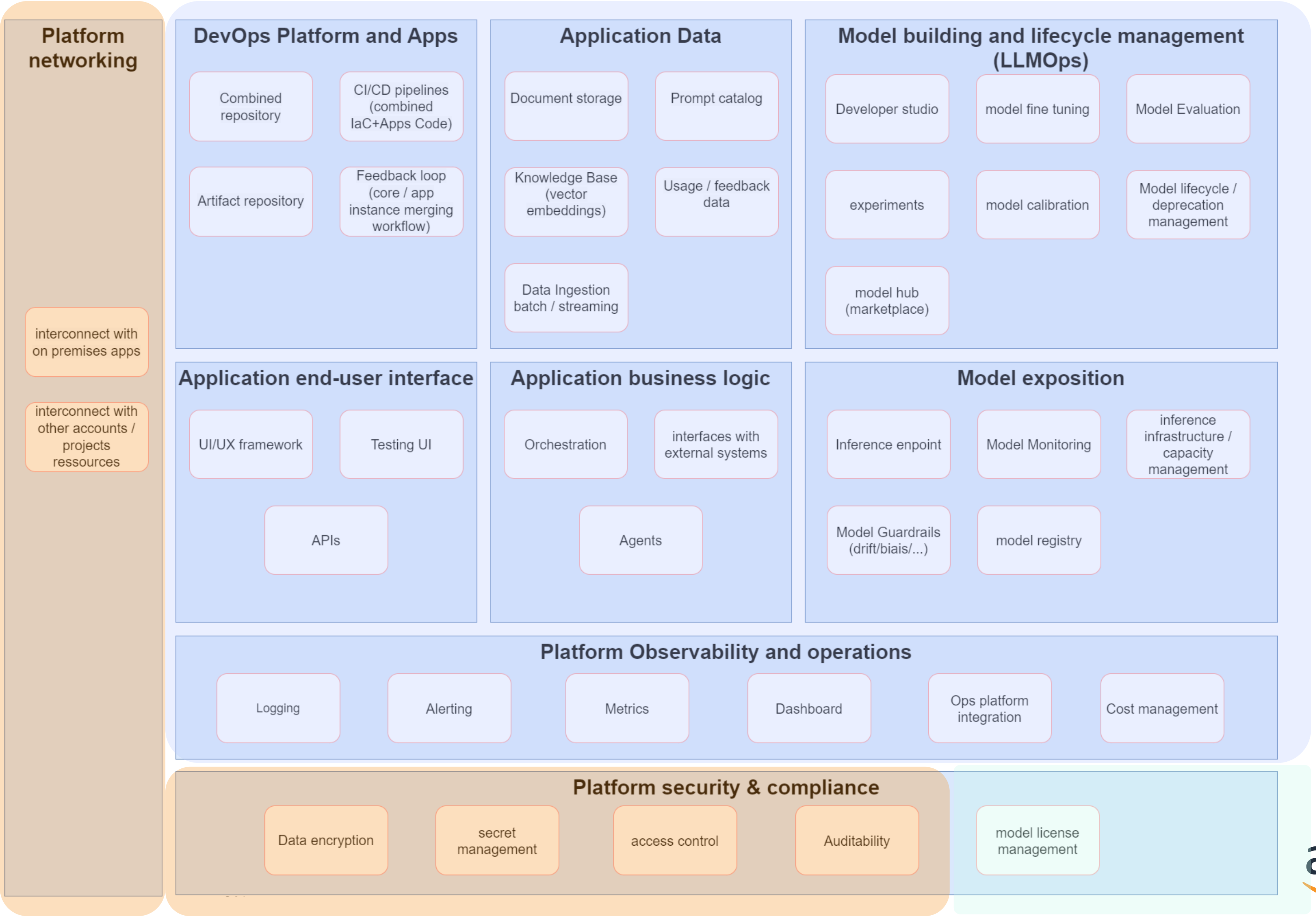
The global alignment of all stakeholders is crucial to ensure that the various architectural documents (GAD, DAD, RPAD) are implemented within the suggested timeframe (end of May).

Logical components of a Gen AI platform

Managed by the Landing zone

Core platform capabilities

Governance dependant capability



Platform networking

- interconnect with on premises apps
- interconnect with other accounts / projects ressources

DevOps Platform and Apps

- Combined repository
- CI/CD pipelines (combined IaC+Apps Code)
- Artifact repository
- Feedback loop (core / app instance merging workflow)

Application Data

- Document storage
- Prompt catalog
- Knowledge Base (vector embeddings)
- Usage / feedback data
- Data Ingestion batch / streaming

Model building and lifecycle management (LLMOps)

- Developer studio
- model fine tuning
- Model Evaluation
- experiments
- model calibration
- Model lifecycle / deprecation management
- model hub (marketplace)

Application end-user interface

- UI/UX framework
- Testing UI
- APIs

Application business logic

- Orchestration
- interfaces with external systems
- Agents

Model exposition

- Inference enpoint
- Model Monitoring
- inference infrastructure / capacity management
- Model Guardrails (drift/biais/...)
- model registry

Platform Observability and operations

- Logging
- Alerting
- Metrics
- Dashboard
- Ops platform integration
- Cost management

Platform security & compliance

- Data encryption
- secret management
- access control
- Auditability
- model license management

aws financial services

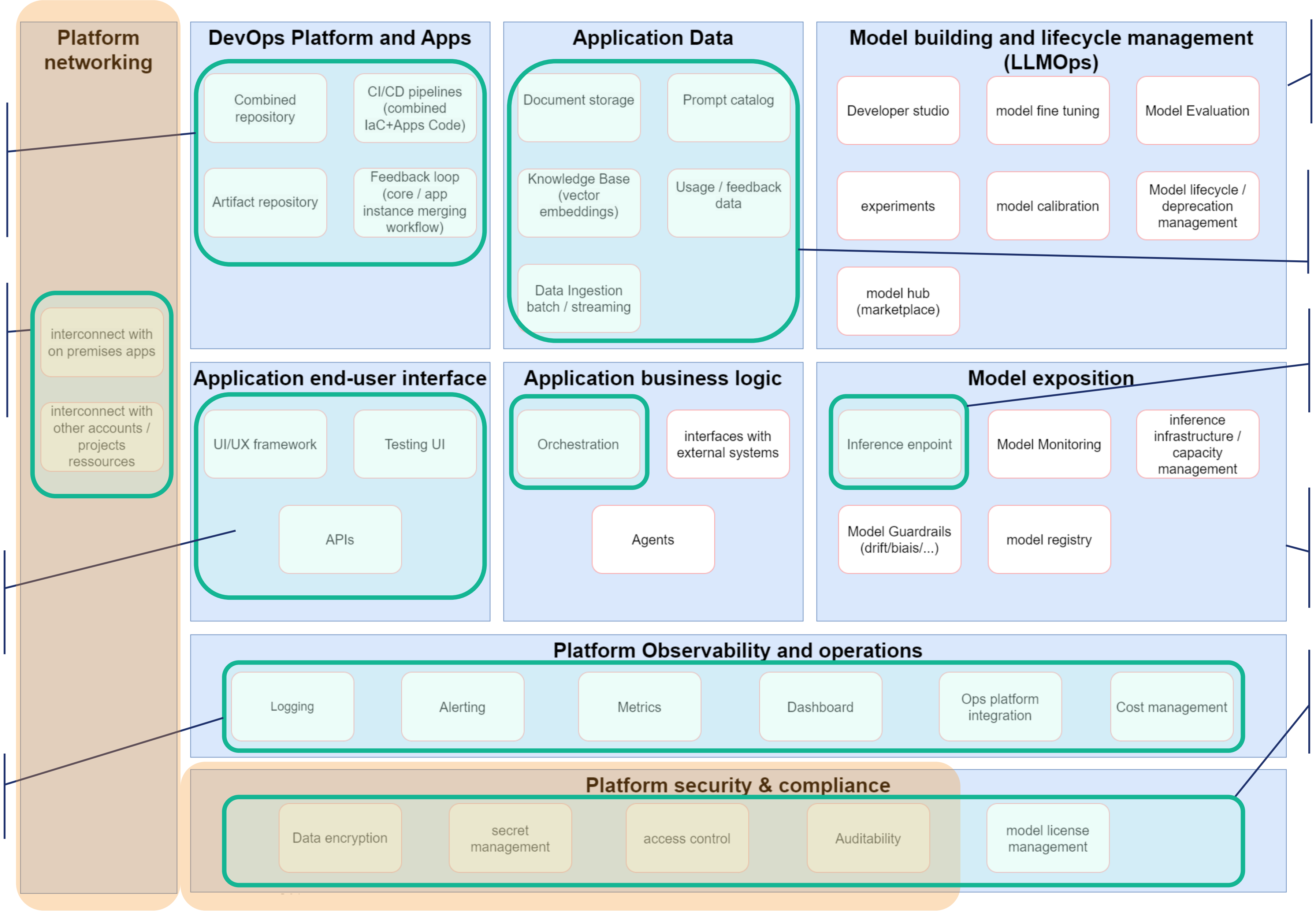
Requirements for wave 1 applications go to production

Foundation for production grade platform.
Combined repo required for proper management of serverless resources under data science governance

Required for :
- flow opening to DevOps tooling,
- User Authentication
- Data ingestion from on-prem sources
…

Required for hosting existing and future UI components
Out of scope (v1): development framework for UI – such as React

Foundation for production grade platform.



Platform networking

- interconnect with on premises apps
- interconnect with other accounts / projects ressources

DevOps Platform and Apps

- Combined repository
- CI/CD pipelines (combined IaC+Apps Code)
- Artifact repository
- Feedback loop (core / app instance merging workflow)

Application Data

- Document storage
- Prompt catalog
- Knowledge Base (vector embeddings)
- Usage / feedback data
- Data Ingestion batch / streaming

Model building and lifecycle management (LLMOps)

- Developer studio
- model fine tuning
- Model Evaluation
- experiments
- model calibration
- Model lifecycle / deprecation management
- model hub (marketplace)

Application end-user interface

- UI/UX framework
- Testing UI
- APIs

Application business logic

- Orchestration
- interfaces with external systems
- Agents

Model exposition

- Inference enpoint
- Model Monitoring
- inference infrastructure / capacity management
- Model Guardrails (drift/biais/...)
- model registry

Platform Observability and operations

- Logging
- Alerting
- Metrics
- Dashboard
- Ops platform integration
- Cost management

Platform security & compliance

- Data encryption
- secret management
- access control
- Auditability
- model license management

Out of scope (v1): custom model development, fine-tuning and model lifecycle management

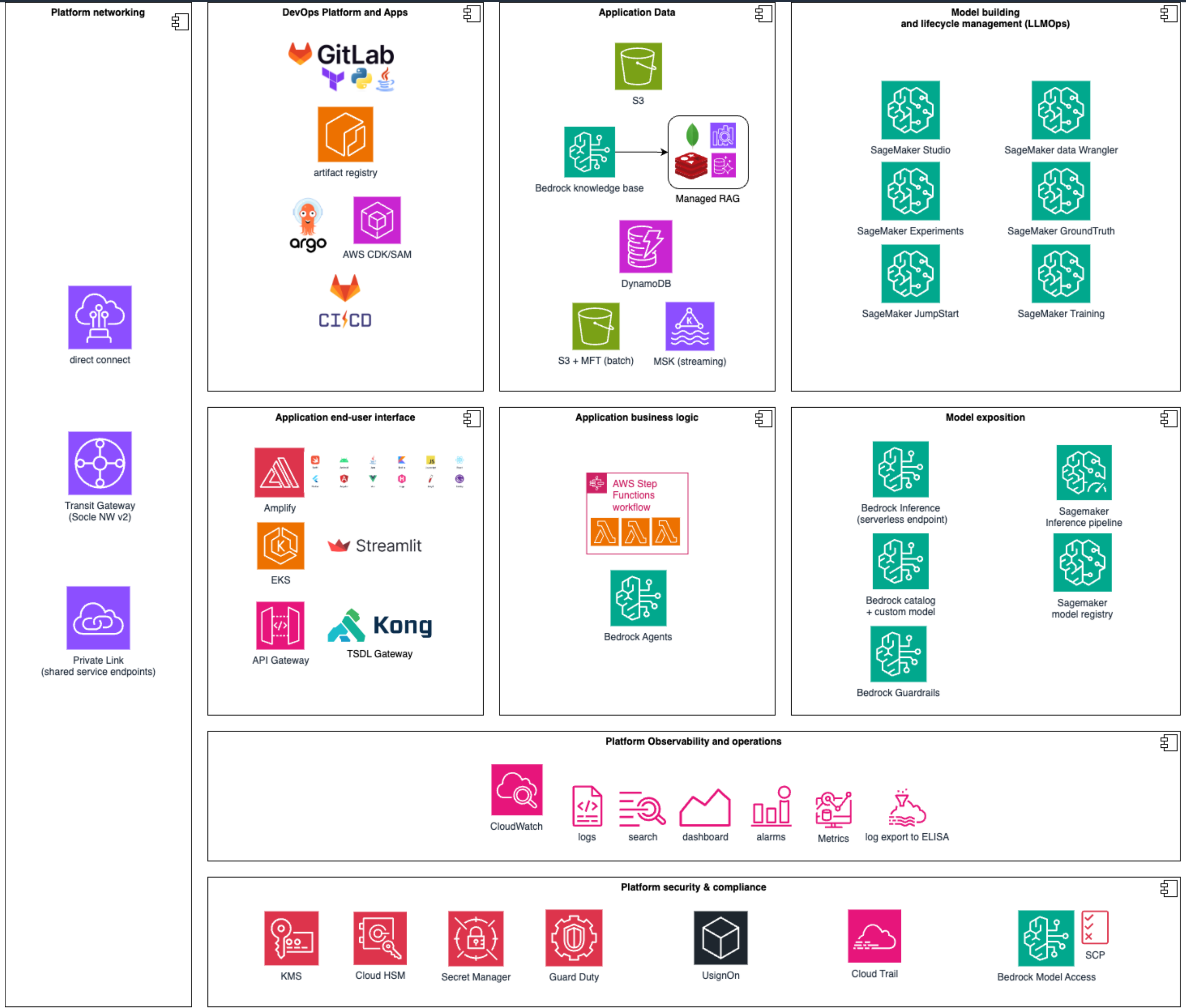
Required for RAG-based apps

Out of scope (v1): orchestration that requires integration with 3rd party systems.

Out of scope (v1): infrastructure management for non-serverless endpoint. Automated model monitoring (performance, not observability)

Foundation for production grade platform.

Mapping logical components with AWS services



Platform networking

direct connect

Transit Gateway
(Socle NW v2)

Private Link
(shared service endpoints)

DevOps Platform and Apps

GitLab

artifact registry

argo

AWS CDK/SAM

CI/CD

Application Data

S3

Bedrock knowledge base

Managed RAG

DynamoDB

S3 + MFT (batch)

MSK (streaming)

Model building and lifecycle management (LLMOps)

SageMaker Studio

SageMaker data Wrangler

SageMaker Experiments

SageMaker GroundTruth

SageMaker JumpStart

SageMaker Training

Application end-user interface

Amplify

EKS

Streamlit

API Gateway

Kong

TSDL Gateway

Application business logic

AWS Step Functions workflow

Bedrock Agents

Model exposition

Bedrock Inference
(serverless endpoint)

Sagemaker
Inference pipeline

Bedrock catalog
+ custom model

Sagemaker
model registry

Bedrock Guardrails

Platform Observability and operations

CloudWatch

logs

search

dashboard

alarms

Metrics

log export to ELISA

Platform security & compliance

KMS

Cloud HSM

Secret Manager

Guard Duty

UsignOn

Cloud Trail

SCP

Bedrock Model Access

Design orientation principles

- Prioritize serverless services over managed or non-managed infrastructure resources to start small, iterate quickly and scale fast.
- Prefer managed over non-managed services if no restrictions stating the opposite are in place.
- Modularize functionalities and components whenever possible to improve platform evolvability and scalability.
- Define the code base and its automation processes (merge, deploy) in a single repository that acts as the source of truth for all the base components of the core platform.
- Reuse existing CA and CACIB vetted cloud architecture patterns (SSO, integration, etc) whenever possible to streamline deployment and simplify operations.
- Enforce zero trust security accross the core platform, within and across accounts.

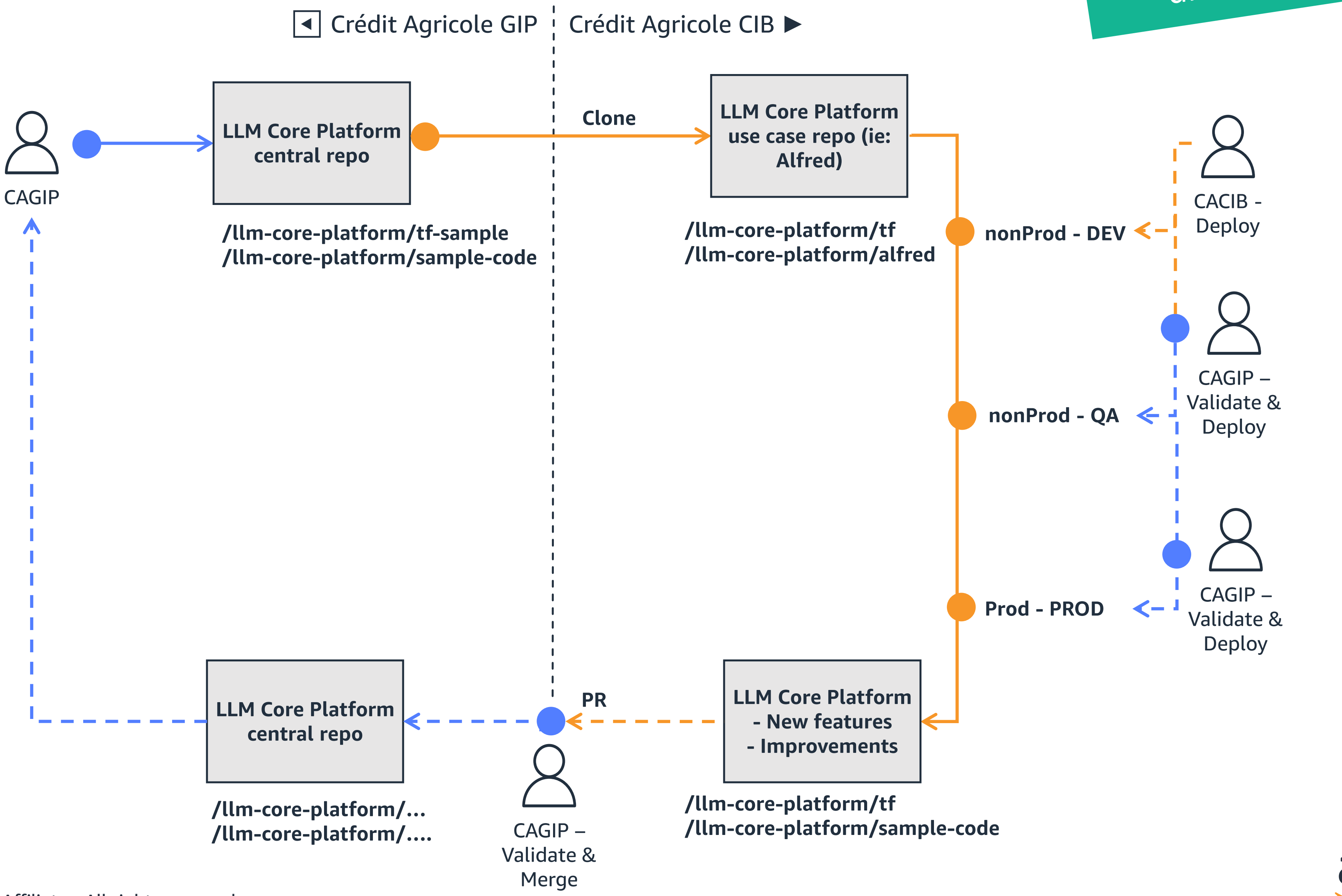
What is the Core Platform ?

Based on the design principles and the architecture decisions proposed, the **Core platform is**:

- ✓ Infrastructure as code blueprint
- ✓ Re-usable modules, artifacts & patterns as you go
- ✓ Shared standards for LLM applications
- ✓ Evolutive modular & decoupled architecture
- ✓ One-to-one project and blueprint mapping (1 project = 1 deployed blueprint)
- ✓ Traceable and scalable feature addition and improvements (via PRs)
- ✓ Centralized roadmap with decentralized management and ownership
- ✓ Quality gates between promotion stages to ensure best practices

DevOps proposed approach

WiP - Pending for DevOps discovery workshop



◀ Crédit Agricole GIP

Crédit Agricole CIB ▶

CAGIP

LLM Core Platform central repo

/llm-core-platform/tf-sample
/llm-core-platform/sample-code

Clone

LLM Core Platform use case repo (ie: Alfred)

/llm-core-platform/tf
/llm-core-platform/alfred

CACIB - Deploy

nonProd - DEV

CAGIP – Validate & Deploy

nonProd - QA

CAGIP – Validate & Deploy

Prod - PROD

LLM Core Platform
- New features
- Improvements

/llm-core-platform/tf
/llm-core-platform/sample-code

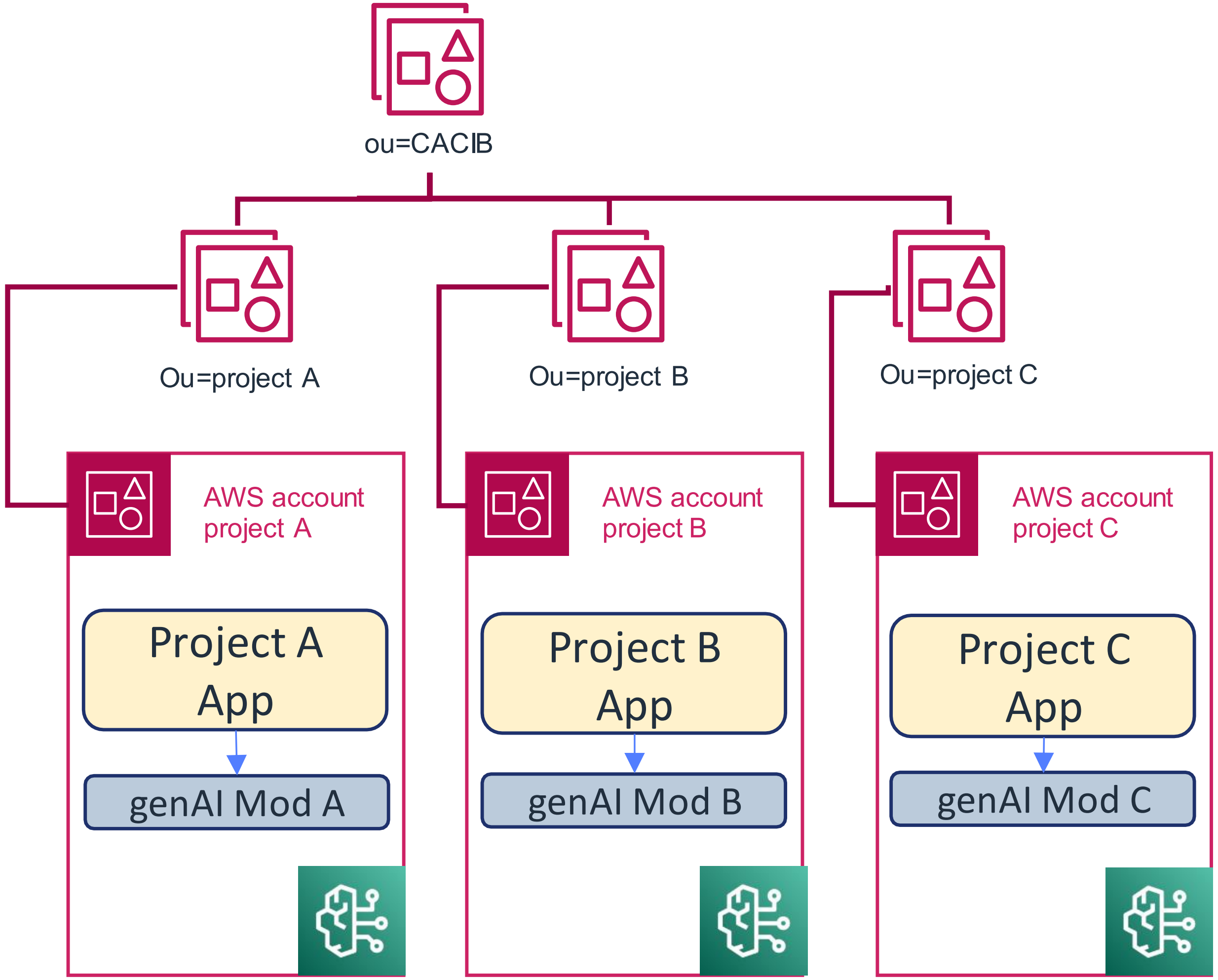
PR

CAGIP – Validate & Merge

LLM Core Platform central repo

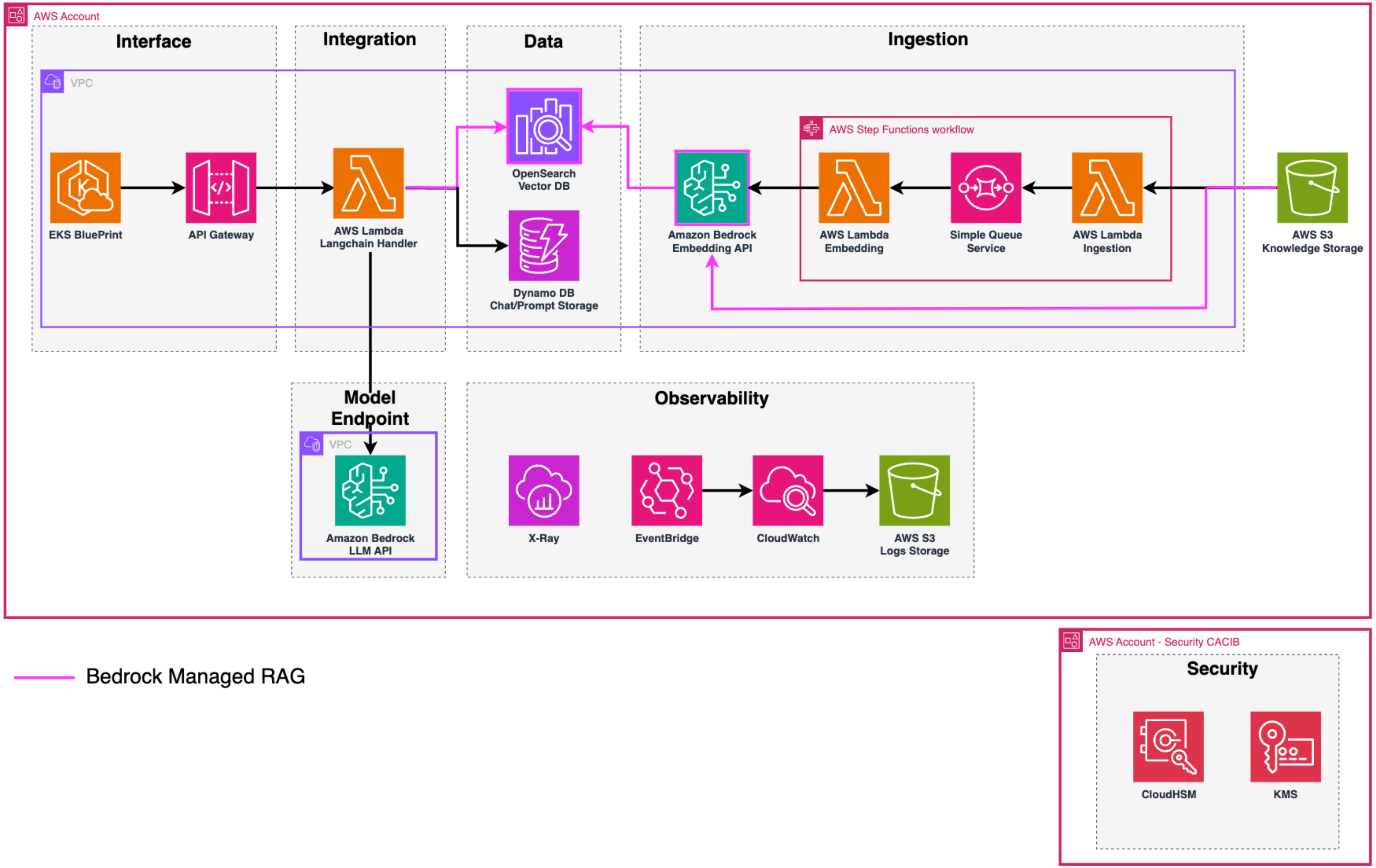
/llm-core-platform/...
/llm-core-platform/....

Decentralized platform approach



- Easier project level FinOps
- More flexibility for core model tuning
- Promote decentralized DevOps practices and empowers teams
- Facilitate reversibility through project independence
- Aligned with CACIB governance and processes for project qualification, architecture validation and security review.
- Aligned with general cloud best practices and Landing Zone usage guidelines.
- Aligned with existing CACIB projects on AWS
- No minimal deployment footprint (scale to 0)

Gen AI Core Platform Blueprint

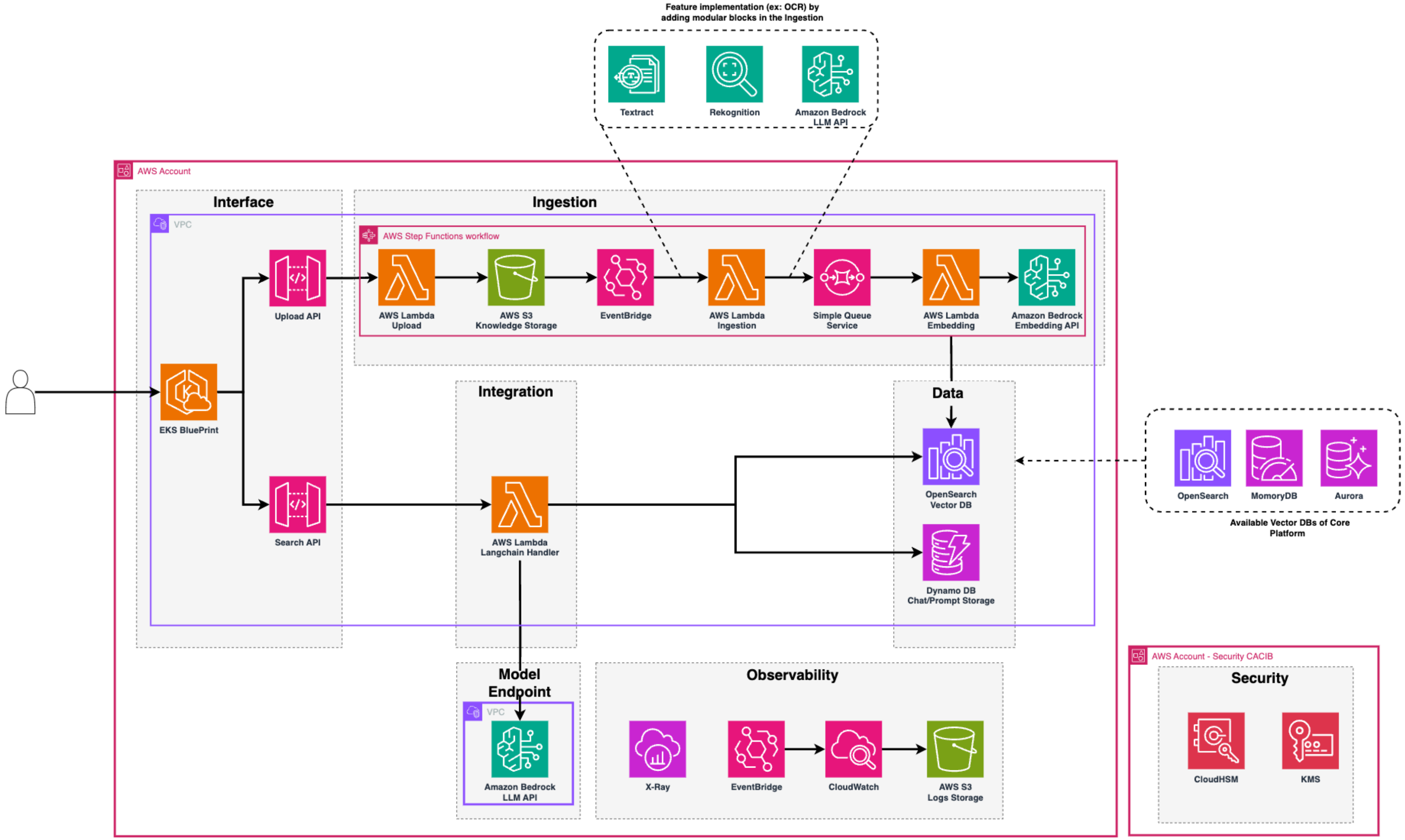


aws financial services

Alfred Discovery card

1. Use case objective:
   - Alfred is a GenAI application that provides intelligent document assistance for users to ask questions and receive answers based on ingested documents.
   - The success criteria are not defined yet, performance metrics is being defined
2. Data Ingestion:
   - Data types: Currently supports PDF files.
   - Data volume: Maximum 20 MB per document, no overall limit on the number of documents.
   - Data ingestion process: Manual import of documents.
   - Data organization: Concept of "folders" to group multiple documents as one.
3. Technical stack:
   - Front-end: React SPA with custom components for displaying answers.
   - Data storage: Local file storage for documents, Elasticsearch for indexing.
   - Language model: GPT-3.5 Turbo (chosen for speed over GPT-4 due to cost).
4. Challenges & evolutions:
   - OCR capability.
   - Support for C3, Word & PPT source data.
   - Table & Image handling.
   - Access to other source of data (internet, paid press like Bloomberg).
   - Performance: Improve upload performance (10-15 seconds for PDF text extraction)

Proposed ALFRED Architecture on AWS

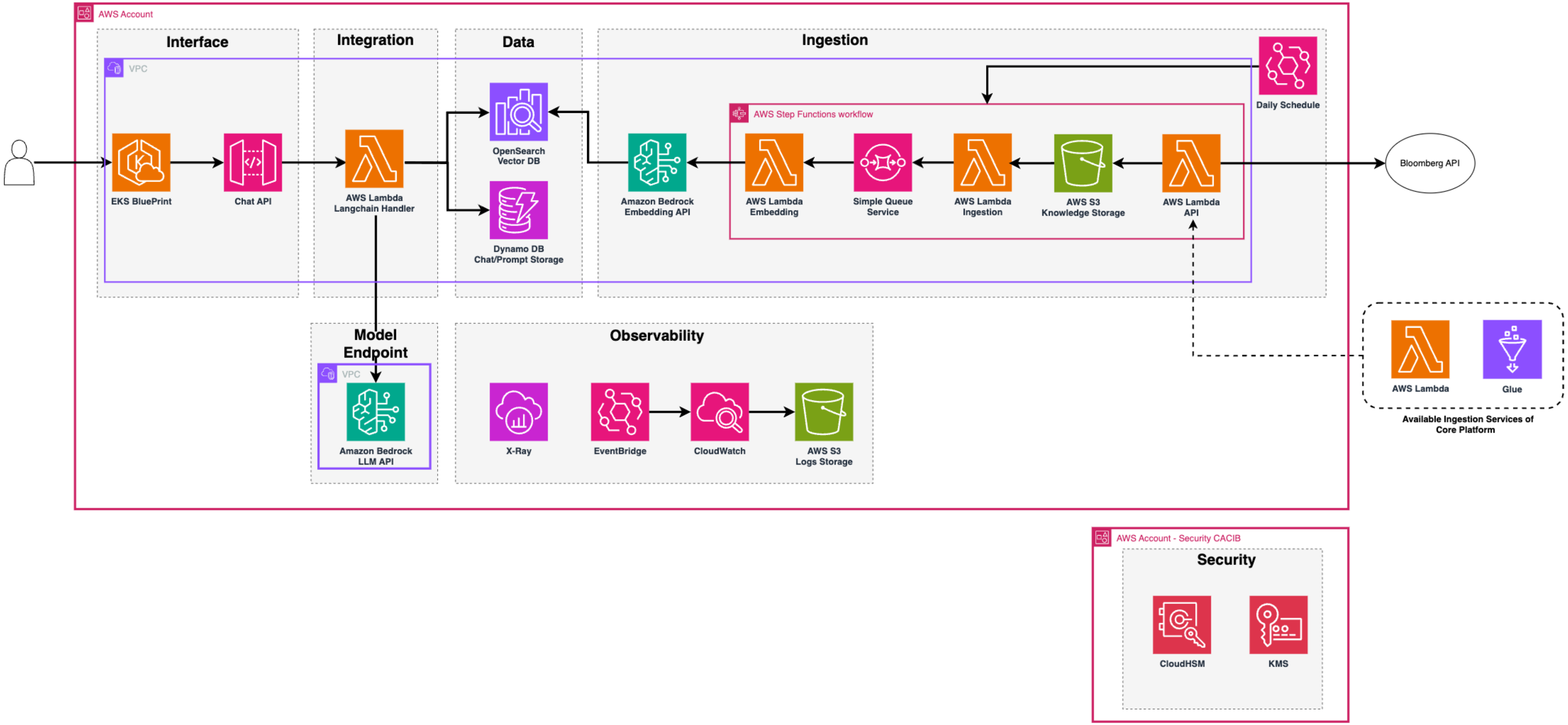


- - - - - - - - - - Re-usable modules, artifacts & patterns

Augmented Sales Discovery card

1. Use case objective:
   - Augmented Sales support: Enhance the response time for sales teams to answer customer questions.
   - Success criteria: Less than 3 seconds response time with 300 users (front office).
2. Data Ingestion:
   - Data types: PDFs, with the ability to choose the source (only in English).
   - Data volume: 15 documents per day (max 20 pages), 1500 documents in batch.
   - Data sources: Bloomberg, Wiki (Ayca).
   - Ingestion frequency: Daily for Bloomberg, more frequent for Wiki.
3. Technical stack:
   - Selected model: Mistral 8x7b (based on chunk size and date range).
   - Index: Elastic Search.
   - UI: Streamlit.
4. Challenges & evolutions:
   - Data sources: Multiple data sources and segmentation based on user sign-on.
   - User feedback: Implement thumbs up/down and written feedback.
   - Document handling: Document splitter for tables and images (requirements, document analyzer developed internally).
   - Language support: Currently English, but planning to support French.
   - New features: Recommendation on past queries, suggestions.
   - Evaluation: Quantify the impact of hyperparameters (chunk size) on response quality and time.

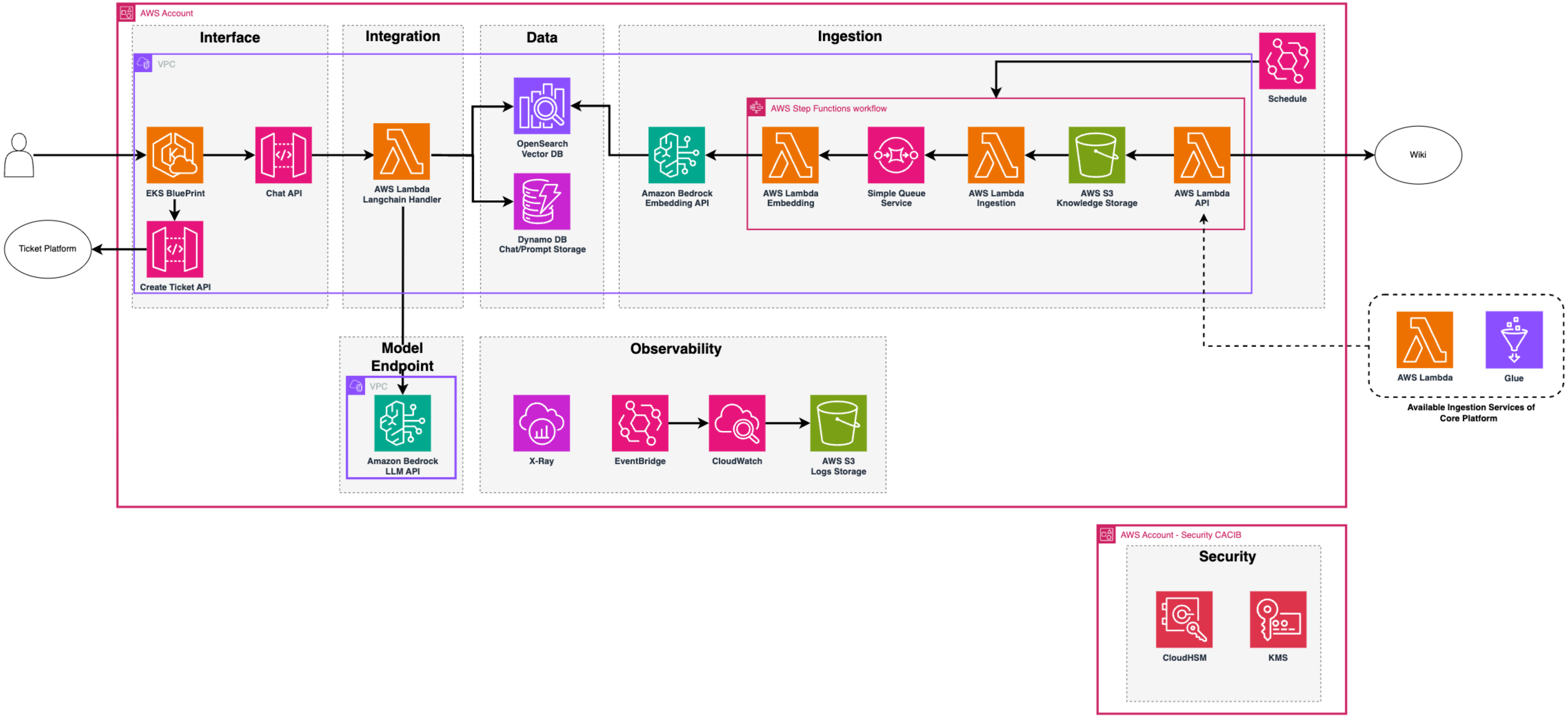
Augmented Sales Architecture on AWS



L1/L2 Support discovery card

1. Use case objective:
   - Technical Answer questions based on the target knowledge base
   - Generate pre-filled tickets
   - Provide a centralized access point for end-users to interact with a chatbot before reaching L1/L2 support
2. Data Ingestion:
   - 4 knowledge base sources, mainly from Confluence
   - Planned weekly batch ingestion
3. Technical stack:
   - Using LLaMA 2 and Naive RAG approach, no OpenAI integration
   - Using LangChain
   - FAISS vector DB et SQLite DB
4. Challenges and evolutions:
   - Table and image extraction, French documentation
   - Need to create labeled data to validate the use case
   - Wish for feedback loop, SSO authentication, UI, and performance monitoring

L1/L2 Support Architecture on AWS

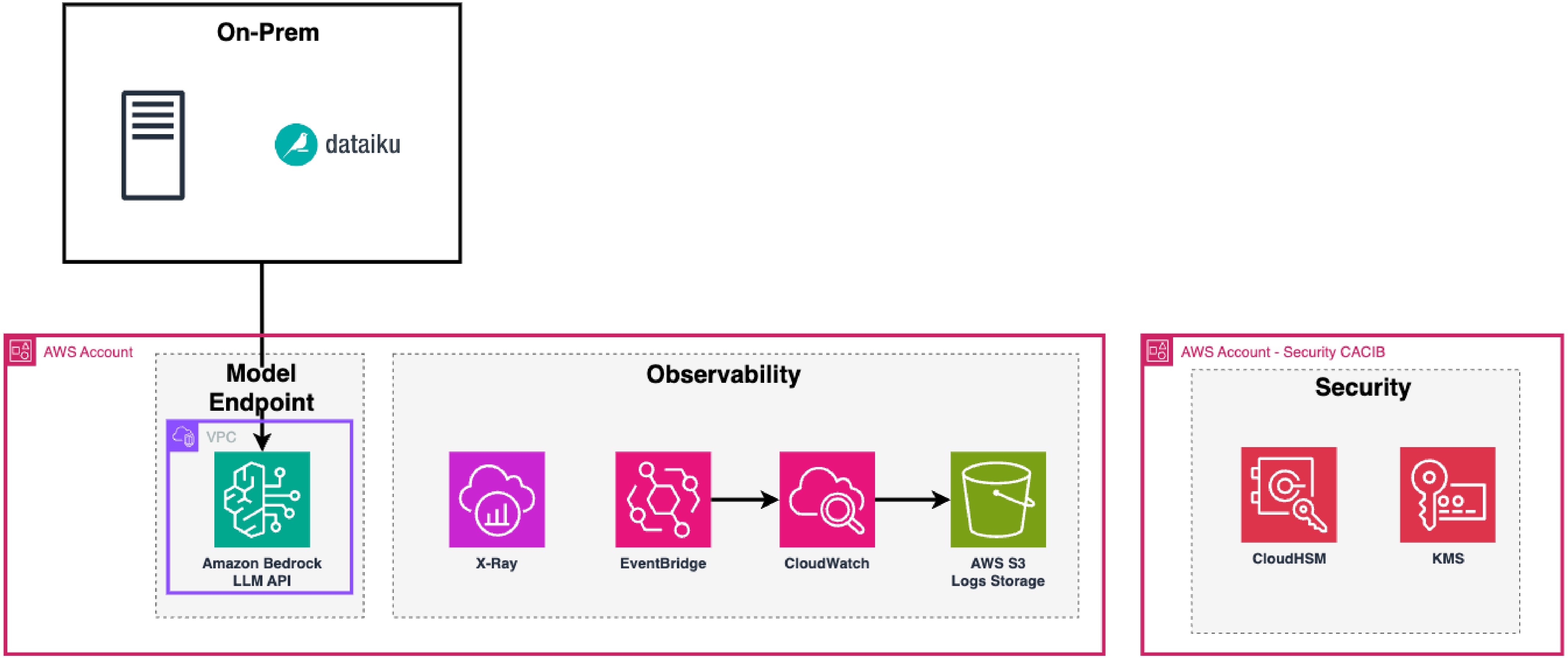


aws financial services

Market Colors Discovery discovery card

1. Use case objective:
   - The objective is to generate a daily newsletter/report for traders, combining public articles (news selection) and traders' insights.
   - The success criteria are to generate the report within approximately 30 minutes to 1 hour.
2. Data Ingestion:
   - Data sources: Structured data (Repo Data (KDB) and Market Data, Unstructured data (Web scraping from specific sites and user copy/paste)
   - Data ingestion frequency: Daily basis
3. Technical stack:
   - Data preparation Ingestion and ingestion: Dataiku (Data Science Platform - DSS), GPT
4. Challenges & evolutions:
   - Model insight generation improvement
   - Architecture decision - Hybrid architecture approach Front / Back remains On Premise leveraging Dataiku + Integration to AWS to access to LLM capabilities leveraging Amazon BedRock

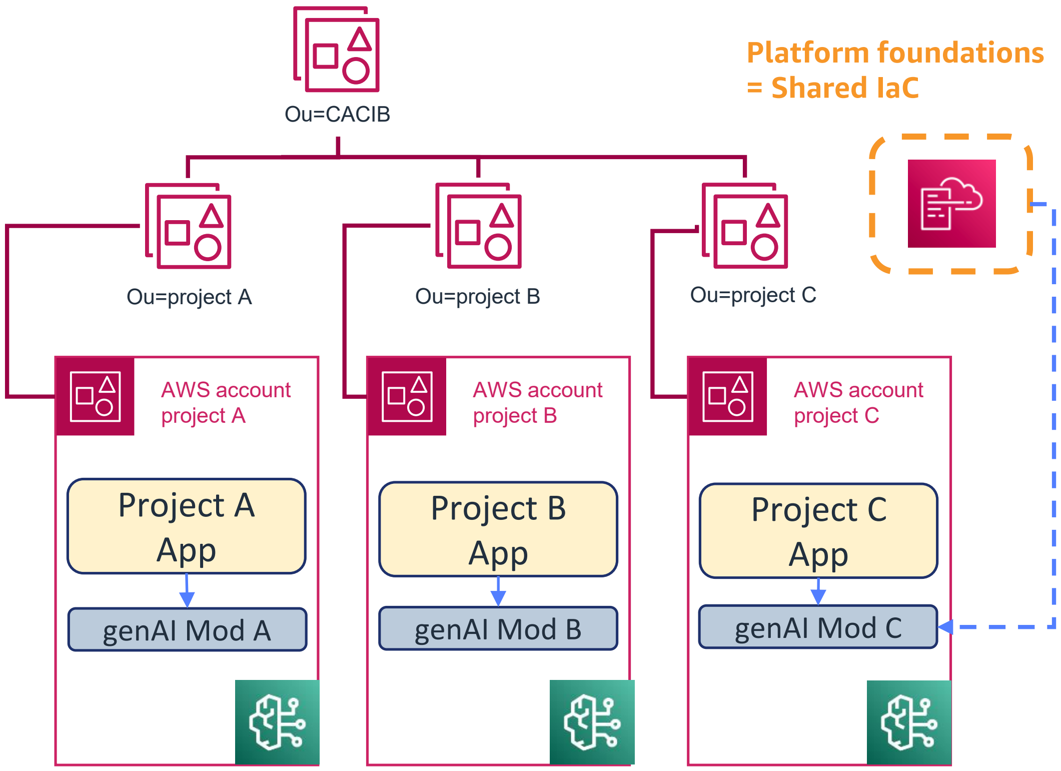
Market Colors Architecture on AWS



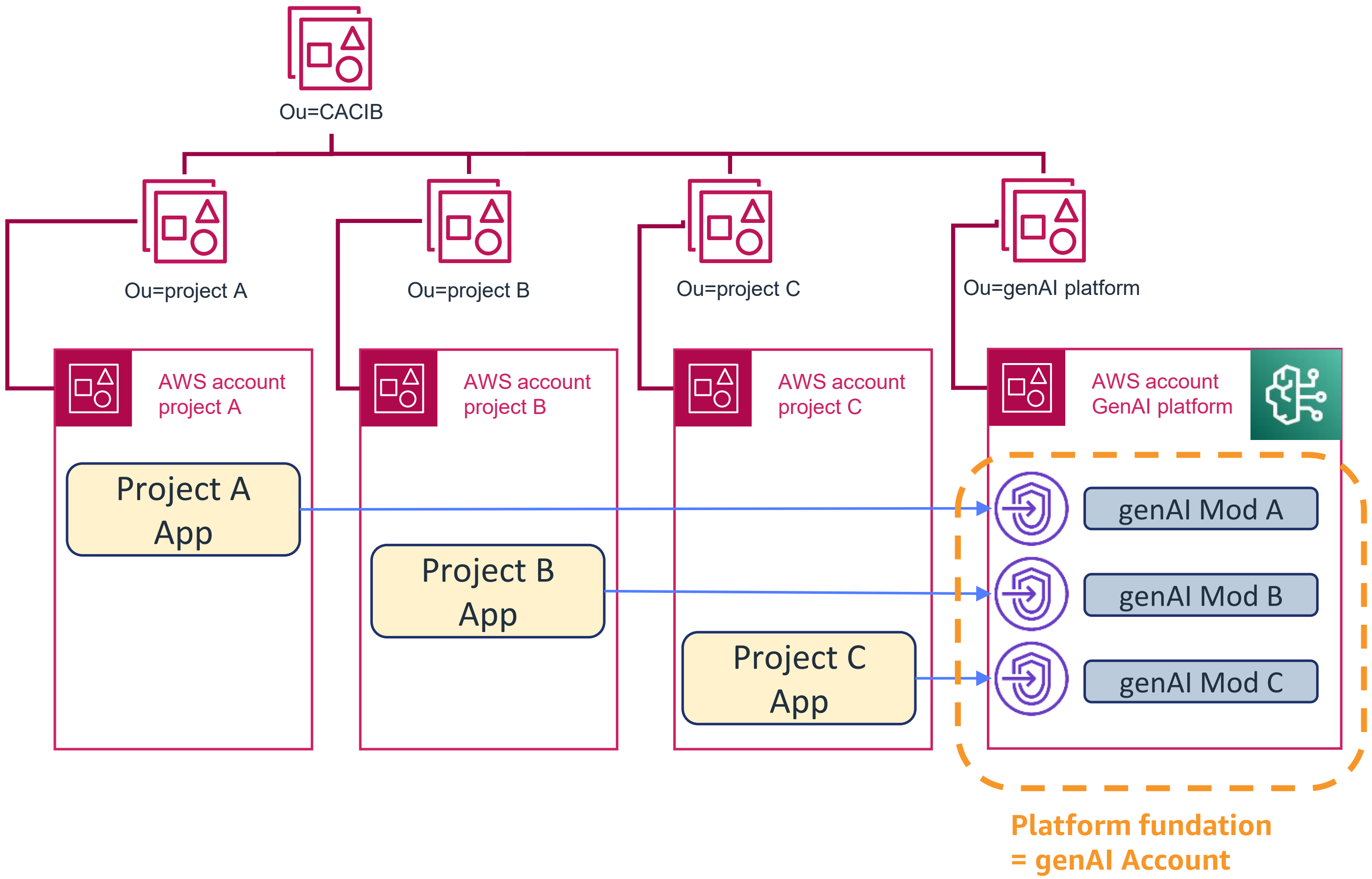
Appendix

Centralized and Decentralized platform approaches

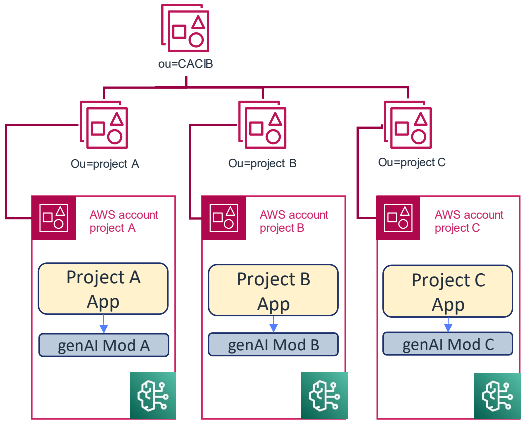
Decentralized platform



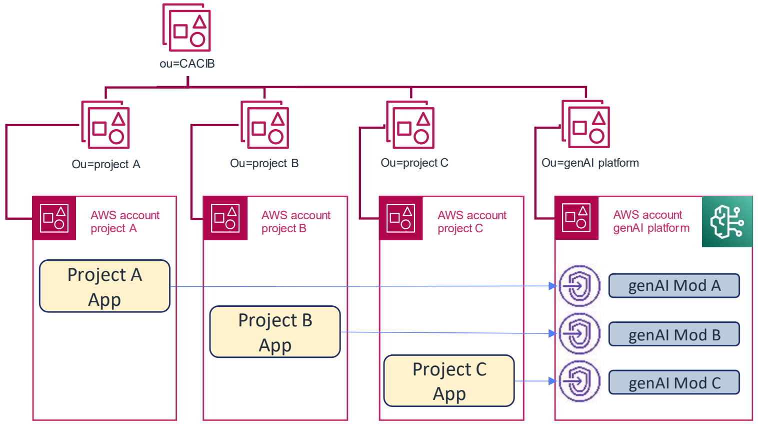
Centralized platform



Centralized and Decentralized platform approaches



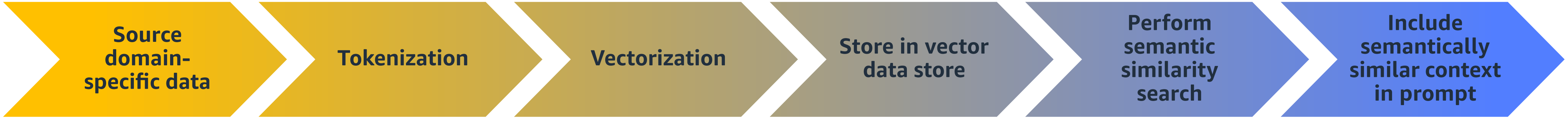
- Single tenant/project deployment
- Platform provides instantiated running services per project
- Platform centralized operations. Requires genAI platform cross project operations team.
- Architecture guidelines are enforced by DevOps practices and LZ policies
- genAI consumption structured per project according to LZ guidelines for general projects. FinOps is at the project level.
- Capabilities can be expanded for each project.



- Multi tenant/project deployment
- Platform provides shared running services
- Platform decentralized operations. Requires each project operations to integrate genAI component operations.
- Architecture guidelines are enforced by genAI platform project and DevOps practices.
- genAI consumption consolidated in a single account. Opportunity for cross projects finOps.
- Capabilities constrained by platform.

aws financial services

What are vector embeddings?



Source domain-specific data → Tokenization → Vectorization → Store in vector data store → Perform semantic similarity search → Include semantically similar context in prompt

Documents

Audio/video

Images

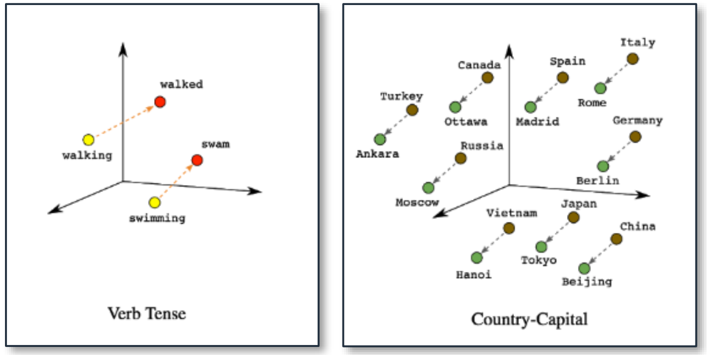
Semantic elements:

- Words, phrases
- Paragraphs, documents
- Scenes, song sections
- Faces, detected picture elements
- And more

0.35 0.1 0 0.9 001.0 00 0001.0 0 0...

0.35 0.1 0 0.8 001.0 00 0001.0 0 0...

0.15 0.1 0 0.7 001.0 00 0001.0 0 0...

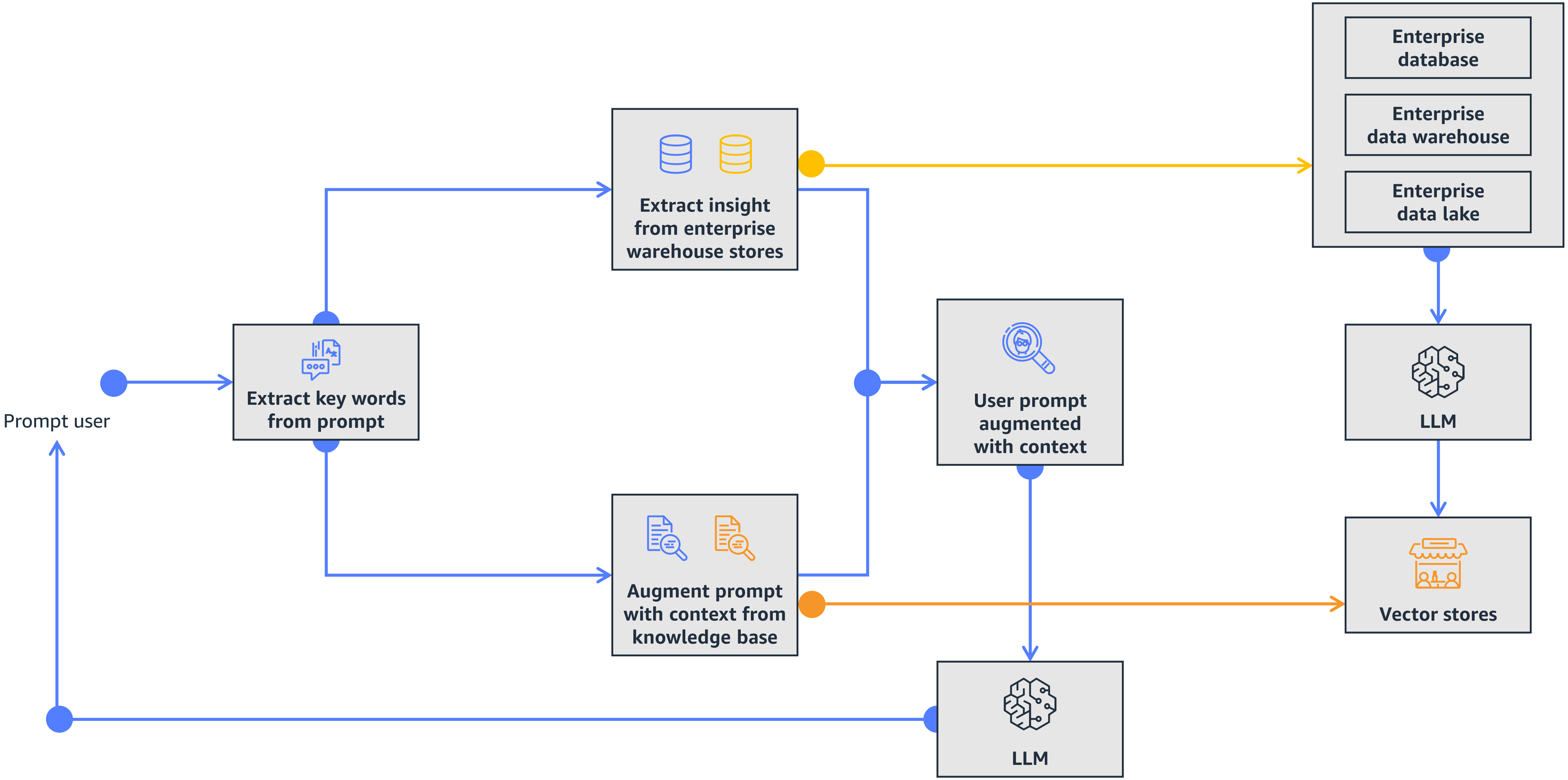


3D simplified representation. Embeddings can have thousands of dimensions. Source: https://daleonai.com/embeddings-explained

Embeddings: When vector elements are semantic, used in generative AI

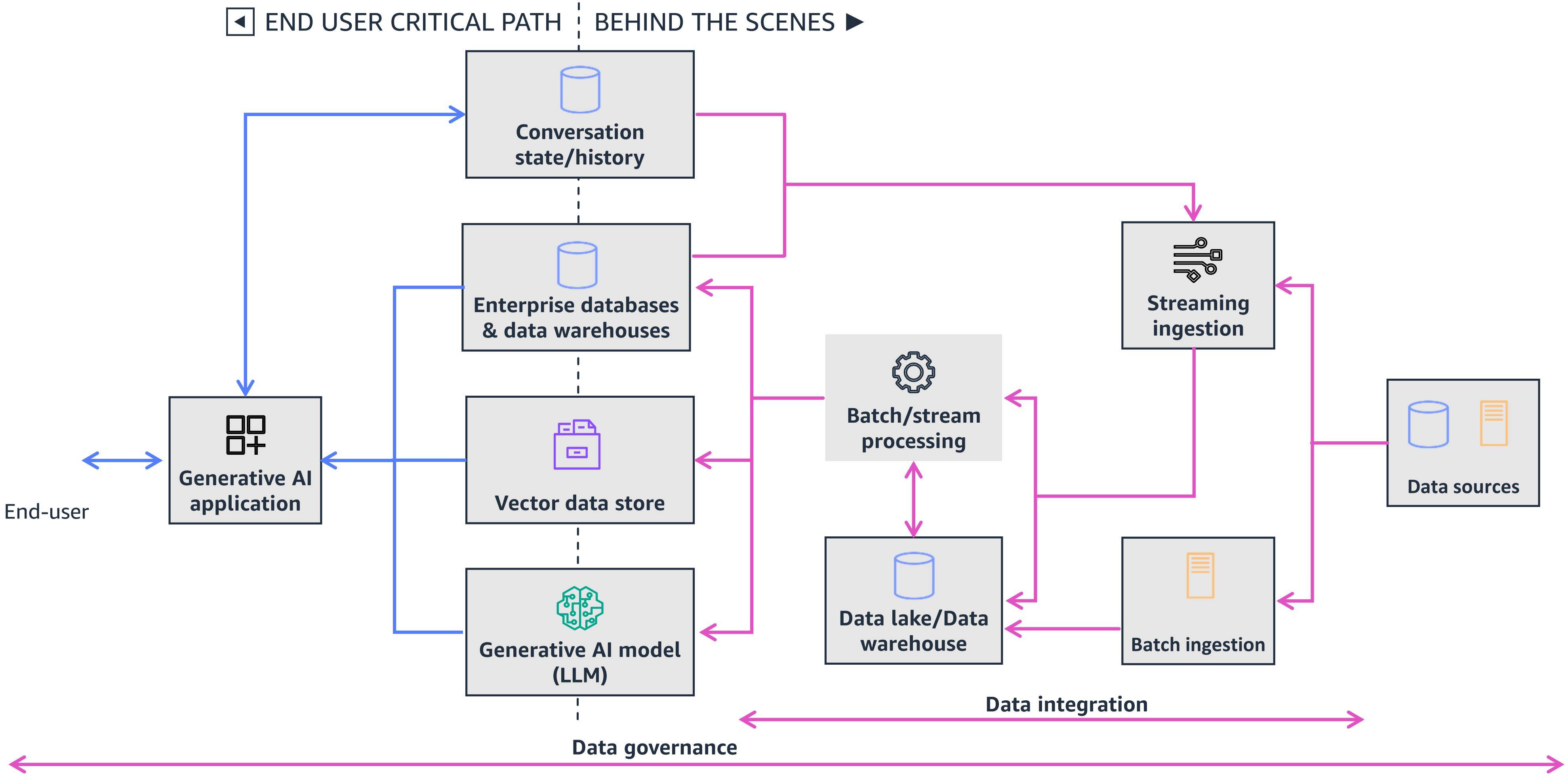
aws financial services

A Retrieval-Augmented Generation (RAG) pattern


Enterprise database
Enterprise data warehouse
Enterprise data lake
Extract insight from enterprise warehouse stores
Extract key words from prompt
Prompt user
User prompt augmented with context
LLM
Augment prompt with context from knowledge base
Vector stores
LLM

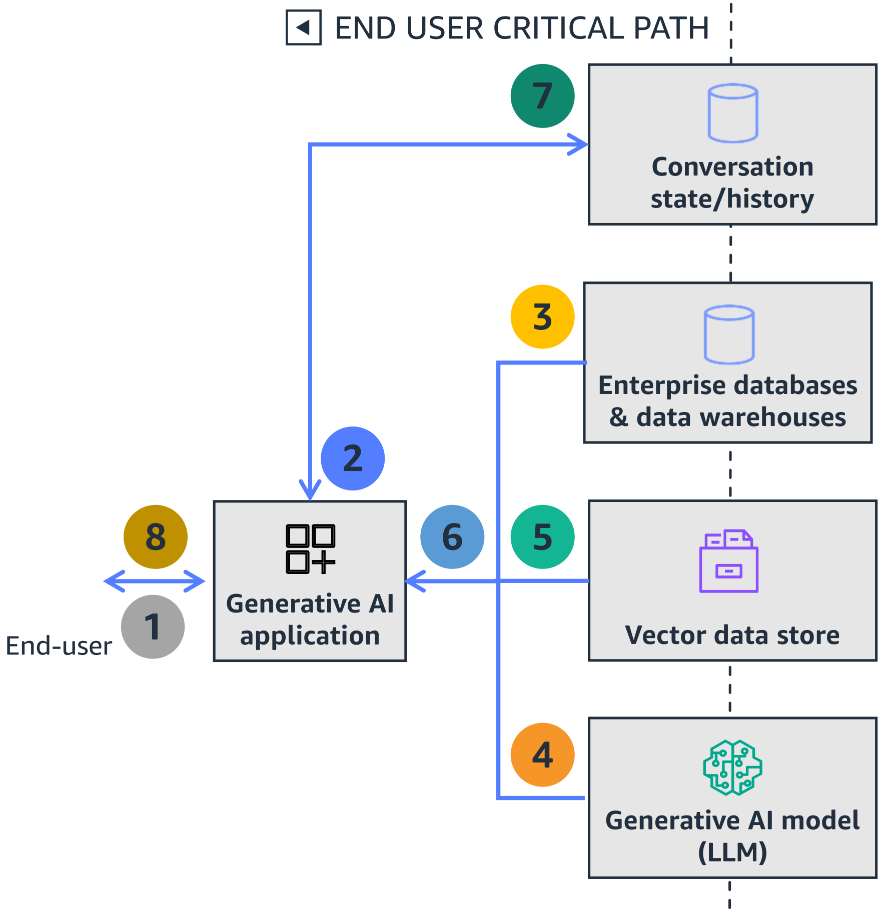
aws financial services

RAG reference architecture



aws financial services

Data in the end user critical path


◄ END USER CRITICAL PATH
7
Conversation state/history
3
Enterprise databases & data warehouses
2
8
6
5
Generative AI application
Vector data store
End-user
1
4
Generative AI model (LLM)

1. User asks question
2. Get conversation state/history
3. Get situational context
4. Tokenize, get question embedding from LLM
5. Run similarity search for question embedding, get top matching text
6. Invoke LLM w/ engineered prompt
7. Update conversation state/history
8. Return response

Knowledge bases for Amazon Bedrock

Fully managed native support for Retrieval Augmented Generation (RAG)

Automatically converts text documents into embeddings

Stores embeddings in your vector database

Retrieves embeddings and augments prompts

Vector databases and embedding models for Amazon Bedrock

Vector engine for Amazon OpenSearch Serverless

Redis Enterprise Cloud

Pinecone

Amazon Aurora

MongoDB

Amazon Titan Embeddings Text

Amazon Titan Multimodal Embeddings

Cohere Embed

Amazon Bedrock managed vs non-managed RAG

Amazon Bedrock **managed RAG**

+

- Automatic embeddings
- Managed embeddings in vector DB
- Automatic retrieval

−

- Limited embedding models
- Limited region availability
- A few control on retrieval prompts

Amazon Bedrock **non-managed RAG**

+

- Vast choice of embeddings models
- Several options of vector DB
- Custom retrieval prompts

−

- You might need to tune the embedding model
- You maintain the vector DB
- You integrate the systems