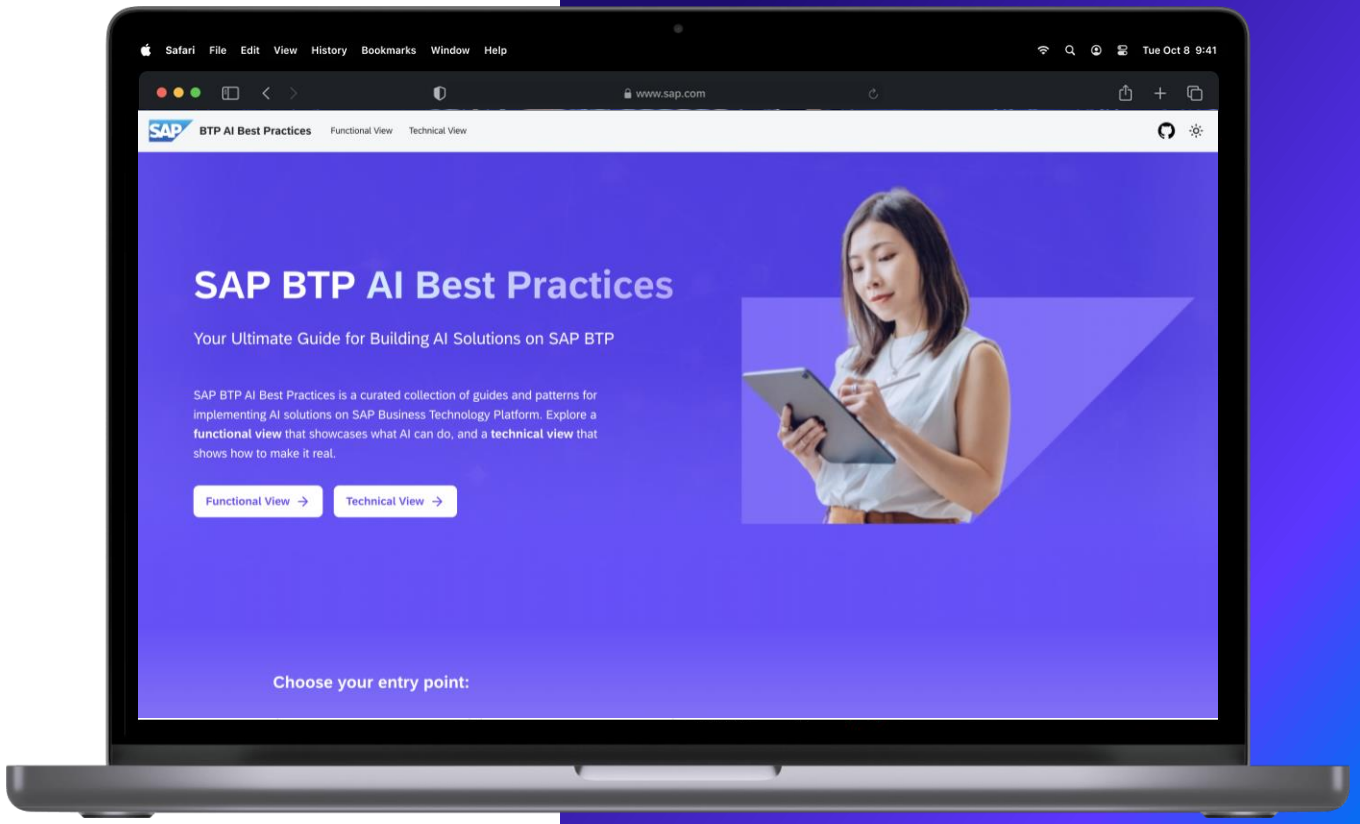
# SAP BTP AI Best Practices

## Vector-based RAG – Query Pipeline

A powerful approach to effectively improve LLM responses with augmented context.




**BTP AI Services Center of Excellence**

**12.05.2025**

# Steps

1 Overview

2 Pre-requisites

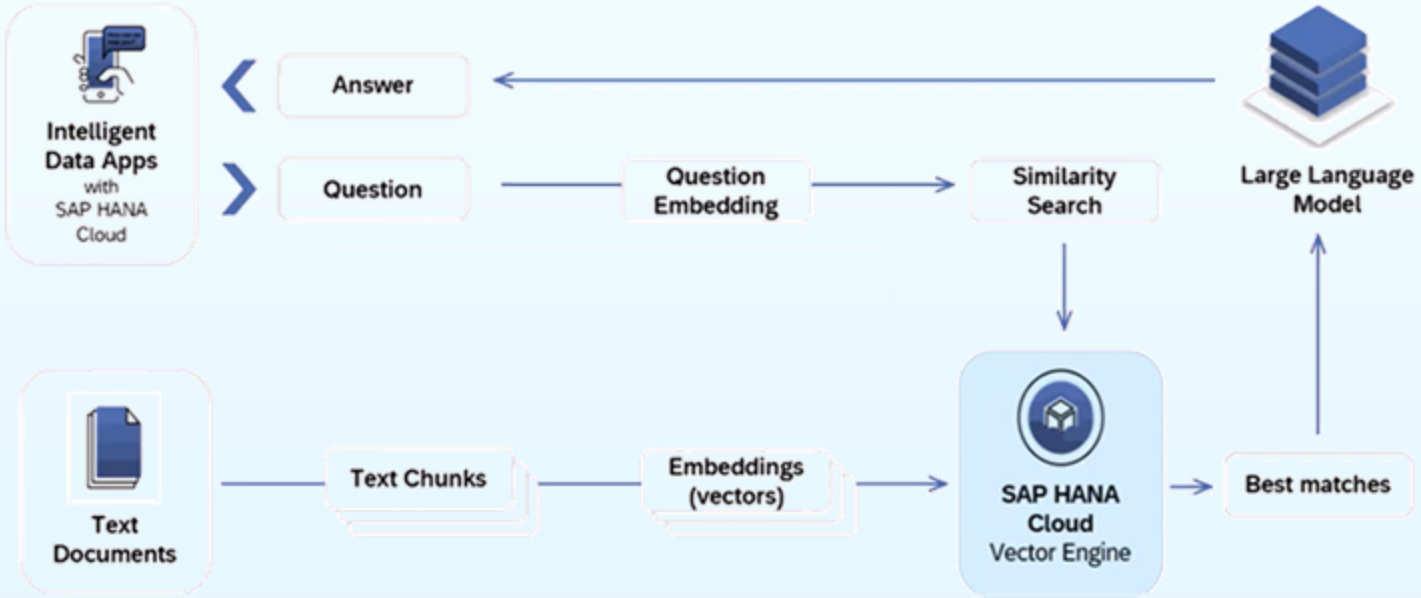
3 Key Choices and Guidelines

4 Implementation

# Vector-based RAG Process Flow

In order to leverage a vector-based RAG technique, there are two key process steps:

- Embedding Creation - In this step, you convert the knowledge base to embedding and store them in a vector databases.
- Query Pipeline – In this step, you create question embedding, perform similarity search on a vector database and provide additional context to LLMs for response creation.




## Expected Outcome

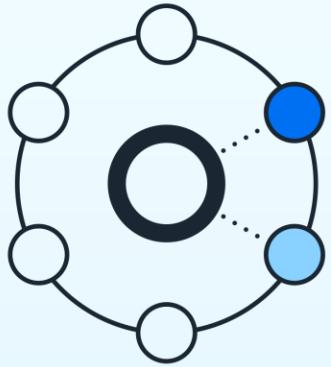
Help LLMs to provide more accurate and contextually relevant answers
Human like interaction improves user experience
Increase trust and reliability as responses are grounded on authentic knowledge sources.

# Key Benefits

Why leverage vector-based RAG?

## Contextual Relevance

**With better context awareness, responses are likely to match intent and meaning**



## Dynamic Adaptability

User has flexibility to ask open-ended or unstructured questions

## Reduced Hallucinations

**With Semantic matching on reliable knowledge source reduce hallucinations**

# Use cases

RAG technique extends capability of LLMs to several scenarios

- **Question Answering Systems**
  RAG technique powers Q&A systems that deliver accurate and up-to-date responses, especially in domains with frequently changing information.

- **Chatbots and Virtual Assistants :**
  RAG-powered chatbots can provide more helpful and informative responses, drawing on relevant information from knowledge bases

- **Code Completion and Suggestion Systems :**
  RAG can enhance code suggestion systems by incorporating context-specific knowledge from code repositories or documentation

# Key Concepts

- **Vector Similarity Search**

  The process of finding the most relevant pieces of content based on closeness in vector space (using similarity metrics like cosine similarity).

- **Context Window**

  The amount of retrieved content the LLM can “see” when generating a response.

- **Hybrid Search:**

  Combining vector search with keyword search for even better accuracy.

- **Data Masking**

  A technique used to protect sensitive information by anonymizing or pseudonymizing data inputs, ensuring privacy and compliance.

- **Templating**

  Templating is feature of the orchestration layer in Generative AI Hub that allow for the creation of structured prompts and the integration of domain-specific data to enhance AI model outputs.

# Pre-requisites

## Business

- SAP AI Core with the “Extended” tier on SAP BTP (Pricing Information)
- SAP HANA Cloud on SAP BTP (Pricing Information)
- SAP AI Launchpad (Pricing Information)

## Technical

- SAP Business Technology Platform subaccount (Setup Guide)
- SAP AI Core (Setup Guide)
- SAP HANA Cloud - Vector Engine (Setup Guide)
- SAP AI Launchpad (Setup Guide)

## SAP Business Technology Platform (SAP BTP)

- SAP Business Technology Platform (BTP) is an integrated suite of cloud services, databases, AI, and development tools that enable businesses to build, extend, and integrate SAP and non-SAP applications efficiently.

## SAP HANA Cloud

- SAP HANA Cloud is a database as a service that powers mission-critical applications and real-time analytics with one solution at petabyte scale. Use relational, property graph, spatial, vector, and semi-structured data along with embedded machine learning to power intelligent data applications.

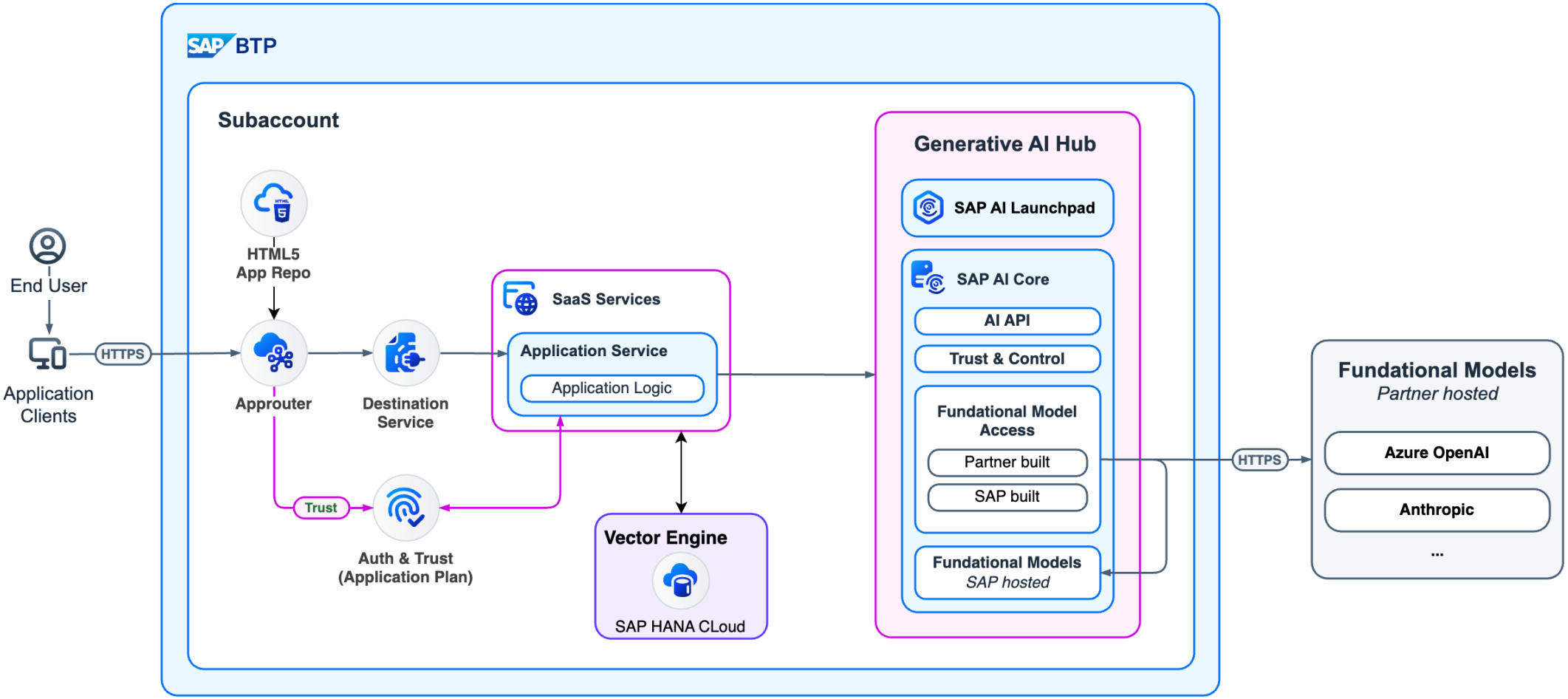
## SAP AI Core

- SAP AI Core is a managed AI runtime that enables scalable execution of AI models and pipelines, integrating seamlessly with SAP applications and data on SAP BTP that supports full lifecycle management of AI scenarios.

## SAP AI Launchpad

- SAP AI Launchpad is a multitenant SaaS application in SAP BTP. Customers can use SAP AI Launchpad to manage AI use cases (scenarios) across multiple instances of AI runtimes.

# High-level reference architecture



# Key Choices and Guidelines



## Similarity Metric

Similarity metric is a measure to determine how close two text embeddings are.

- Cosine Similarity is the most common for text embeddings
- Choose the metric best supported by embedding model or vector database

## Similarity Search

- Different retrieval technique (e.g. Query Expansion, reranking, hybrid search etc.) can be employed to fetch the accurate and relevant chunks.

## LLM Prompt Design

Use effective prompt design to further improve LLM response

- Include retrieved context clearly (e.g., “Based on the following documents…”).
- Provide instructions or constraints (e.g., “Answer using only the context below.”)
- Use separators (--- or XML tags) to help the LLM distinguish source material from the prompt.

# Key Choices and Guidelines



## Retrieval Parameters and Context Size

While retrieving from vector database, it may result in multiple chunks related to the user query.

- You may start with top_k = 3–5, and tune based on output quality.
- You may consider relevance scoring thresholds to avoid retrieving weak matches.
- Context size is a trade-off between information coverage vs no. of tokens (also cost).

## Evaluation Strategy

- Use retrieval metrics (e.g., precision@k, recall@k) and generation metrics (e.g., ROUGE, BLEU, factual consistency).
- Gather qualitative feedback from users (“Does the answer feel grounded?”)

# Implementation

Programming Model reference to implement vector embeddings

## Python

**SDK**

- SAP Generative AI hub SDK (For building apps)
- SAP AI Core SDK and AI API Client SDK (AI Core lifecycle)
- HANA_ML SDK

**Reference Code**

- Vector RAG - Query Pipeline

**Learning Journeys**

- Predictive AI with SAP AI Core
- RAG with HANA Vector Engine

## JavaScript/TypeScript

**SDK**

- SAP Cloud SDK for AI

**Reference Code**

- SAP Cloud SDK for AI - Sample Code

**Learning Journeys**

- There are currently no learning journeys using the official SDK.

## CAP App

**SDK**

- SAP Cloud SDK for AI (Recommended)
- CAP LLM Plugin

**Reference Code**

- SAP Cloud SDK for AI - Sample Code

**Learning Journeys**

**Recommended**

- There are currently no learning journeys using the official SDK.

**Other**

- GenAI Mail Insights: Develop a CAP application using GenAI and Retrieval Augmented Generation (RAG).

# Code Sample

Python

```python
from gen_ai_hub.proxy.langchain.init_models import init_embedding_model, init_llm
# Initialize the embedding model
embedding_model = init_embedding_model(model_name="your_embedding_model_name")
# Initialize the language model

llm = init_llm(model_name="your_llm_name")
# Example function to perform RAG
def perform_rag(query, context):
  # Retrieve relevant information using the embedding model
  retrieved_info = embedding_model.retrieve(query, context)
  # Generate response using the language model
  response = llm.generate(query, retrieved_info)
  return response
# Example usage
query = "What is the impact of data masking in AI?"
context = "Data masking is a technique used to protect sensitive information..."
response = perform_rag(query, context)
print(response)
```

# Contributors


Gupta, Chirag


Behera, Bhagabat Prasad


Marques, Luis


CIGAINA, MARCO

RF
Robledo, Francisco


Antonio, Dan

HI
Humphries, Ian

# Thank you