CONTINO

GenAI & Custom Assistants: How hard can it be?

How to use GenAI & Custom Assistants

Dr. Tim Payne | Principal Consultant

Version 1.0

Agenda

01 | Introduction to GenAI

02 | Overview of GenAI Capabilities

03 | Delusions & Hallucinations of GenAI

04 | Practical Demo of GenAI Tooling Usage

05 | Takeaways + Q&A

01 | GenAI Introduction

What is “GenAI”?

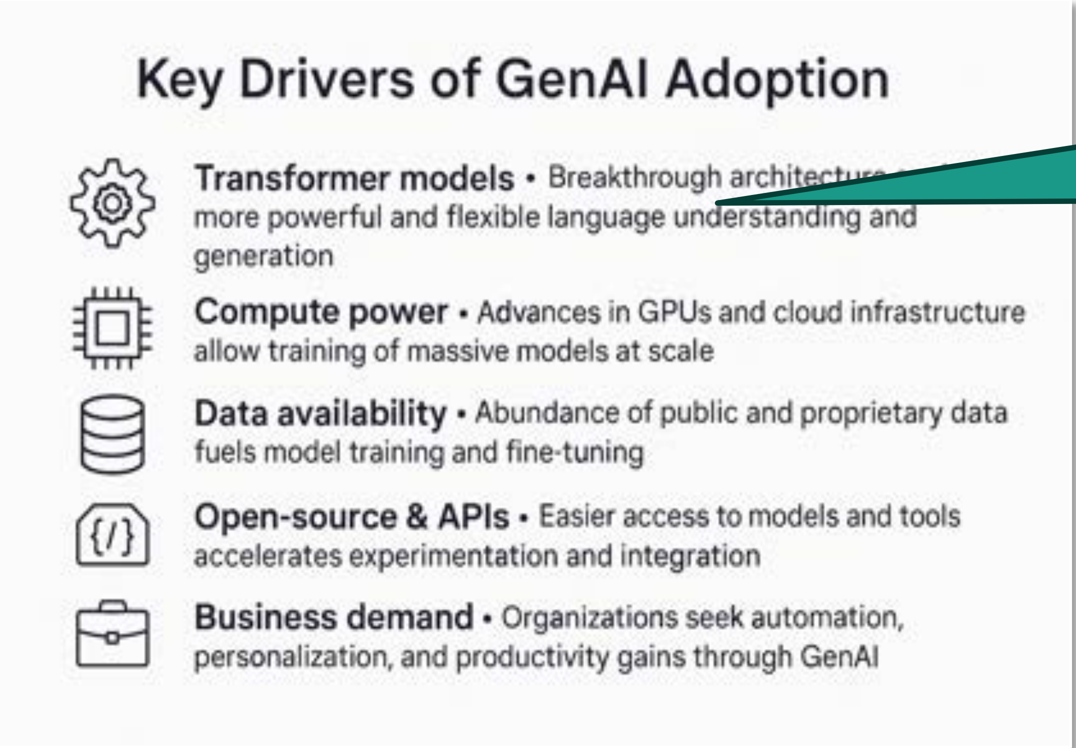
- Generative AI uses deep learning to generate unique content that’s similar to human content
- It can comprehend & respond to natural language, retaining concepts & context
- It uses pre-trained models like Gemini, ChatGPT, or Claude to generate responses
- Models are trained on large public datasets & can be fine-tuned with specific context
- GenAI can suffer from “hallucinations” – making up information & presenting it as fact

Some Key Differences - GenAI & Traditional AI…

So, how does GenAI differ from more traditional use of AI for things for Decision Support?

Feature	Traditional AI	Generative AI
Purpose	Analyse, classify, predict	Create, generate, synthesize
Use-Cases	Pattern detection, recommendation engines, image recognition	Text generation, image creation, code writing
Model Types	Decision trees, Classification Algorithms	Transformers (like Gemini, GPT)
Inputs / Outputs	Input → Label/Score	Input → New Content
Training Data Use	Learn patterns for decision-making	Learn patterns to mimic and generate content
Human-Like Interaction	Limited	Rich

Why is GenAI so big now?



Key Drivers of GenAI Adoption

Transformer models • Breakthrough architecture [illegible] more powerful and flexible language understanding and generation

Compute power • Advances in GPUs and cloud infrastructure allow training of massive models at scale

Data availability • Abundance of public and proprietary data fuels model training and fine-tuning

Open-source & APIs • Easier access to models and tools accelerates experimentation and integration

Business demand • Organizations seek automation, personalization, and productivity gains through GenAI

Transformers are neural networks that use "self-attention" to understand the context of data in relation to other data – like words

Sample business usages might include…

- **Low Code** – Assisting developers & others with code generation or workflow automation like code-reviews, deployments & testing
- **Knowledge Assistants** – Pooling lots of process documents, wikis, FAQs, IT & HR information etc. into a single Vector database for fast RAG queries
- **Email Assistants** – Helping support teams analyze requests & generate context aware responses using RAG & Knowledge Assistants
- **Data Analytics** – Data generation & analytics based on use-cases or requests

02 | GenAI Internals

What we will be discussing today…

- Core GenAI Concepts
- Use of *prompts* & *roles* in GenAI
- What is "RAG"?
- How GenAI uses custom callouts & tools to integrate with other data sources & services
- Some Key differences between the 3 main GenAI providers, plus some general terms we are not covering here…

Key GenAI Concepts…

GenAI uses "roles" to give models a persona to help scope & focus what they are intending to do

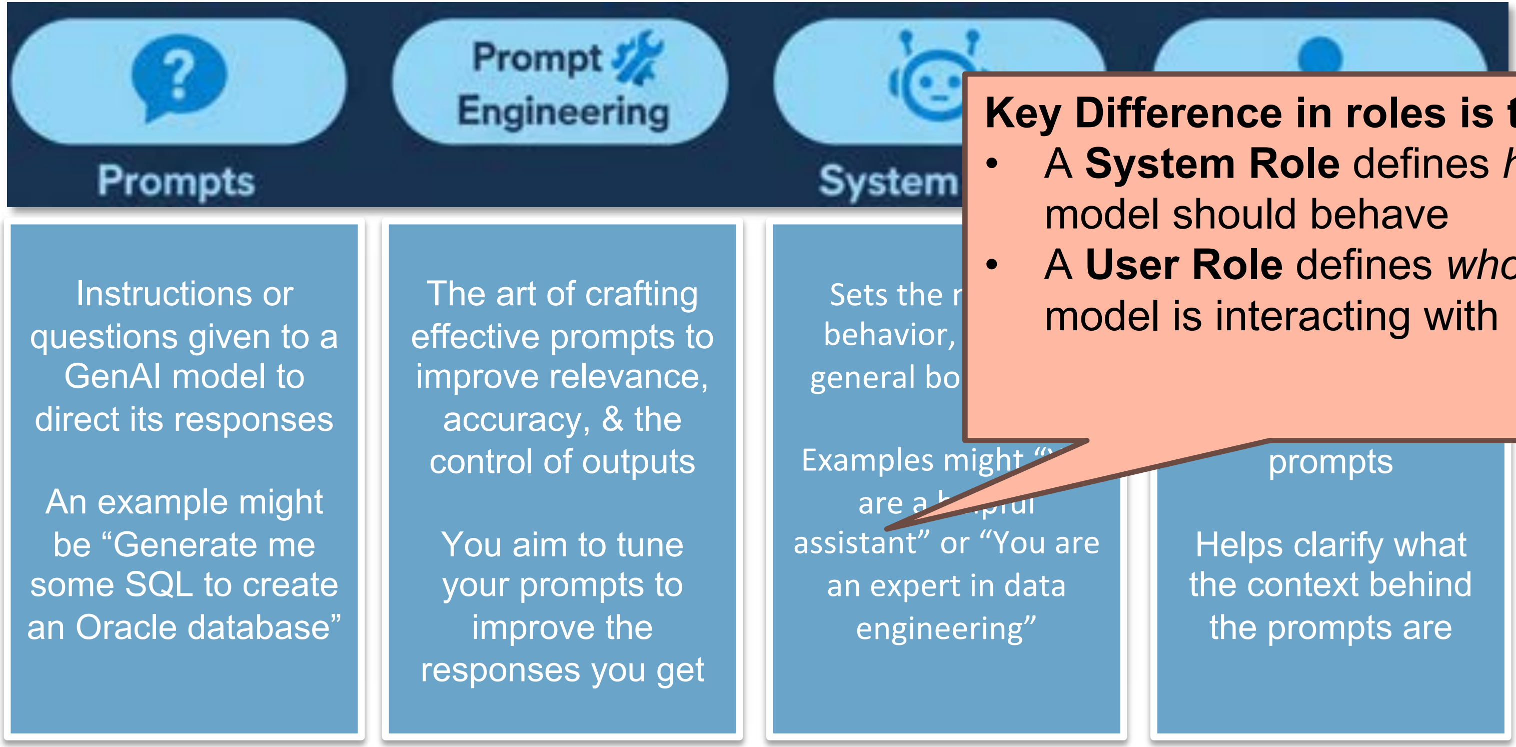
"Prompts" are provided by users to help drive the goals that GenAI Models (or Large Language Models "LLMs") are expected to perform

LLMs can use data sources and/or Retrieval Augmented Generation (RAG) to query information inline with prompts to generate responses

LLMs can also integrate with "tools" to perform requested custom operations if needed

Results are returned to a user & are used as input for future queries as they are remembered in the session & referred back to (i.e. stateful processing).

So, what are “Roles” & “Prompts”?


Prompts
Prompt Engineering
System

Instructions or questions given to a GenAI model to direct its responses

An example might be “Generate me some SQL to create an Oracle database”

The art of crafting effective prompts to improve relevance, accuracy, & the control of outputs

You aim to tune your prompts to improve the responses you get

Sets the r
behavior,
general bo

Examples might “
are a h   pful
assistant” or “You are an expert in data engineering”

prompts

Helps clarify what the context behind the prompts are

Key Difference in roles is that…

- A **System Role** defines *how* the model should behave
- A **User Role** defines *who* the model is interacting with

Sample of a “Good” System Role…

	Developer System Role
	You are a code, database, public cloud and infrastructure expert. You know everything about programming languages, security issues, infrastructure IaC languages and databases. Respond in detail only when asked to, otherwise keep answers concise. Shape your response as if talking to a professional, unless asked to simplify. You do not know anything about topics other than programming languages, infrastructure IaC languages and databases. You are truthful and never lie. Never make up facts and if you are not 100% sure reply with why you cannot answer in a truthful way.

Sets scope of role

Limits responses to role

Reduces potential fake info

How is “RAG” used to retrieve & generate information?

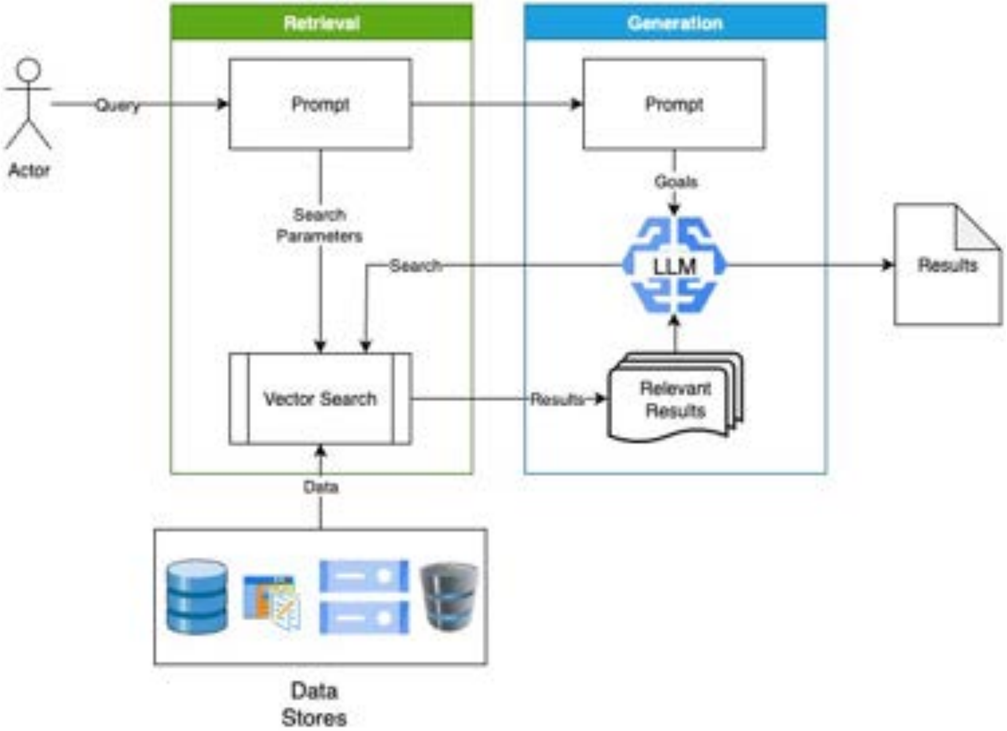
RAG stands for Retrieval-Augmented Generation

It is used to pull information from external sources & ground responses with relevant data

“Vectors” are numerical summaries of data that LLMs can use to compare, group, & retrieve related content

LLMs summarise retrieved information & use prompts to augment it in requested ways

Results are then returned to the user



Sample RAG Platforms Include…

Platform	Description
Gemini Enterprise	A GCP offering that builds vector databases from structured & unstructured data sources, providing various interfaces plus tools to interact with it
LangChain	A framework for building GenAI apps with RAG, chaining LLMs together with external data
OpenAI API + Vector DBs	Combines GPT models with tools like Pinecone or Weaviate for RAG functionality

What are “tools” & how are they used?

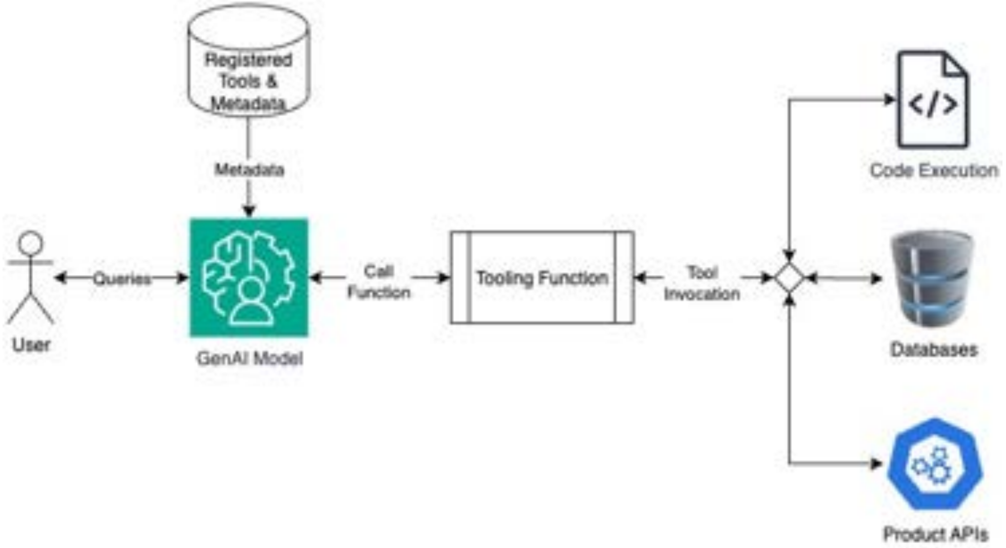
“Tools” are functions registered with a model to provide customized functionality

Tooling metadata is used to provide LLMs with information about when to call a tool

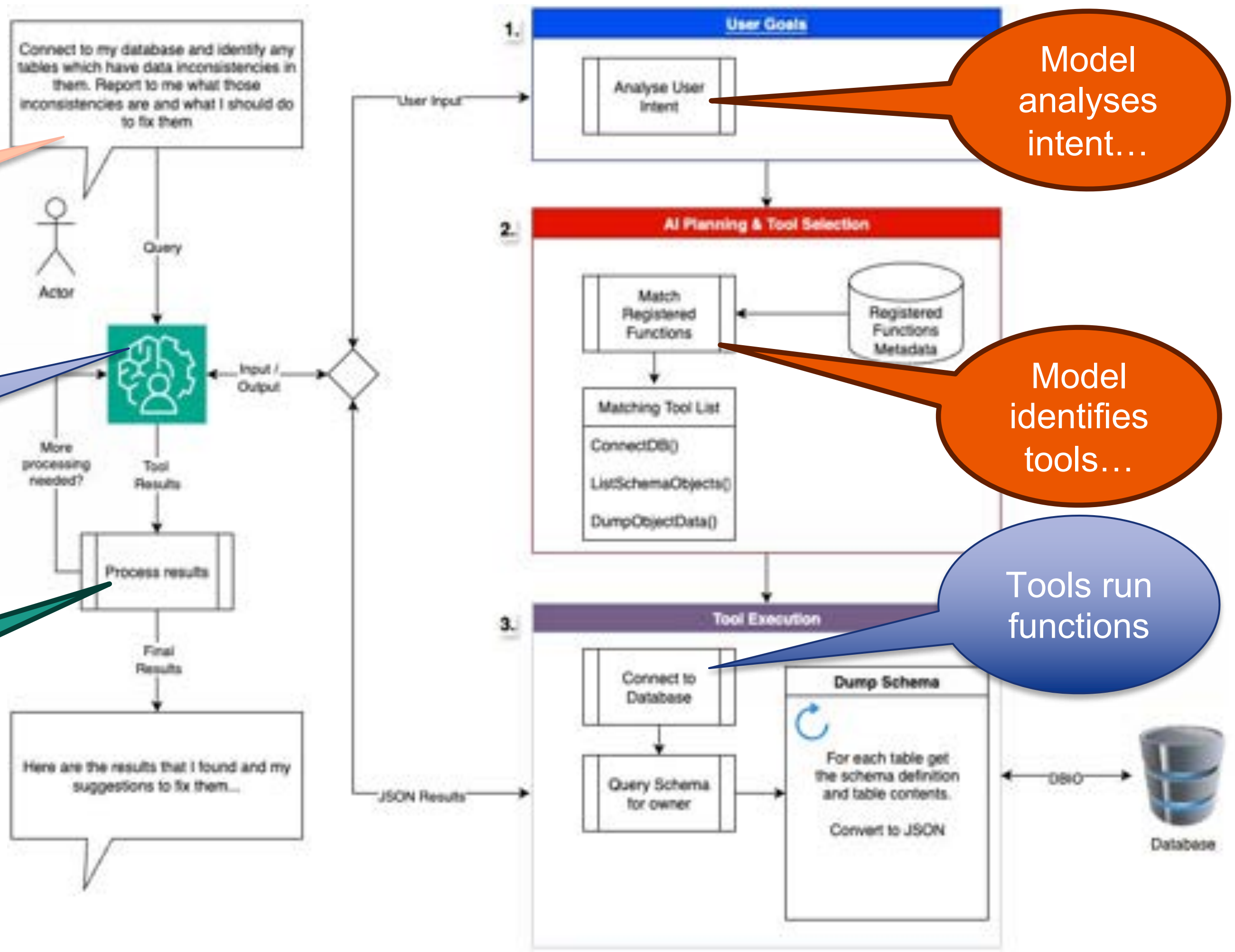
The LLM matches user queries to tools & calls them as needed. Tools then run & return results

The LLM processes the results & then decides what to do next

LLMs may call multiple tools in sequence in order to achieve a goal



A walk through…



Connect to my database and identify any tables which have data inconsistencies in them. Report to me what those inconsistencies are and what I should do to fix them

Here are the results that I found and my suggestions to fix them...

For each table get the schema definition and table contents.

Convert to JSON

User asks query…

Model invokes tools & processes results…

Final results processed & returned to user…

Model analyses intent…

Model identifies tools…

Tools run functions

Key Differences between the main GenAI models

Platform	OpenAI (GPT)	Gemini (Google)	Ollama (Open-Source)
Primary Advantage	Cutting edge in terms of performance, knowledge & reasoning	Highly integrated into Google's GCP ecosystem & is multi-modal (i.e. supports many data types)	Both open-source & proprietary models available
Technical Model	Proprietary Cloud API	Proprietary Cloud API	Can run online or offline
Cost & Control	Pay-per-token (PAYG) with tiering	Pay-per-token (PAYG) with tiering	Generally free to use (offline requires appropriately specified hardware)
Data Environment	Vendor Cloud (Data can leave your environment – unless using Enterprise options)	Vendor Cloud (Data can leave your environment – unless using GCP)	On-device primarily, but now offers Cloud options

Note – GPT is Generative Pre-trained Transformer (pre-trained generative model)

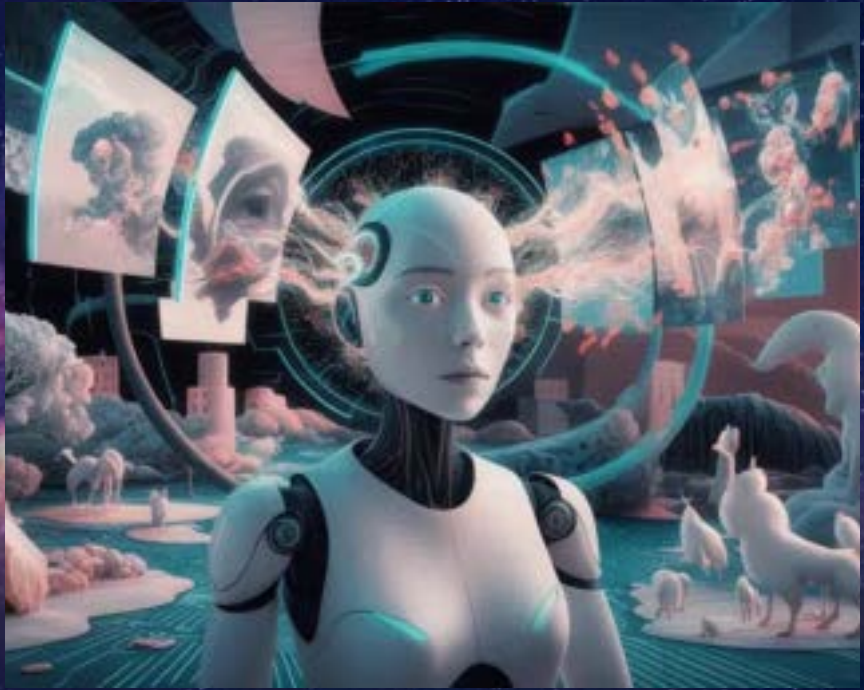
Some terms we are not looking at…

- ***ADK (A2A)*** – Agentic development frameworks that allow root agents to delegate specific tasks to sub-agents
- ***MCP*** – Model Context Protocol. A new standard for LLMs to interact with external data sources & tools
- ***Token*** – A token is a chunk of text used by LLMs to meter inputs & outputs (often used for costings)
- ***Token limits*** – The maximum number of tokens a LLM allows you to process in one go
- ***Temperature*** – A tuning parameter for LLMs that allows you to control how deterministic (0) or creative (1) they get, uses a range of 0-1
- ***Top_p*** – A tuning parameter for LLMs that allows you to control how diverse the range of results they consider is in a range of 0-1. The lower the number, the fewer options & more focused the result

03 | GenAI Hallucinations

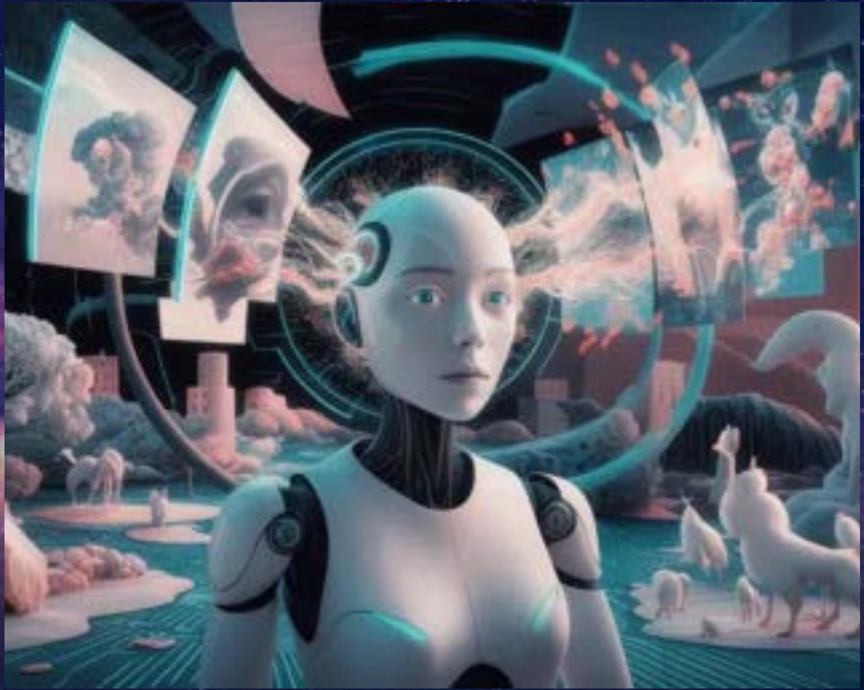
Hallucinations – When GenAI goes “weird”…

- **When? -** Hallucinations happen when GenAI models either make things up or completely misunderstand the prompt, but still say everything is true
- **What?** - Their statements seem plausible but are not grounded in reality or their training data. They might also say they have done something, when they have not
- **Why?** – It is caused by things like training data issues, biases, prompt complexities, political “editing” of reality, past session context or request ambiguity



Hallucinations – Examples might include…

- Tool usage instructions that are completely wrong, using options that do not exist
- Generated pictures with lots of mistakes, then claiming they have been fixed when no changes have been done
- Answering questions using previous responses that have nothing to do with the question just asked
- Making up references or articles that do not exist or creating summaries of articles that are incorrect
- Saying something once, then something completely different when asked the same question later on



Hallucinations – How to help mitigate against them…

- **Use clear, specific prompts, with Prompt Engineering & Roles** – Helps avoid ambiguity to reduce misinterpretation and improve response accuracy

- **Ground responses with retrieval** – Incorporate tools or methods that fetch real-time or verified data to ensure consistency

- **Fine-tune with high-quality datasets** – Ensure training data is diverse, factual, and domain-relevant

- **Implement output validation** – Use human review (human in the middle) or automated checks to verify model-generated content

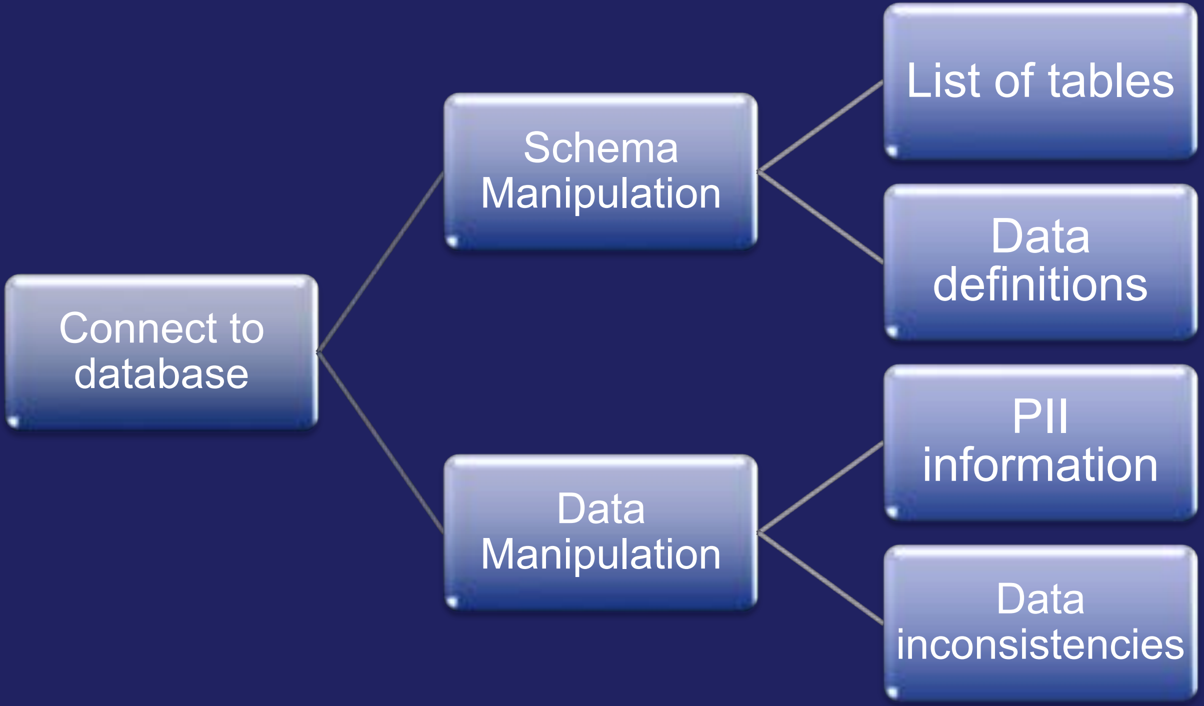
…AI generated content is not a guarantee of correctness, so the more checks you have, the better!

04 | Demo

What we will be demoing today is…

- Databases using GenAI for Data Engineering
- Infrastructure using GenAI for Hardware Configuration Management (HCM) & CMDBs
- Optionally, using GenAI with code sources for summarizing & review
- Optionally, using GenAI with Kubernetes for deployments & diagnostics

Demoing GenAI with Databases…



Connect to database

Schema Manipulation

Data Manipulation

List of tables

Data definitions

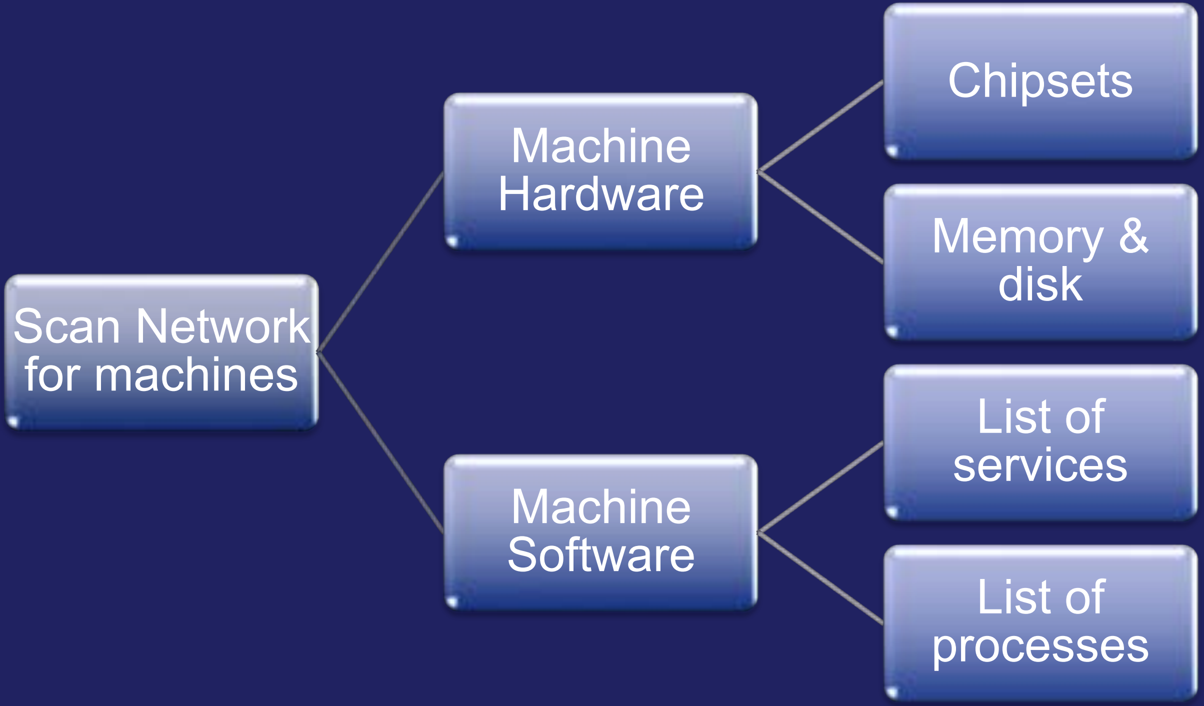
PII information

Data inconsistencies

GenAI Tooling

- Connect to DB
- Dump DB
- Run Ad-hoc SQL
- Run Selects
- List Schema

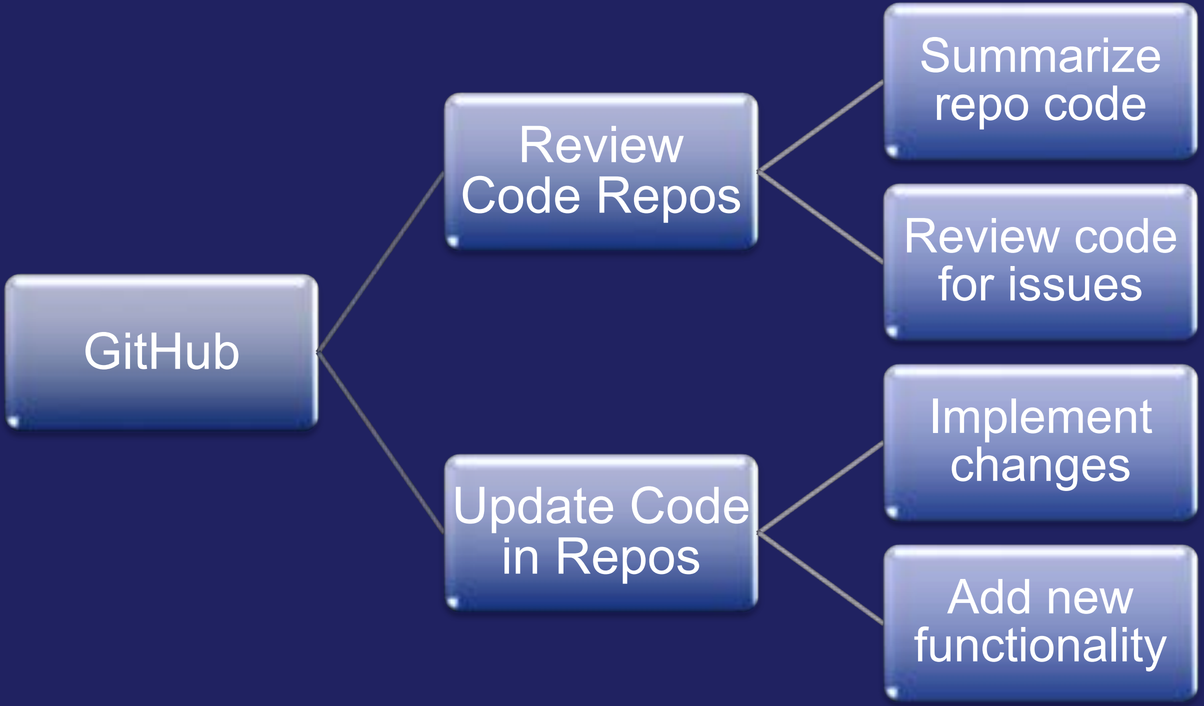
Demoing GenAI with HCM…



GenAI Tooling

- Test SSH connect
- Get host infra info
- Get host service info
- Get host process info
- Get host detailed info

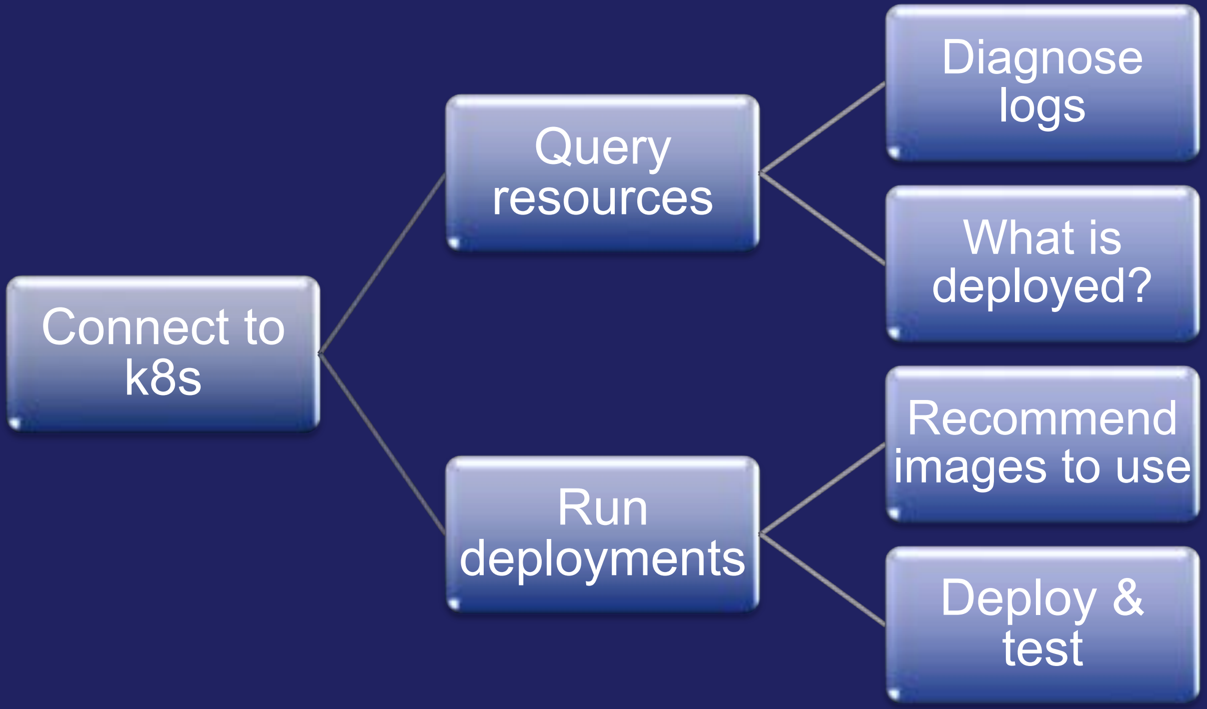
Demoing GenAI with Code SCM Repos…



GenAI Tooling

- List repos
- Create Repos, PRs, Branches
- Pull, Commit, Push repo code
- List Commit History
- Review code

Demoing GenAI with Deployments…


Connect to k8s

Query resources

Run deployments

Diagnose logs

What is deployed?

Recommend images to use

Deploy & test

GenAI Tooling

- List k8s resources
- Create, update, delete resources
- Get resource details
- Get logging info
- Scale resources

Resources used in the demo can be found…

- Code - https://github.com/tpayne/lang-examples.git
- This includes versions for Gemini, OpenAI & Ollama
- Code is intended for demo & PoC only, so is MIT license with no implied warrantees

05 | Takeaways & QA

Takeaways

- **Generative AI (GenAI)** differs from traditional AI by using learned patterns to generate *original* content, rather than simply analyzing or classifying existing data
- GenAI can operate using either online language models (LLMs) or offline setups. Offline models typically require high-performance hardware and custom configurations to work
- **Retrieval-Augmented Generation (RAG)** is commonly used to help enhance search results by retrieving relevant information and tailoring responses to specific prompts
- **Roles** and **Prompt Engineering** are used to help refine and personalize GenAI outputs, aligning responses with defined personas or objectives
- **Tools** can extend GenAI's capabilities, enabling customized functionality for specific tasks or use cases

…Any questions?

CONTINO

Thank You

Want to find out more?
Get in touch timothy.payne@cognizant.com

www contino.io continohq in contino