1. Conversational AI on Vertex AI and Dialogflow CX

Description:

In this course you will learn how to use the new generative AI features in Dialogflow CX to create virtual agents that can have more natural and engaging conversations with customers.

Discover how to deploy generative fallback responses to gracefully handle errors and omissions in customer conversations, deploy generators to increase intent coverage, and structure, ingest, and manage data in a data store.

And explore how to deploy and maintain generative AI agents using your data, and deploy and maintain hybrid agents in combination with existing intent-based design paradigms.

Objectives:

- Articulate and explain the functionality of the new generative AI features in Dialogflow CX.
- Implement generators to increase intent coverage in customer conversations.
- Enable generative fallback responses to gracefully handle errors and omissions in customer conversations.
- Deploy and maintain Generative AI agents that use your data.

1.1. Course Introduction

Discover what is covered in the Conversational AI on Vertex AI and Dialogflow CX course, including the target audience, prerequisites and the agenda, before refreshing your understanding of Dialogflow CX key concepts and terminology.

1.1.1. Video - Course Introduction

- YouTube: Course Introduction

Conversational AI on Vertex AI and Dialogflow CX

Welcome to this course on Conversational AI on Vertex AI and Dialogflow CX. This course is designed for developers and conversational designers, and builds upon concepts learned in the course "Customer Experiences with Contact Center AI". If you haven't already, please review that course before proceeding.

Prerequisites and Background Information

If you are familiar with Dialogflow CX but need a refresher, there's a short review available. Having a good understanding of Google Cloud fundamentals is also helpful, as there is a course on this topic in Cloud Skills Boost.

Course Objectives

In this course, you'll learn:

- About Vertex AI Conversation, a single powerful platform for building conversational AI solutions that use Generative AI.
- How to integrate the new generative AI features of Dialogflow CX and the benefits they bring.
- Generative AI Agents, Data Stores, Generators, and Generative Fallback, including how to configure and deploy generators and enable generative fallback in your Dialogflow CX solution.
- When to use intent-based flows, generative AI, or a mix of both for certain use cases.
- How to enable Conversational AI features in your Dialogflow CX solution, with costs information for both standard and enterprise customers.

1.1.2. Video - Dialogflow CX recap

- YouTube: Dialogflow CX recap

Dialogflow CX Recap

This lesson provides a recap of Dialogflow CX, which is covered in full in the "Customer Experiences with Contact Center AI" course. Dialogflow CX enables you to create virtual agents for dynamic and personalized 24/7 self-service.

Key Features of Dialogflow CX

Dialogflow CX offers several key features, including:

- An easy-to-understand interface using a state-machine based model and flow-based modules optimized for large enterprise customers.
- Handling detours by answering customer follow-up questions at any point and seamlessly transitioning the customer back to the dynamic flow.
- Building and deploying agents in multiple languages.
- Automatically running tests and experiments on the agent before deployment and A/B testing during production, with integrated CI/CD.
- Analyzing conversational paths, assessing agent performance, and improving the agent.
- Native Interactive Voice Response (IVR) features such as DTMF, timeouts for the agent to react if the user didn't respond for some time, and Barge-In.

Pre-built Components and Benefits

Dialogflow CX includes pre-built components that allow you to transform the user experience and optimize contact centers through:

- Building faster by starting training with only a few examples and leveraging over 40 pre-built agents.
- Engaging your audience more efficiently with built-in natural language understanding.
- Utilizing multiple fulfillment options, with natural, conversational responses through generative AI.

Dialogflow CX User Interface

The Dialogflow CX UI offers a state diagram overview of your virtual agent logical flows. The interface includes several tabs:

- Build: While working on your agent, you'll have the Build tab enabled much of the time.
  - FLOWS section: "Default Start Flow" is all you need unless you have a complex virtual agent.
  - Manage tab: While working on other resources such as intents, entities, and webhooks.

The Manage tab includes features such as:

- Graph settings to change your view of the agent configuration.
- Task indicator to show whether any agent is training or another task is in progress.
- Agent settings and test agent buttons.

1.1.3. Video - Dialogflow CX recap: Key terms

- YouTube: Dialogflow CX recap: Key terms

Dialogflow CX Recap: Key Terms

Let's recap on some key terms used to describe certain concepts, functionality, and objects in Dialogflow CX.

Key Concepts

- **Head Intent**: The main topic of conversation.
- **Flows**: The overall sequence or block of instructions.
- **Pages**: Specify what to do in specific states of the conversation.
- **Routes**: Where to go next when the state of the conversation changes.

Page Configuration

A page is composed of elements related to the state of the conversation. It's where you tell Dialogflow CX what to say and do, and where to go next. Pages can include:

- **Entry Dialogue** (or agent prompt): What to say to the user.
- **Parameters**: Something that should be collected from or played back to the user.

Transition Routes

Transition routes are used to specify where to take the conversation next. They are used to control whether Dialogflow should follow a configured route based on either a matched intent or condition that evaluates to true.

State Handlers vs Event Handlers

- **State Handlers**: Address the virtual agent's expected behavior to cover all the services offered by your virtual agent.
- **Event Handlers**: Address how the virtual agent will behave when anticipated routes aren't followed, and provide ways for the virtual agent to course correct so the user isn't abandoned.

Fulfillment

In Dialogflow CX, fulfillment has a broader scope than in Dialogflow ES. It includes prompts, parameters, webhooks to read from or write to an external source, and event handlers to handle no-input scenarios.

Each page in Dialogflow CX is designed to be independent and can transition to another flow or page based on the configured route. This allows for flexibility and customization in designing conversational flows.

1.1.4. Video - Dialogflow CX recap: Entities and Intents

- YouTube: Dialogflow CX recap: Entities and Intents

Dialogflow CX Recap: Entities and Intents

A Virtual Agent conversation is made up of representative words and phrases, which include:

- **Intents**: The main topic of conversation.
- **Entities**: Representative words or phrases that describe specific information.
- **Parameters**: Specific values associated with entities.

To configure these, you'll need to follow a process similar to this:

Creating Entities

1. Click on the Manage tab in the Dialogflow CX menu.
2. Under Resources, click on Entity Types and then Create.
3. Give your new custom entity type a name (e.g., "tier").
4. Add values to your entity type (e.g., "silver", "gold", and "platinum").
5. Provide each entity value with synonyms.

6. Click Save to complete the entity type.

Creating Intents

1. Click on the Manage tab in the Dialogflow CX menu.
2. Under Resources, click on Intents and then + Create.
3. Give your new custom intent a name (e.g., "change-tier").
4. Provide training phrases that represent how customers frequently use the term to describe the action they want performed.
5. Click Save.

Form Filling or Slot Filling

When creating intents, you'll notice highlighted entities in the training phrases, referred to as form filling or slot filling. This is because Dialogflow recognizes these terms as already defined entity types.

Parameter ID and Entity Type

In intent definitions, you may see a section for Parameter ID, which refers to the parameter value (e.g., "tier"). The entity type referenced by Parameter ID will depend on whether the entity value can be in the form of a list (e.g., "Redact in log" for sensitive information).

By configuring entities and intents correctly, you'll enable your Virtual Agent to take on more human-like intelligence and provide better customer experiences.

1.2. Generative AI for Dialogflow CX

What is generative AI for Dialogflow CX? How does it work? When would I use it? Why is it useful?

Start your journey of discovery with generative AI enablement in Dialogflow CX.

1.2.1. Video - Introduction to Generative AI for Dialogflow CX

- YouTube: Introduction to Generative AI for Dialogflow CX

Introduction to Generative AI for Dialogflow CX

Dialogflow CX is now supercharged with generative AI features. With generative AI, you can help users find the exact answers they're looking for quickly using natural language queries.

Benefits of Generative AI for Dialogflow CX

- Provide rich, multimodal experiences using natural language

- Allow users to communicate expressing intent in natural language, instead of following a set of rigid questions
- Give users generic answers or summarizations from LLM capabilities

Characteristics of Generative AI for Dialogflow CX

- Informational: gets information from company documents and websites to provide information or gives generic answers or summarizations from LLM capabilities
- Responsibly controlled: allows you to define controls with flows and business logic, helping reduce risk and improve security, generate answers grounded in your content, and choose the degree of generated content that is presented to users

Goals and Benefits of Vertex AI Conversation

- Create virtual agents that can understand and respond to natural language in a human-like way
- Bring the conversation back on track with a single powerful platform for building conversational AI solutions that use Generative AI

Benefits of Using Vertex AI Conversation Features in Dialogflow CX

- A developer controls the user experience, unlike using a pure LLM approach
- Ensure chatbots provide accurate and helpful information, conversations remain on track, and generated content is restricted to what you want it for

Vertex AI Search and Conversation

- Can create virtual agents that can handle a wide range of tasks, from simple FAQs to complex transactions
- Blend Generative AI features within the user experience

Benefits of Google Cloud's Conversational AI Platform

- Scalability: deployed on cloud platform and scales elastically
- Security: managed solution deployed following standard security and data residency standards
- Reliability and resiliency: served with a SLA

Types of Virtual Agents in Dialogflow CX

- Informational: provide information to users
- Transactional: complete transactions with users
- Companions: anticipate end user needs within context and provide intelligent and useful propositions to help

End User Channels for Virtual Agents

- Text or chatbot: user interacts over text on a website or other text-based medium

- Rich Text: provides rich responses like chips, buttons, URL links, or answer summaries
- Voice: user interacts with their voice in real-time
- Custom or Multi-Modal: custom mobile application, rich web application combining text, voice, and image input

Natural Language Understanding (NLU) Technologies

- Intent based: highly deterministic intent-based matching based on training phrases
- State machine and Intent based: controls conversational paths deterministically through a state machine
- State machine and Intent based with Generative AI: handles complex natural language tasks with generative or summarization language features
- Instruction-based Control LLM and LLM-based NLU: using a framework such as ReAct, a virtual agent can be implemented by using instructions to achieve a task

1.2.2. Video - Generative AI additions to Dialogflow CX

- YouTube: Generative AI additions to Dialogflow CX

Generative AI Additions to Dialogflow CX

There are two ways to add generative AI capabilities to virtual agents in Dialogflow CX:

1. **Create a new Generative AI Agent using Vertex AI Conversations**: This involves creating a data store containing the content that the virtual agent will search, and then indexing it to answer questions about the data.
2. **Upskill existing Dialogflow CX agents by adding Vertex AI Search and Conversation data stores**: This allows for another layer of question coverage and enables the agent to respond to complex or unexpected customer inquiries.

Generative AI Agent Features

- Can crawl content from a URL, index it, and then generate responses to questions about the content
- Can auto-create bots from website or other business documents with minimal setup
- Uses Vertex AI Search & Conversation to create data stores from websites, structured, or unstructured data

Generative AI Capabilities

- **Generators**: can use Google's latest generative large language models during Dialogflow CX fulfillment to generate agent responses at runtime
- **Generative Fallback Responses**: can be configured with a text prompt to instruct the LLM on how to respond in case of unexpected customer inquiries or off-track conversations

Benefits and Applications

- Can handle complex natural language tasks, such as conversation summarization, sentiment analysis, and question answering
- Can provide answers to customer inquiries that don't match any intent routes or are outside the scope of existing Dialogflow CX solutions
- Can generate text for emails or conversation summaries using generators
- Can use generative fallback responses to handle off-track conversations in a friendly and natural way

Comparison to Intent-Based Routing

- Intent-based routing requires manual construction of every possible outcome during a customer conversation, which can be time-consuming and error-prone
- Generative AI enables the creation of more flexible and adaptable conversational flows that can respond to unexpected or complex customer inquiries

Key Use Cases

- **FAQs**: use generative AI to generate responses to frequently asked questions on a company website or internal documents
- **Complex Transactions**: use generative AI to handle complex transactions, such as processing documents or handling multiple inputs
- **Off-Track Conversations**: use generative fallback responses to handle conversations that have gone off-track or deviated from the intended route.

1.2.3. Quiz - Generative AI for Dialogflow CX Quiz

1.2.3.1. Quiz 1.

> **Important**
> **What are two of the main generative AI capabilities that have been added to Dialogflow CX? (Select two options)**
>
> √ Infobot
> √ Generative Search
> √ Generative AI Response
> √ Generators
> √ Generative AI Agent

1.2.3.2. Quiz 2.

> **Important**
> **Recently, some of your customers have been asking questions that your Dialogflow CX virtual agent is unable to answer. You realize that intents have not been defined for these types of questions. What generative AI feature would you add to your solution to handle conversations that have gone off-track?**

- √ Generative Fallback
- √ Generative AI Agents
- √ Generators
- √ Generative Playbooks

1.2.3.3. Quiz 3.

Important

You have been asked to add generative AI capabilities to your Dialogflow CX virtual agent that will enable it to reference the existing FAQs on the company website. What generative AI feature would you add to your solution to enable this functionality?

- √ Generative Fallback
- √ Generative AI Agent
- √ Generative Playbooks
- √ Generators

1.2.3.4. Quiz 4.

Important

What are two of the key benefits of using generative AI features in Dialogflow CX? (Select two options)

- √ Virtual agents can handle every type of conversation
- √ Chatbots provide accurate and helpful information
- √ Generated content is restricted to what you want it for
- √ Conversations will always remain on track
- √ You can create enterprise-scale Search applications

1.2.4. Link - How to build better conversational experiences with Generative AI

- How to build better conversational experiences with Generative AI launch

1.3. Generators

Explore generators on Dialogflow CX, with tunable prompts, parameters and model selection. You will learn how to define, configure and deploy generators in your Dialogflow CX solutions. Also, discover some use case examples and code-based responses.

1.3.1. Video - Generators on Dialogflow CX

- YouTube: Generators on Dialogflow CX

Generators on Dialogflow CX

The generator feature is a Dialogflow CX feature that allows developers to use Google's latest generative large language models and custom prompts to generate agent responses at runtime.

What are Generators?

- A Dialogflow CX feature that enables the use of generative large language models and custom prompts to generate agent responses.
- Can handle generic responses that involve general knowledge from a large textual dataset or context from the conversation.
- Enables the injection of dynamic, LLM-based responses into Dialogflow CX agents.

Benefits of Generators

- Use generative large language models during Dialogflow CX fulfillment.
- Pull context from a conversation (e.g., conversation summary).
- Add generated, personalized text or use generated text as input to a webhook (e.g., generated email or preset parameters).
- Chain LLM calls, enabling the building of fully generative virtual agents in Dialogflow CX.

Responsible AI

- Critical for generators: control what content is used to ground the bot in factual and relevant data.
- Put up guardrails on how the Generative AI Agent responds to users.

How do Generators work?

- Can call an LLM inline within Dialogflow CX.
- Perform multiple capabilities, such as:
  - Article Summarization
  - Conversation Summarization
  - Real-time customer sentiment analysis
  - Updating Structured Data Formats (e.g., JSON)
- Can use Google Cloud's large language models to dynamically make informed decisions based on the context of the conversation and knowledge base.

Example Use Case

- An LLM can assess the outcome of an eligibility quiz to make a decision on a user's progress, and then handle the conversation appropriately.

Use Cases for Generators

- Support complex customer inquiries with dynamic, personalized responses.
- Enhance conversational flows with real-time sentiment analysis or conversation summarization.
- Automate tasks such as article summarization or JSON updates.
- Build fully generative virtual agents in Dialogflow CX.

1.3.2. Video - Tunable prompt, parameters, and model selection

- YouTube: Tunable prompt, parameters, and model selection

Tunable Prompt, Parameters, and Model Selection

Generators can be customized and configured to generate dynamic responses or text that can be used during fulfillment.

Text Prompts

- A clear question or request is sent to the generative model at runtime.
- Can select which language model to use (e.g., text-bison or code-bison).
- Adjust LLM parameters (e.g., Temperature, Top P and Top K).

Prompt Structure

- Can contain:
  - Questions
  - Instructions
  - Contextual information
  - Examples
  - Partial input for the model to complete or continue

Customizable Text Prompts

- Fully customizable.
- Configure the text prompt sent to the generative model.

Example Prompt

- Instructing the LLM to be an expert at Text Summarization.
- Given an input text, summarize the information in 1 to 2 sentences.
- Think about the input content and do your best to provide a concise summary of the content for the user.

Contextual Prompts

- Mark words as placeholders by adding a $ before the word.
- Later associate these generator prompt placeholders with session parameters in fulfillment and replace them with session parameter values during execution.

Session Parameters

- Used to capture and reference values supplied by the end-user during a session.
- Have a name and an entity type.
- Unlike raw end-user input, parameters are structured data that can easily be used to perform some logic or generate responses.

Built-in Generator Prompt Placeholders

- $conversation - The conversation between the agent and the user, excluding the very last user utterance.
- $last-user-utterance - The last text input submitted by the user.

Model Selection

- Can choose a model (e.g., text-bison or code-bison).
- Adjust LLM attributes (e.g., Temperature, Top P and Top K).

Comparison to Vertex AI PaLM 2 for Text

- Dialogflow CX Generators use the same backend as Vertex AI PaLM 2 for Text.
- Similar features:
  - Define a prompt
  - Choose a model
  - Adjust LLM attributes
- Differences:
  - Parameters can be used and are automatically replaced within the tool in Dialogflow CX.
  - Vertex AI has more integrated features (e.g., Max responses, stop sequences, streaming responses).

Diagram

- Shows how Dialogflow CX leverages Vertex AI Foundation models to power the Generators feature.

1.3.3. Video - Generator examples

- YouTube: Generator examples

Generator Examples

This transcript provides examples of how to write prompts for different generator tasks.

Prompt Structure

- Instruction for the goal of the generator
- Text prefix with a prompt placeholder for any given text input
- Description of the format of the desired output

Example 1: Summarizing Content

- Instruction: Provide a concise summary of the text in 1 or 2 sentences
- Text prefix: Specify the Text prefix and prompt placeholder for any given text input
- Desired output: Format of the desired output (e.g., concise summary)

Example 2: Conversation Summary

- Instruction: Provide a concise summary of the conversation between a human and an AI
- Text prefix: Explicitly specify the prefixes of the conversation turns in the conversation history
- Desired output: Format of the desired output (e.g., concise summary)
- Example input:
    - Human: Hi, which models can I use in Dialogflow CX's generators?
    - AI: You can use all models that Vertex AI provides!
    - Human: Thanks, thats amazing!

Example 3: Question Answering with Self-Knowledge

- Instruction: Politely answer questions based on the provided information
- Text prefix: Specify the goal is to politely answer questions based on the provided information
- Desired output: Format of the desired output (e.g., concise answer)
- Example input:
    - User: What can I use in Dialogflow CX's generators?
    - Model: You can use all models that Vertex AI provides!
    - User: That's amazing!

Example 4: Handling Escalation to a Human Agent

- Instruction: Act as a polite customer service agent and handle requests from users to speak with an operator
- Text prefix: Specify the goal is to politely respond to the user about their request to speak with an operator
- Desired output: Format of the desired output (e.g., concise response)
- Example input:
    - User: I need to speak with an operator, please.
    - Model: I'm happy to help you. However, our policy is that all issues are handled by our automated system.

Example 5: Optimizing Google Search Query

- Instruction: Act as an expert at Google Search and use "Google Fu" to build concise search terms
- Text prefix: Specify the goal is to optimize the user's input query
- Desired output: Format of the desired output (e.g., optimized Google Search query)
- Example input:

- User: How can I find a job in my city?
- Optimized query: "job search [city]"

1.3.4. Video - Define, configure and deploy Generators

- YouTube: Define, configure and deploy Generators

Define, Configure and Deploy Generators

Now that we have introduced generators, let's explore how to get started with defining, configuring, and deploying them for your virtual agents.

Getting Started with Generators

1. In the Dialogflow CX Console, select your Google Cloud project and agent.
2. Select Generators in the Manage tab and click Create new.
3. Enter a descriptive display name for the generator.
4. Configure the text prompt, model, and controls.

Configuring Route Settings

1. When configuring the route for any page in a flow, enable generators in the Fulfillment section by selecting the generator you created.
2. Associate the generator prompt placeholders with session parameters.

Using Multiple Generators

1. You can add multiple generators to one fulfillment, which will be executed sequentially.
2. Each generator takes the output from the previous one as input.
3. This allows chaining multiple LLMs, making it easier for debugging specific steps.

Best Practices for Generator Configuration

- Use built-in placeholders in your generator prompt when possible.
- Associate generator prompt placeholders with session parameters to avoid issues with incomplete prompts.
- Use unique output names (e.g., `request.generative.destination`) for each generator's response.

Example Deployment Scenario

1. One generator provides information about destinations.
2. Another generator formats or summarizes the destination output to the user.
3. The two generators are chained together, providing a concise and clear response to the user.

Using Generative Output in Agent Responses

- Use the `request.generative.destination` variable as input for other agents or webhooks.
- Use the `request generative.output parameter` like any other session parameter in Agent Says, Webhooks, or Preset Parameters.

1.3.5. Video - Generators demo

- YouTube: Generators demo

Generators Demo

In this section, we'll be discussing a new dialogue feature called generators in Dialogflow CX.

What are Generators?

A generator is a single LLM (Large Language Model) that allows you to provide a text prompt for what you want it to do.

Creating a New Generator

1. In the Manage tab of your Dialogue FX agent, select Generators on the left-hand side.
2. Click Create new to create a new generator.
3. Provide a descriptive display name for the generator.

Configuring a Generator

1. Configure the text prompt for what you want the generator to do (e.g., provide a summary).
2. Apply the generator anywhere throughout your agent on a fulfillment (e.g., in a page).

Example Generator: Summarization Agent

Create a sample generator and say that you're an expert summarization agent.

- Provide a text prompt for what you want the generator to do: "Provide a summary of the current conversation in 1-2 sentences."
- Configure the generator to be polite and not tell any jokes.

Applying Generators to Routes

1. Select a page and find a fulfillment that you want to apply the generator on.
2. In the new generator section, select the generator you created.
3. Fill out the input and output sections:
    - Input: Provide a clear text prompt or use a session parameter (e.g., `request.generative.destination`).
    - Output: Specify what response you want the generator to provide.

Example Generator: Operator Handler

Create a generator called Operator Handler to describe a customer service agent that responds to user inquiries based on the last user utterance.

- Configure the generator to respond dynamically and politely.

Applying Generators to Routes (continued)

1. Select a page and find a fulfillment that you want to apply the operator Handler generator on.
2. In the new generator section, select the Operator Handler generator.
3. Fill out the input and output sections:
    - Input: Use the last user utterance as input for the generator.
    - Output: Specify what response you want the generator to provide (e.g., a summary of the conversation).

Using Generators for Summarization

1. Create a new route that uses a search intent.
2. Send a query to the Ser API and retrieve results.
3. Use a generator to summarize the information in a short snippet of text.

Using Generators for Dynamic Responses

1. Create a new operator intent on your start page.
2. Configure an Operator Handler generator to respond dynamically based on the last user utterance.
3. In the Agent says section, select the operator Handler generator instead of hardcoded responses.
4. Test the dynamic response with a user input (e.g., "This is crazy! I want to speak to a manager now").

The operator Handler generator responds dynamically based on the user's request and provides a concise summary of the conversation.

1.3.6. Video - Code-based responses with Generators

- YouTube: Code-based responses with Generators

Code-Based Responses with Generators

The Dialogflow CX generators feature allows you to generate code based on natural language descriptions in agent responses at runtime.

Powered by PaLM 2 for Text and Code-Bison

Generators are powered by PaLM 2 for Text, with the text-bison language model, and the code-bison code generation model.

Configuring Generators for Specific Purposes

You can configure generators for specific purposes, such as:

- Code generation
- Unit test generation
- Code fixing
- Code optimization
- Code translation

Important Considerations

When using generators to generate code, please note that:

- This does not replace human involvement.
- Code output by a generator should be comprehensively tested before the solution is used in production.
- Generated code is not intended or designed to be a replacement for code development.
- You should not use generated code to implement solutions for sensitive industries, such as cybersecurity and hacking prevention.

Generators Can Answer Code-Based Questions

Generators can answer code-based questions, specifically:

- Code generation
- Unit test generation
- Code fixing
- Code optimization
- Code translation

Code Generation Models Support Multiple Coding Languages

Code generation models support multiple coding languages, including:

- Go
- Google Standard SQL
- Java
- JavaScript
- Python
- TypeScript

Creating a Generator for Code Generation

To create a generator for code generation, simply select the code-bison model when configuring the model for your generator.

Example Prompt for Code Generation

The prompt is written in a similar way to how you would write a prompt for conversation. Here's an example:

- First, state the goal of the prompt: "Write code in [programming language] to solve [problem]."
- Next, specify the problem to solve by using a placeholder for the problem parameter.
- Then, specify the programming language by using a programming language parameter placeholder.
- Before asking for the solution from the generator.

This way, whatever problem and programming language that the user specifies in their natural language description input, the generator will be able to provide an answer in code that attempts to solve the problem.

1.3.7. Quiz - Generators quiz

1.3.7.1. Quiz 1.

> **Important**
>
> **Which two language models can a Generator use to generate code? (Select two options)**
>
> - √ codechat-bison
> - √ code-bison
> - √ code-gecko
> - √ text-gecko
> - √ text-bison

1.3.7.2. Quiz 2.

> **Important**
>
> **Generators have a particular set of capabilities that can be utilized by a Dialogflow CX virtual agent in a customer conversation. What are two capabilities of Generators? (Select two options)**
>
> - √ Accessing APIs and tools
> - √ Sentiment analysis of customer responses
> - √ Handling No Match errors
> - √ Conversation Summarization
> - √ Accessing information in data stores

1.3.7.3. Quiz 3.

> **Important**
>
> **With generators, you can make a prompt contextual by marking words as placeholders by adding a** *beforetheword.Theseplaceholdersusuallyholdapositionintheprompttthatwillbesubstit* **text prompt placeholder?**
>
> - √ @text
> - √ Any of the session parameters identified in the Intents
> - √ @last-user-utterance
> - √ @conversation

1.3.7.4. Quiz 4.

> **Important**
>
> **What is the correct syntax for a Generator's output parameter?**
>
> - √ $response.generator
> - √ $request.generator
> - √ $response.generative
> - √ $request.generative

1.3.7.5. Quiz 5.

> **Important**
>
> **Read the text prompt below and answer the following question. What information would the highlighted $conversation placeholder parse into the prompt?Your goal is to summarize a given conversation between a Human and an AI.**
>
> Conversation:
>
> ```
> 1  $conversation{USER:"Human:" AGENT:"AI:"}Human: $last-user-utterance
> ```
>
> **A concise summary of the conversation in 1 or 2 sentences is:**
>
> - √ The last text input submitted by the user.
> - √ The conversation between the agent and the user, including the very last user utterance.
> - √ The conversation between the agent and the user, excluding the very last user utterance.
> - √ It would not parse anything into the prompt as there is a mistake in the syntax

1.3.7.6. Quiz 6.

> **Important**
>
> **When configuring a Generator there are some controls that can be used to customize the behavior of the generative responses. What are the controls available to you to customize the generative AI responses? (Select two options)**
>
> - √ Text Prompt
> - √ LLM parameters, such as Temperature, Top P and Top K
> - √ Maximum responses
> - √ Stop sequences
> - √ Safety filters

1.4. Generative Fallback

Explore Generative Fallback responses powered by conversational design theory. Discover the different enablement levels for Generative Fallback, how to configure a text prompt and handle customer interactions.

1.4.1. Video - Generative Fallback responses

- YouTube: Generative Fallback responses

Generative Fallback Responses

The generative fallback feature uses LLMs with a custom prompt to respond to users when your chatbot doesn't know the answer.

Example Conversation Topic

Think about a user inquiring about their subscription to a service. The dialog could lead to several activities such as:

- Identifying the user
- Updating the tier on their account
- Getting confirmation that the tier was updated as requested
- Asking the user whether they needed anything else

At every stage in the flow, there's a chance that the Virtual Agent will not be able to match the user's intent to an appropriate response.

Generative Fallback Mechanism

Generative fallback is a mechanism for handling points in the conversation where the user moves away from the intended flow and doesn't trigger another action or flow.

Common User Requests

There are several common user requests that may occur between key use cases, such as:

- Repeating what the agent said in case the user didn't understand
- Holding the line when the user asks for it
- Summarizing the conversation
- Greet and say goodbye to the user

No-Match Errors

Even with robust intents, there is still room for error. Users may go off script by saying something unexpected, or making a mistake while filling in a form.

Generic Responses

Generic responses like "Sorry I'm not sure how to help" are often not good enough and can frustrate users.

Error Prompts

Error prompts should be inspired by the Cooperative Principle, according to which efficient communication relies on the assumption that there's an undercurrent of cooperation between conversational participants.

Grice's Maxims

The Cooperative Principle is based on four rules called Grice's Maxims:

- The truth of what we say
- The Maxim of Quality (the quantity of information that we provide)
- The Maxim of Quantity (the relevance of what we contribute)
- The Maxim of Manner (the way we strive to communicate clearly, without obscurity or ambiguity)

Research on Human-to-Human Conversation

People respond to technology as they would to another human. Users rely on their existing mental model of human-to-human conversation, following the Cooperative Principle, even when interacting with a machine.

Advances in Automatic Speech Recognition

The machine persona almost always knows exactly what users have said, but determining what the users meant is still a challenge.

Contextual Understanding

Utterances often can't be understood in isolation; they can only be understood in context. The LLM needs to keep track of context to understand the user's utterances.

Generative Fallback with Context

With generative fallback, the agent understands the pronoun 'he' in the context of the conversation and is able to progress the conversation along the desired path.

Managing Event Handlers

Generative fallback can help you manage event handlers without requiring someone to configure logic to handle each and every scenario that could occur in a customer conversation.

1.4.2. Video - Enablement levels

- YouTube: Enablement levels

Enablement Levels for Generative Fallback

Generative Fallback increases intent coverage and helps handle errors gracefully.

Enabling Generative Fallback on No-Match Event Handlers

- Can be used in flows, pages, or during parameter filling
- Whenever a no-match event triggers, Dialogflow will attempt to produce a generated response that will be said back to the user
- If the response generation is unsuccessful, the regular prescribed agent response will be issued instead

Enabling Generative Fallback for Entire Flow

- Can catch any no-match event that isn't explicitly handled in the flow
- Enables generative responses for no-match events for the entire flow by enabling it for the event handlers in the Agent response section of the Start Page of the flow

Using Generative Responses for No-Match Events at Page Level

- If enabled, Dialogflow will trigger generative fallback if the user input doesn't match any active intents in scope for the page
- Enables generative responses for specific pages by enabling it in the Agent response section of the event handler for a particular page

Using Generative Responses for No-Match Events at Parameter Level

- Enables handling a no-match raised by a violation of the entity type of the parameter
- Allows specifying reference values and synonyms, or a numeric value range that can be used to determine intents
- Regular expressions can be used to define the range for the parameter

Example of Flow Description

- "Search, find, and book liveaboards"

- Contains important information about the overall purpose of the Agent
- Keep this in mind when designing your own description of flows

Example of Intent Description

- "Assist users with group or full charter reservations"
- Initially collect travel details including departure period, destination, number of guests (min 4 max 15 people), contact details
- The description contains important information to complete the form, such as the available destinations, the minimum and maximum number of passengers allowed on a boat, the departure period and contact details.

Configuring Entity Type

- Can specify reference values and synonyms, or a numeric value range that can be used to determine intents
- Regular expressions can be used to define the range for the parameter

Enabling Generative Fallback at Parameter-Level

- Enables handling a no-match raised by a violation of the entity type of the parameter
- Allows specifying reference values and synonyms, or a numeric value range that can be used to determine intents
- Can enable generative fallback to handle invalid inputs when the agent asks for the number of guests

Example of Enabling Generative Fallback at Parameter-Level

- Open the page that contains the form parameters (e.g. number-of-guests parameter)
- Scroll down to the Reprompt event handlers section
- Select the relevant No-match event handler within the event handler settings
- Enable generative fallback in the Agent responses

1.4.3. Video - Configure a text prompt

- YouTube: Configure a text prompt

Configure a Text Prompt for Generative Fallback

The generative fallback feature passes a request to a large language model to produce a generated response.

Text Prompt Structure

- A mix of natural language and information about the current state of the agent and conversation.
- Must include context to set the tone of voice.

ML Tab in Agent Settings

- Select Generative AI sub-tab.
- Can select predefined prompt template or define custom prompt.

Predefined Prompts

- Default and Example are available initially.
- Later, additional templates can be added.

Defining a Custom Prompt

- Click "new template" option to create a new prompt.
- Select an existing template or define a new one.

Default Template

- Not directly modifiable, but can influence agent responses by adding details to Data store prompt.

Example Template

- Modifiable and serves as a guide for writing custom prompts.
- Walkthrough editing the Example prompt.

Text Prompt Editing

- Enter Template name and add Text prompt.
- Add context to generative fallback to set tone of voice.

Placeholders

- $conversation: conversation between agent and user, excluding last user utterance.
- $last-user-utterance: last text input submitted by user.
- $flow-description: flow description in Start page of active flow.
- $route-descriptions: intent descriptions of active intents.

Example Text Prompt

- "You are a friendly agent that likes helping traveling divers. You are under development and you can only help $flow-description placeholder."
- Includes placeholders for conversation, last user utterance, flow description, and route descriptions.

Replacing Placeholders with Contextual Information

- Custom text prompt has been configured with all four placeholders.
- Text prompt includes contextual information to provide necessary responses in customer conversation.

Generated Response

- LLM falls back on Flow and Page level descriptions to generate answer to customer.
- Output reads "I'm sorry Alessia, we can only help you with liveaboards in Costa Rica, Galapagos Islands and several locations around Mexico."

1.4.4. Video - Handling customer interactions

- YouTube: Handling customer interactions

Handling Customer Interactions with Generative Fallback

The first location in a Dialogflow CX solution where generative fallback can be enabled is at the flow level.

Flow Level Generative Fallbacks

- Enable more natural conversation before entering specific well-defined tasks.
- Example: chatbot asking "how are you doing?" and responding with a generative answer to refocus on diving trips.

Generic Questions

- Use generative responses to explain purpose and provide context.
- Examples: "how can you help me?", "what's a liveaboard?"

Page Level Generative Fallbacks

- Specific intents for specific use cases.
- Example: intent for booking a trip, directing to Collect further info page with description explaining chatbot assistance.

Parameter-Level Generative Fallbacks

- Take over when sending invalid responses.
- Example: user answering that they want to travel to the Canary Islands (destination not available), LLM guiding with custom answer explaining available locations.

No-Match Event Handler

- Invoked when users ask questions outside of specified intents.
- With generative fallback enabled at page level, LLM can provide context and generate response based on conversation history.

Banned Phrases

- Add list of words and phrases that would cause generative fallback to fail (e.g. "dangerous", "high crime-rate").
- If end-user input includes banned phrases, generation will be unsuccessful and regular prescribed response will be issued instead.

Best Practice

- Prevent no-match event from being triggered by providing good, robust, and varied training phrases for intents.
- Increase intent matching accuracy and keep conversations on intended route.

Generative Fallback Feature

- Awesome feature, but doesn't solve all problems.
- Leverages Vertex AI Foundation models, specifically PaLM2 for Text, text-bison, to power Generative Fallback feature.

This high-level overview shows how Dialogflow CX leverages Vertex AI Foundation models to power the Generative Fallback feature.

1.4.5. Link - Applying built-in hacks of conversation to your Voice UI

- Applying built-in hacks of conversation to your Voice UI launch

1.4.6. Quiz - Generative Fallback quiz

1.4.6.1. Quiz 1.

> **Important**
>
> **Which statement is true for Generative Fallback:**
>
> √ With Generative Fallback you no longer need to provide good, varied training phrases to your intents.
> √ Generative Fallback will always be able to handle a customer conversation if it goes off-track.
> √ Generative Fallback will always increase intent matching accuracy and keep your conversations on an intended route.
> √ Aim to prevent no-match scenarios by providing good, varied training phrases to your intents.

1.4.6.2. Quiz 2.

> **Important**
>
> **Generative Fallback is a mechanism for handling points in a conversation with a Dialogflow CX virtual agent, where the user moves away from the intended flow and doesn't trigger another action or flow, known as a No Match error. What kind of user response could invoke a No Match error? (Select two options)**
>
> √ A valid input while form filling
> √ An invalid input while form filling
> √ Remaining silent
> √ Saying something that matches defined intents
> √ Saying something unexpected

1.4.6.3. Quiz 3.

Important

A Dialogflow CX flow has been designed for booking diving trips. A particular page in the flow is configured to assist users with group reservations or full charters, the intent description states:'Currently you can assist users who are looking for a group reservation or a full charter. Initially collect travel details including departure period, destination, number of guests (min 4 max 15 people), contact details. The destination must be one of the following in the Pacific: Costa Rica, Mexico, Galapagos Islands.'Which Generative Fallback enablement levels would be triggered if the user tries to book a full charter for 18 people? (Select two options)

- √ The agent-level would be triggered as the minimum and maximum number of guests is defined as an entity parameter. The LLM uses the intent description to generate the response.
- √ The flow-level would be triggered as Generative Fallback always steps up through the levels to be handled by the top flow-level. The LLM uses the intent description to generate the response.
- √ All levels would be triggered as Generative Fallback always steps up through the levels to be handled by the top flow-level. The LLM uses the intent description to generate the response.
- √ The page-level would be triggered as the minimum and maximum number of guests is specified in the intent description. The LLM also uses this description to generate the response.
- √ The parameter-level would be triggered as the minimum and maximum number of guests is defined as an entity parameter. The LLM uses the intent description to generate the response.

1.4.6.4. Quiz 4.

Important

You can add a list of banned phrases for your Generative Fallback response. What happens if a banned phrase has been entered by a user as part of their response?

- √ A prescribed answer defined in Agent Says will be used to respond.
- √ The conversation will be directed to a human agent.
- √ Generative Fallback will highlight the issue and ask the user to repeat their response
- √ The conversation will be terminated.

1.4.6.5. Quiz 5.

> **Important**
>
> **What is the correct definition of the $route-descriptions prompt placeholder?**
>
> - √ The page description of the active pages.
> - √ The route descriptions of the active routes.
> - √ The intent descriptions of the active intents.
> - √ The flow description, in the Start page of the active flow.

1.4.6.6. Quiz 6.

> **Important**
>
> **You have been asked to enable generative responses for no-match events for specific pages, by enabling it in the Agent response section of the event handler. Where would you provide the information for the LLM to reference to generate its response?**
>
> - √ Page description
> - √ Flow description
> - √ Agent response
> - √ Intent description

1.5. Generative AI for Virtual Agents

Explore Generative AI Agents and hybrid agents, when to use generative AI or intent-driven flows in your Dialogflow CX solutions and how to create and configure Generative AI Agents. Also, learn how to create a data store, ingest data into a data store, and manage data in a data store

1.5.1. Video - Generative AI Agents and hybrid agents

- YouTube: Generative AI Agents and hybrid agents

Generative AI Agents and Hybrid Agents

Vertex AI Search and Conversation provides the backend technology for Dialogflow CX to build Generative AI Agents.

Two Components of Generative AI Agents

- Data stores: where company data is parsed and indexed.
- Generative AI Agent: powered by a Large Language Model, queries the data inside data stores.

Types of Generative AI Agents

- Website agent: answers questions based on website contents.

- FAQ agent: answers frequently asked questions.
- Employee-facing agent: answers help and support questions using internal helpdesk or HR support documents.

Special Intents

- Agent identification: questions like "Who are you?" or "Are you human?"
- Escalating to a human agent: questions like "I want to talk to a human" or "I want to talk to a real person".

Hybrid Agents

- Combine the power of precise conversation controls with data store handlers for generative features.
- Can be created by upgrading an existing Dialogflow CX agent.

Adding Generative AI Capabilities

- Important to remember that generative AI is not a solution to everything.
- Approach should be to keep what is working well in Dialogflow CX and incrementally add capabilities or build new use cases.

Understanding When to Use Intent Routes and Generative AI

- Intent-based routing for scenarios that require deterministic agent responses or actions that must be taken by the agent.
- Generative AI Agents and data stores for use cases where content can answer questions, such as website content or FAQs.

Benefits of Generative AI Agent Features

- Enables extension of coverage at low cost with answers from organizational knowledge documents or website content.
- Can replace certain intents with generative answers to increase quality and reduce maintenance.

Hybrid Agent Evaluation

- Dialogflow evaluates end-user input in the following order for hybrid agents:
    a. Parameter input while form filling
    b. Intent matches for routes in scope
    c. Data store handler with FAQ data store content
    d. Data store handler with unstructured data store content

1.5.2. Video - Intent-driven and Generative AI use cases

- YouTube: Intent-driven and Generative AI use cases

Intent-Driven and Generative AI Use Cases

The new Generative AI with LLM approach can now address use cases that were too costly or too complex in a cost-effective way.

Determine the Appropriate Option

Review the requirements of your specific use cases to determine the appropriate option between Intent-driven and Generative AI use cases.

Intent-Driven Use Cases

- Fully deterministic responses
- Auditable step-by-step process
- Transactional requests
- Strict data security and compliance
- Used when:
  - Manual intents, training phrases, routes, and conversation transitions are required.
  - Specific, high-volume intents with very specific business logic.

Generative AI Use Cases

- Allow for some variance in responses to offer dynamic, broad question coverage.
- Lower data security and compliance needs.
- Used when:
  - Allowing the bot to scan content or describe a bot task.
  - Achieving dynamic flows and extending intent bots with broad unstructured data.

Google Cloud's Approach

- Allows customers to design user journeys using both Intent-based flows and Generative AI in the same bot design, all within Dialogflow CX.

Intent-Based Flows vs. Generative AI

- Intent-based use cases are best for fully deterministic responses, auditable step-by-step processes, or existing bots delivering high containment.
- Generative AI use cases are best for extracting answers from unstructured data and chaining LLM calls to build fully generative virtual agent flows.

Hybrid Agents

- Use a mix of intents-based flows and generative AI for certain use cases, such as:

- Adding question coverage from unstructured data to an intent-based bot.
- Adding generative fallback coverage when intent-based flows hit a no-match outcome or an invalid parameter value is provided.
- Adding summarization or generated personal responses to intent-based flows.

Dialogflow and Vertex AI Search and Conversation

- Dialogflow CX Agent configured for business needs.
- Create and manage own data stores in Vertex AI Search and Conversation, covering website domains, unstructured documents, and structured FAQs.
- Virtual Agent uses intent detection to route customer questions to the relevant information.
- State handler route can be replaced by a Generative AI Agent that uses data stores to search for and return relevant information to the customer.

Non-Deterministic Responses

- Customer input to Dialogflow Agents can be resolved with a query to Vertex AI Search and Conversation for non-deterministic responses.

Example Scenario

- A customer uses an enterprise chat app and asks a natural language question relating to the company's operational manuals for a product.
- The Dialogflow Agent directs this question to Vertex AI Search and Conversation to access the relevant data store to answer the question.

Vertex AI Large Language Models

- Leveraged to generate answers, providing:
  - Results of the search
  - Snippets and summarizations of relevant documents or information
  - Multi-turn conversation capabilities to facilitate further support

Natural Language Response

- The information is returned as a natural language response to the customer, powered by:
  - PaLM 2 for Text foundation model
  - text-bison language model

Customer Experience

- The customer continues their conversation comfortably through the chat app, unaware of what is going on behind the scenes.
- The Dialogflow Agent has seamlessly integrated with Vertex AI Search and Conversation to provide a natural language response that meets the customer's needs.

Other intent routes, such as conditional route evaluation, or no match events can be supported by LLM based generative responses during fulfillment. This is a high-level overview of how the new generative AI features slot into existing processes to help virtual agents handle specific customer interactions.

Vertex AI Large Language Models are leveraged to generate the answers, providing the results of the search, snippets and summarizations of the relevant documents or information, and enables multi-turn conversation with the customer, in order to facilitate further support along this line of enquiry.

1.5.3. Video - Creating a Generative AI Agent for use in Dialogflow CX

- YouTube: Creating a Generative AI Agent for use in Dialogflow CX

Creating a Generative AI Agent for Use in Dialogflow CX

Now that we have explored how to create and configure a Hybrid Agent for Dialogflow CX, let's dive into the process of creating a Generative AI Agent.

Auto-Generate Route

To start with the Auto-generate route, follow these steps:

1. Create a chat application in Vertex AI Search and Conversation.
2. Provide a company name, time zone, and language.
3. Define an agent name and location.
4. Add existing data stores or create a new one.
5. Connect the data store to an existing agent.

Build-Your-Own Route

To add Generative AI Agent features to an existing agent using the Build-your-own route:

1. Select your flow from the FLOWS section of the Build tab.
2. Choose a page in the conversation flow.
3. Open the configuration window and click Add state handler.
4. Enable the Data Stores state handler by clicking the checkbox next to it.
5. Edit which data stores will be used for each page in the conversation flow.

Configuring Data Stores

For each field, you can either locate an existing data store or create a new one:

1. Create a new data store: Specify the type of app you want to create and provide a name for the data store.
2. Connect to an existing data store: Locate an existing data store that has been set up with your own data.

Using Generative AI Agent Features

Once the Data Stores have been enabled, you can:

1. Use the output of the Vertex AI Conversation request to the data store and combine it with some additional text in the Agent says text box.
2. Provide custom responses that reference generative answers using `$request.knowledge.questions[0]` in the Agent says section.

Customizing Responses

You can customize which types of scores to allow, allowing you to tailor the range of answers provided to a user based on context.

Tone and Identity

You can give a tone of voice and an identity to your Generative AI Virtual Agent by providing data store prompt information, such as a name and task description for the agent.

Language Support

The feature currently supports a range of languages, including Danish, Dutch, English, French, German, Hindi, Italian, Brazilian Portuguese, Spanish, and Swedish.

Global Region and Customer Managed Encryption Keys

Only the global region is supported, and customer managed encryption keys are not supported.

1.5.4. Video - Creating a data store

- YouTube: Creating a data store

Creating a Data Store

In this module, we'll explore creating an app engine, ingesting data into a data store, and managing data in a data store.

How to Create a Data Store

One of the greatest features of Vertex AI Search and Conversation is its ease of use for developers. You can create a data store based on certain data types, such as:

- Structured data that follows a schema (e.g., BigQuery, JSON)
- Unstructured data (e.g., HTML, PDFs with embedded text)

Step 1: Choose the Type of Data

Choose the type of data you want to import into your data store. This can be website data, structured data from BigQuery or Cloud Storage, or unstructured data.

Step 2: Prepare the Data

Prepare the data to be ingested into your data store. If the data is stored in Cloud Storage, it can be either structured (e.g., JSON or ND JSON) or unstructured (e.g., HTML, PDFs).

Step 3: Create a Schema

Create a schema for your data store. You can choose to:

- Auto-detect the schema
- Provide a custom schema
- Edit the auto-detected schema

Auto-Detecting the Schema

If you choose to auto-detect the schema, Vertex AI Search and Conversation will automatically detect the structure of your data. This is the recommended method for your first engine, as it's straightforward.

However, if you need more control over the schema, you can edit it manually. Keep in mind that editing the schema after auto-detection may require re-indexing, which can take a long time.

Providing a Custom Schema

If you choose to provide a custom schema, you'll need to create a JSON file manually.

Importing Data into the Data Store

Once you've created your schema, import data into your data store. You can do this by:

- Ingesting existing data from Cloud Storage or BigQuery
- Creating new documents in the data store

Configuring Searchable and Indexable Fields

When creating a schema, you'll also need to configure searchable and indexable fields. These fields will be used for search and indexing purposes.

- Indexable: defines whether a field can be filtered, faceted, boosted, or sorted.
- Searchable: defines whether a field can be reverse-indexed to match unstructured text queries.
- Retrievable: indicates whether a field can be returned in a search response.

Sharing Data Stores Across Engines

Data stores can be shared across engines. This is useful if you want to power both a search and conversation solution with the same data store.

Creating Multiple Data Stores

An engine can have multiple data stores, which is particularly useful if you need to query different types of data.

By following these steps and considerations, you'll be able to create and manage your own data stores in Vertex AI Search and Conversation.

1.5.5. Video - Ingesting data into a data store

- YouTube: Ingesting data into a data store

Ingesting Data into a Data Store

In this lesson, we'll learn how to ingest data into a data store.

Process of Ingesting Website Data

The process of ingesting website data into a data store is currently only available through the Google Cloud console. You can enter the URLs of the websites you want to include in your data store, and feel free to include any website you want (not just the ones you own).

Limitations of Ingesting Website Data

- Only public websites can be ingested.
- There is a maximum limit of 100,000 files that can be imported each time.
- A single file size cannot exceed 100 megabytes.

Creating and Ingesting Structured or Unstructured Data

You can create a data store and ingest structured or unstructured data from a cloud storage bucket or BigQuery using the Google Cloud console or code (e.g., Python).

Example Code Snippet for Importing Documents from Cloud Storage and BigQuery

Here's an example written in Python about how to import documents from both cloud storage and BigQuery:

```
import os

# Set up the credentials for your Google Cloud project
os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = 'path/to/credentials.json'

# Create a client instance of the Vertex AI Search and Conversation API
client = vertex_ai_search_and_conversation.V1Client()
```

```
8
9  # Define the URL of the website you want to ingest data from
10 website_url = 'https://example.com'
11
12 # Perform a full or incremental import of the data into the data store
13 import_data = client.import_data(website_url, 'structured' if is_structured else
   'unstructured')
14
15 # Handle any errors that may occur during the import process
16 try:
17     import_data.execute()
18 except Exception as e:
19     print(f'Error importing data: {e}')
```

Benefits of Ingesting Data into a Data Store

- Improves the quality and accuracy of search results.
- Boosts click-through rates for search results with higher engagement values.

Sending User Events

You can send real-time events using the API or JavaScript pixel to help improve the quality of your recommendations and search results.

1.5.6. Video - Managing data in a data store

- YouTube: Managing data in a data store

Managing Data in a Data Store

In this lesson, we'll learn how to manage data in a data store.

Life Cycle of a Data Store

The life cycle of a data store consists of the following stages:

1. **Creating a Data Store**
2. **Importing and Updating the Schema**
3. **Refreshing Data**
4. **Deleting an Existing Schema or Full Data Store**

Creating a Data Store

- Depending on the type of data store, you may need to create a schema for the data to be processed correctly.
- Alternatively, you can import the data without creating a schema initially.

Importing and Updating the Schema

- You can update the schema after importing the data by either adding incremental data or redoing an import with the new schema.
- Some changes are not supported, such as changing a field type (e.g., map to integer).

Refreshing Data

- You can refresh data by adding incremental data or redoing an import of the data.

Deleting an Existing Schema or Full Data Store

- Deleting an existing schema or full data store requires deleting the data first.
- Purging deletes only the documents from the data store, leaving the engine schema and configurations intact.

Python Example: Updating a Schema

```
import os

# Set up the credentials for your Google Cloud project
os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = 'path/to/credentials.json'

# Create a client instance of the Vertex AI Search and Conversation API
client = vertex_ai_search_and_conversation.V1Client()

# Define the URL of the website you want to ingest data from
website_url = 'https://example.com'

# Update the schema by adding incremental data or redoing an import with the new schema
update_schema = client.update_schema(website_url, {'new_schema': {'fields': [{'name': 'new_field', 'type': 'string'}]}})

# Handle any errors that may occur during the update process
try:
    update_schema.execute()
except Exception as e:
    print(f'Error updating schema: {e}')
```

Requirements and Limitations of Updating Schemas

- The new schema should be backward compatible with the original schema.
- Non-backward compatible schemas need to be deleted and recreated.
- Some changes are not supported, such as changing a field type.

Purging a Data Store

```
import os

# Set up the credentials for your Google Cloud project
os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = 'path/to/credentials.json'

# Create a client instance of the Vertex AI Search and Conversation API
client = vertex_ai_search_and_conversation.V1Client()

# Define the URL of the website you want to ingest data from
website_url = 'https://example.com'

# Purge the data store
purge_data_store = client.purge_data_store(website_url)

# Handle any errors that may occur during the purge process
try:
    purge_data_store.execute()
except Exception as e:
    print(f'Error purging data store: {e}')
```

Purging Data from a Structured or Unstructured Data Store

You can use the `purge` method to delete only the documents from the data store, leaving your engine schema and configurations intact. There is no option to purge data from a website data store.

1.5.7. Quiz - Generative AI for virtual agents quiz

1.5.7.1. Quiz 1.

> **Important**
>
> **Which two statements are true for Generative AI Agents? (Select two options)**
>
> - √ Keep what is working well in Dialogflow CX and add generative AI capabilities.
> - √ Generative AI Agents can be used to replace intent-based flows.
> - √ With Hybrid Agents you can use a mix-and-match approach to leverage both intent-based flows and generative AI use cases.
> - √ You have to clearly define use cases as either intent-based or generative AI.
> - √ Generative AI Agents can handle points in the conversation where the user moves away from the intended flow and doesn't trigger another action or flow.

1.5.7.2. Quiz 2.

> **Important**
> **Which two features are used by Generative AI Agents to provide a response based on your data? (Select two options)**
>
> √ Data stores
> √ Entity parameters
> √ Playbooks
> √ Large Language Models
> √ Intent descriptions

1.5.7.3. Quiz 3.

> **Important**
> **Data stores are used to find answers for end-user's questions. A data store can consist of different sources.**
> **What types of source data are available for data stores in a Vertex AI Search and Conversation?**
>
> √ Website, structured and unstructured data.
> √ Document, Table, and Image data.
> √ SQL, No-SQL and Data Warehouse data.
> √ Customer facing and internal search data.

1.5.7.4. Quiz 4.

> **Important**
> **What are two other important and useful features that are part of Generative AI Agents and data stores? (Select two options)**
>
> √ Data store prompt
> √ Output parameter
> √ Grounding confidence
> √ Banned phrases
> √ Training phrases

1.5.7.5. Quiz 5.

> **Important**
> **A Virtual Agent uses intent detection to route customer questions to the relevant information. Generative AI capabilities can be added to handle certain intent detection routes. Which aspect of Dialogflow CX Agent intent detection do Generative AI Agents cover?**
>
> √ Intent detection or parameter/conditional route evaluation
> √ No Match / no input

- √ State handlers
- √ Parameter input while form filling

1.6. Vertex AI Search and Conversation Architecture and Security

Familiarise yourself with Vertex AI Search and Conversation architecture, authentication to Vertex AI Search and Conversation apps, access control with Identity and Access Management (IAM), data governance and compliance, and pricing.

1.6.1. Video - Vertex AI Search and Conversation architecture

- YouTube: Vertex AI Search and Conversation architecture

Vertex AI Search and Conversation Architecture

In this lesson, we'll explore the Vertex AI Search and Conversation architecture.

Security Controls

- Vertex AI Search and Conversation provides the same security controls as many other Google Cloud perimeter products.
- Customer data is always stored with encryption using customer-managed encryption keys.
- Access to customer data is logged and traceable by using access transparency.
- Data residency and VPC security controls continue to be respected.

LLM Architecture

- LLMs are stateless, meaning weights are frozen and cannot be reconfigured.
- They do not store customer data including customer-specific parameters.
- Alternative cloud data can be used with LLMs, but the customer must process the data into a specific format before moving it to Cloud Storage.

Data in Flight

- Data is always encrypted when transmitted between systems.
- Customer data is not used to benefit another customer.

Google Cloud's AI Stack

- Google Cloud provides scale and speed for organizations to deploy large language models or run batch prediction jobs on structured data.
- The stack is optimized for AI workloads in three main ways:
    a. **Scale and Speed**: Google's AI infrastructure makes it fast and cost-effective for organizations to scale their end-to-end machine learning workloads.

b. **Price, Performance, and Efficiency**: Google is a leader in hardware-optimized for running large models and has created the Tensor Processing Unit (TPU) which is highly specialized for training large models.
c. **Sustainability**: Google's TPU infrastructure runs on 90% clean energy and is the most sustainable Cloud option for Enterprise customers.

Benefits of Vertex AI Search and Conversation

- Provides the security expected from Google Cloud perimeter to enable the best of AI.
- Enables organizations to deploy large language models or run batch prediction jobs on structured data without compromising security.
- Optimizes for scale, speed, price, performance, and efficiency while ensuring sustainability.

1.6.2. Video - Authentication to Vertex AI Search and Conversation apps

- YouTube: Authentication to Vertex AI Search and Conversation apps

Authentication to Vertex AI Search and Conversation Apps

In this lesson, we'll learn how to gain programmatic access to Vertex AI Search and Conversation and authenticate your application.

Vertex AI Search and Conversation Client Libraries

- Available for many popular programming languages:
  - Python
  - Node.js
  - Java
  - Go
  - PHP
  - Ruby
  - .NET (including C#)

Example: Using the Python Client Library to Perform a Search

1. Import the Google Cloud client library for Vertex AI Search.
2. Create a client connection to the search engine and configure it with:
   - Full resource name of the search engine
   - Serving config

Using the Client Libraries to Manage Resources

- Easy management of Vertex AI Search and Conversation resources using natural language programming style.

Authentication Using a Service Account

1. Create a service account in the Google Cloud console.
2. Generate and download credentials file.
3. Set an environment variable with the path to the downloaded credentials file.
4. Make an authenticated API request in your code.

Example Code in Python

```python
import os
from google.oauth2 import service_account

# Set up credentials from environment variable
credentials = service_account.Credentials.from_service_account_file(
    os.environ['GOOGLE_APPLICATION_CREDENTIALS']
)

# Create client connection to search engine
search_client = vertexai_search.V1Client(credentials=credentials)

# Perform a search and send results back
response = search_client.search(
    request={'query': 'example query', 'language': 'en'}
)
```

Important Notes

- If you don't explicitly specify the credentials when constructing a client, the client library will look for credentials in the environment.
- The example code shows that if you don't specify the credentials explicitly, the client library will automatically use the ones from the environment.

1.6.3. Video - Access control with IAM

- YouTube: Access control with IAM

Access Control with IAM

In this lesson, we'll explore the types of Access Control with Identity and Access Management (IAM) to grant the right level of access and permissions to the right people for your Vertex AI Search and Conversation resources.

Levels of Access in IAM

- **Discovery Engine Viewer**: Read-only access to all Vertex AI Search and Conversation resources.

- **Discovery Engine Editor**: Read-write access to all Vertex AI Search and Conversation resources, including:
    - Importing, updating, and deleting documents in the search engine.
    - Creating, updating, and tuning the model.
    - Creating user events and updating widgets.
- **Discovery Engine Admin**: Full control over all Vertex AI Search and Conversation resources, including:
    - Management of data stores.
    - Controls serving configurations.
    - Target sites.

Predefined IAM Roles

IAM provides three main levels of access for Vertex AI Search and Conversation:

1. Discovery Engine Viewer: Read-only access to all Vertex AI Search and Conversation resources.
2. Discovery Engine Editor: Read-write access to all Vertex AI Search and Conversation resources, including importing, updating, and deleting documents in the search engine.
3. Discovery Engine Admin: Full control over all Vertex AI Search and Conversation resources.

Assigning IAM Roles

You can assign these predefined IAM roles to grant granular access to Vertex AI Search and Conversation resources:

- Assign the Discovery Engine Viewer role to Auditors of the infrastructure.
- Assign the Discovery Engine Editor role to team members who need to import, update, or delete documents in the search engine.
- Assign the Discovery Engine Admin role to administrators who have full control over all Vertex AI Search and Conversation resources.

Best Practices

- Understand the levels of access that are available within IAM and the types of roles and responsibilities they might apply to your organization.
- Grant the right level of access and permissions to the right people for your Vertex AI Search and Conversation resources.
- Use predefined IAM roles to provide granular access to resources and ensure compliance with organizational policies.

1.6.4. Video - Data governance and compliance

- YouTube: Data governance and compliance

Data Governance and Compliance

In this lesson, we'll learn more about data governance and compliance with Google Cloud.

Importance of Data Governance and Compliance

- Highly regulated industries like financial services and healthcare require guaranteeing that no one can access confidential or sensitive data.
- Ensuring that customer data is not being used for any other purpose than determined by the organization.
- Google Cloud's approach to governance of customer data for cloud LLMs and generative AI is designed to provide enterprises with confidence in using generative AI.

Google Cloud's Approach to Data Governance

- Transparency: Google Cloud provides transparency into how customer data is being used and processed.
- Compliance with regulations: Google Cloud complies with regulations like GDPR, HIPAA, and privacy best practices by default.
- No use of customer data for training LLMs without permission: Google Cloud does not use customer data to train LLMs in accordance with the Google Cloud terms and the cloud data processing addendum.

Key Features of Google Cloud's Data Governance

- **Customer data ownership**: Customer data is owned and controlled by the organization.
- **Data minimization**: Only necessary data is collected, processed, and stored.
- **Anonymization**: Data is anonymized to prevent re-identification of individuals.
- **Encryption**: Data is encrypted both in transit and at rest.

Google Cloud's Compliance with Regulations

- GDPR (General Data Protection Regulation)
- HIPAA (Health Insurance Portability and Accountability Act)
- Other regulations and industry standards

Google Cloud's Commitment to Privacy

- Google Cloud is committed to protecting customer data and ensuring that it is used in accordance with the organization's policies.
- Google Cloud provides a range of tools and services to help organizations manage their data governance and compliance programs.

By following these principles, enterprises can trust that they are using generative AI in a responsible and compliant manner.

1.6.5. Quiz - Vertex AI Search and Conversation architecture and security quiz

1.6.5.1. Quiz 1.

> **Important**
>
> **The website team is creating a microservice that will be responsible for creating controls and serving configs to filter out what data is presented to users after they perform a query. Which role should you assign them?**
>
> - √ Discovery Engine Viewer
> - √ Discovery Engine Manager
> - √ Discovery Engine Editor
> - √ Discovery Engine Admin

1.6.5.2. Quiz 2.

> **Important**
>
> **Choose the answer that best describes the security posture of Vertex AI Search and Conversation.**
>
> - √ Vertex AI Search and Conversation offers the same security capabilities as other Google Cloud Perimeter products like BigQuery, including data encryption, no data sharing across customers, and Google will never use your data to train foundation models.
> - √ Vertex AI Search and Conversationoffers the same security capabilities as other Google Cloud Perimeter products, and only uses your logs and metadata to train foundation models.
> - √ Vertex AI Search and Conversation offers the same security capabilities as other Google Cloud Perimeter products, and only uses your data in Google Cloud Storage or BigQuery to train foundation models.
> - √ Vertex AI Search and Conversation offers the same security capabilities as other Google Cloud Perimeter products, and only uses your prompt data to train foundation models.

1.6.5.3. Quiz 3.

> **Important**
>
> **The development team is creating a microservice that will be responsible for ingesting new documents into the Vertex AI Search and Conversation unstructured data store on bulk every hour. Which role should you assign them?**
>
> - √ Discovery Engine Admin

- √ Discovery Engine Editor
- √ Discovery Engine Manager
- √ Discovery Engine Viewer

1.6.5.4. Quiz 4.

> Important
>
> **The security team needs to perform some audit tasks to your datastores on Vertex AI Search and Conversation to make sure the right documents and user events are being ingested. Which role should you assign them?**

- √ Discovery Engine Viewer
- √ Discovery Engine Editor
- √ Discovery Engine Admin
- √ Discovery Engine Manager

1.7. Generative Playbooks

Discovering Generative Playbooks in Dialogflow CX

Generative Playbooks is a feature in Dialogflow CX that enables you to build chat experiences with natural language by leveraging Large Language Models (LLMs). This feature allows you to create custom chatbots that can understand and respond to user queries in a more human-like way.

Benefits of Generative Playbooks

- **Improved conversational experience**: Generative Playbooks enable your chatbot to engage in more natural-sounding conversations with users.
- **Increased flexibility**: With Generative Playbooks, you can create custom chatbots that can adapt to different user queries and contexts.
- **Enhanced customer engagement**: By providing a more personalized and responsive chat experience, Generative Playbooks can help increase customer satisfaction and loyalty.

Use Cases for Generative Playbooks

- **Customer support**: Use Generative Playbooks to create chatbots that can answer common customer support questions and provide personalized solutions.
- **E-commerce**: Leverage Generative Playbooks to power chatbots that can help customers find products, make purchases, and track orders.
- **Education**: Use Generative Playbooks to create interactive learning experiences that can adapt to different student needs and abilities.

Generative Playbooks vs. Generators and Generative Fallback

- **Generators**: Generate responses based on a specific intent or context.

- **Generative Fallback**: Provide default responses when the user's query does not match any predefined intents.
- **Generative Playbooks**: Combine the capabilities of both Generators and Generative Fallback, enabling your chatbot to respond to user queries in a more flexible and natural-sounding way.

Step-by-Step Instructions for Creating a Dialogflow CX Agent with a Generative Playbook

1. **Create a new Dialogflow CX agent**: Go to the Dialogflow CX console and create a new agent.
2. **Enable Generative Playbooks**: In the agent settings, enable Generative Playbooks.
3. **Create a flow**: Create a new flow that will be used as the foundation for your chatbot's conversation.
4. **Invoke the flow within a Playbook**: In the Playbook editor, create a new entity to represent the user's query and invoke the flow using the `Flow` entity type.
5. **Connect the Playbook with a Generative AI Agent data store**: Connect the Playbook with a Generative AI Agent data store to enable your chatbot to access pre-trained models and generate responses.

Connecting a Dialogflow CX Agent with a Generative AI Agent Data Store

1. **Create a new entity in the Playbook editor**: Create a new entity to represent the user's query.
2. **Invoke the flow using the `Flow` entity type**: In the Playbook editor, invoke the flow using the `Flow` entity type.
3. **Connect the Playbook with a Generative AI Agent data store**: Connect the Playbook with a Generative AI Agent data store to enable your chatbot to access pre-trained models and generate responses.

By following these steps, you can create a Dialogflow CX agent that leverages Generative Playbooks to provide a more natural-sounding conversation experience for your users.

1.7.1. Video - Introduction to Generative Playbooks on Dialogflow CX

- YouTube: Introduction to Generative Playbooks on Dialogflow CX

Introduction to Generative Playbooks on Dialogflow CX

Generative Playbooks is a DialogFlow feature that allows developers to build chat experiences with natural language by leveraging Large Language Models (LLMs) within DialogFlow CX.

Key Benefits of Generative Playbooks

- **No need to define flows, pages, intents, and transitions**: Instead, you provide natural language instructions and structured data in the form of playbooks.
- **End-to-end workflow definition**: Generative Playbooks enable you to define end-to-end workflows for your DialogFlow CX Agents.
- **LLM-driven flow development**: Generative Playbooks leverage LLMs to simplify development and maintenance.

Use Cases for Generative Playbooks

- **Customer Service Bots**: Answer customer questions, troubleshoot issues, and provide information.
- **Sales and Marketing Bots**: Generate leads, qualify prospects, and answer questions.
- **Productivity Bots**: Schedule appointments, create tasks, and find information.
- **Education and Training Bots**: Assess a student's level, answer questions, and give feedback.
- **Research Bots**: Collect data, conduct surveys, and analyze data.

ReAct Pattern

Generative Playbooks implement the ReAct pattern: reasoning and action. This prompt engineering technique originated at Google, and Generative Playbooks are a first-class implementation of the technique.

How ReAct Works

1. **Reasoning**: The LLM thinks about how to act.
2. **Action**: The LLM takes an action (e.g., formulates a call to an external system).
3. **Observation**: The external system responds with information (observation).
4. **Repeat**: The cycle repeats until the LLM arrives at an answer.

ReAct Cycle

The ReAct cycle is repeated until the LLM arrives at an answer. This pattern of thought, to action, to observation is called the core ReAct loop or ReAct cycle.

Generative Playbook Tooling

- **OpenAPI format**: Define the structure of the external system's API endpoint.
- **Cloud Functions or external APIs**: External systems can be implemented using Cloud Functions or external APIs.

By leveraging Generative Playbooks, you can create more natural-sounding conversations with your users and simplify development and maintenance.

1.7.2. Video - Creating a Generative Playbook

- YouTube: Creating a Generative Playbook

Creating a Generative Playbook

You can create a DialogFlow agent as usual, either by creating an auto-generated generative agent or by building your own.

Starting with a Default Flow

- You can start with a default flow, which will automatically use Generative Playbooks.
- Alternatively, you can select the generative agent type to begin with a flow that uses Generative Playbooks.

Creating a Playbook

- A playbook is a set of natural language instructions that the virtual agent will carry out.
- You'll create a list of step-by-step execution instructions to accomplish your target goal.
- The help text tells you how to format the text of your instructions, including using markdown syntax for an unordered list.

Nesting Steps

- You can nest steps within other steps.
- You can use the dollar sign followed by a tool name to make external calls or perform actions.

Tool References

- The $PLAYBOOK parameter allows you to refer to another playbook by name.
- Tool references are used to combine external data and LLM information.

Playbook Structure

- A playbook consists of steps, which may include tools, examples, input parameters, output parameters, and more.
- The structure of a playbook is similar to a flow, but with additional features such as tool references and example inputs.

Example Playbook: Renew License

- This playbook is an example of a traditional DialogFlow CX flow where a particular group of tasks happen (renewing a driver's license).
- It uses the $TOOL variable to invoke an external tool.

- The playbook says: "Ask the customer to provide their most recent driver's license number and expiry date and verify them using the $TOOL: drivers_license tool."

Playbook Examples

- Playbooks can include examples, which contribute to the LLM's understanding of the conversation by providing in-context learning, also called few-shot prompting.
- Examples define what a good conversation should look like and specify the input and output for your tools.

Input and Output Parameters

- Input parameters contain values passed into a playbook from a flow.
- Output parameters contain values sent back to a flow from a playbook.
- You can define parameters, then define an example using the parameters.

Tools

- Tools represent a collection of functions that are available to the playbook and the dialog manager.
- They allow the conversational agent to combine external data and LLM information.
- Tools are defined by a schema in the OpenAPI 3.0 format.

OpenAPI Definition for Renewing a Driver's License

- The example shows an OpenAPI definition for renewing a driver's license, including a POST request to the verify last driver's license endpoint.

Testing a Playbook

- You can test a playbook in the DialogFlow simulator.
- Conversation history shows when a Generative Playbook has been invoked and when a tool has been used.

1.7.3. Video - Generative Playbooks - limits and tips

- YouTube: Generative Playbooks - limits and tips

Generative Playbooks - Limits and Tips

Generative Playbooks have some limitations, such as:

- Only able to make five calls per conversation turn.
- Limit of 15,000 input tokens and 500 output tokens summed across a conversation turn.

There are also limitations when it comes to HTTP API calls, including:

- Except for the session ID, query parameters are not supported.
- The body of requests and responses must be empty or in JSON format.

- Advanced schema features like "oneOf" and the "multipleOf" data type are not supported.
- Parameter validation has to be handled with web hooks.

Generative Playbook Tool Schemas

- Currently only the application JSON format is supported.
- Plain text and other formats are not supported.
- A summary field is required in all API paths.
- Currently, only the specification of one path is supported.
- Nesting using keywords like "allOf" or "anyOf" is not supported.

Prompt Engineering

- Prompt engineering is a new and very sensitive virtual agent development technique.
- Generative agents and flows are still in the very early stages of developing best practices.
- LLMs don't always perform in the most predictable way, and the current model performance is far from perfect.
- Writing clear and descriptive instructional steps is important.
- The quality and quantity of your training examples will determine the accuracy of the agent's behavior.

Tips for Using Generative Playbooks

- If the agent is not responding or behaving as expected, it's often due to inadequate or missing training examples.
- Include examples with your instructions to provide a clear template for how to construct a response aligned to your expectations.
- Use few-shot examples to contribute greatly to the performance of model prediction.
- Keep in mind that there's no need to painstakingly obsess over the exact language of your steps if you're not providing examples.

Best Practices

- Write clear instructions combined with thorough training examples.
- Provide four or more representative few-shot examples for each playbook and cover all happy path scenarios.
- Use prompt engineering techniques to create a natural way to create conversational flows.
- Focus on providing high-quality training examples rather than just writing instructions.

1.7.4. Video - Hybrid flows with Generative Playbooks

- YouTube: Hybrid flows with Generative Playbooks

Hybrid Flows with Generative Playbooks

Generative Playbooks can be used with Virtual and Generative AI Agents. You can invoke a Dialogflow CX flow within a Playbook, provided that the flow already exists.

Invoking a Dialogflow CX Flow within a Playbook

- To invoke a Dialogflow CX flow within a Playbook, you can use the variable FLOW and specify a flow name.
- The following instruction says that "If the customer booked an appointment, use the make payment flow to help them pay for the appointment."
- If the condition matches that statement, the conversation will be directed to that flow.

Passing Parameters between Flows and Playbooks

- You can pass parameters between flows and playbooks.
- To invoke a Dialogflow CX flow within a Playbook, you can use the variable FLOW and specify a flow name.

Connecting a Generative Playbook with a Generative AI Agent Data Store

- These data stores can be configured for web, structured or unstructured data.
- A Generative AI Agent can access relevant domains or documents to provide an answer to a customer with that data.
- For example, the Generative AI Agent could search the existing FAQs on the company website to generate a response to the customer through Dialogflow CX.

Connecting a Generative Playbook to a Data Store

- To connect a Generative Playbook to a data store, you need to:
    a. Create a Playbook tool for the Generative AI Agent.
    b. Select the data store in the Tool user interface and save the tool.
    c. Reference the tool in the Playbook instructions to call upon that data.
    d. Test the new Generative Playbook to ensure it is performing as expected.

Creating a Tool for a Generative AI Agent

- You will need to provide a name for the tool, select Data store as the tool type, and then provide a description of what the data store will be used for.
- Next, select the data store that you want to use of the appropriate data store type.
- Then save the tool.

Referencing a Tool in a Playbook

- You can reference the tool in the Playbook instructions by using the $TOOL: FAQ parameter.
- In this example, both steps include a reference to the tool.
- Step 1 reads: "If a user asks a question, trigger the FAQ tool."
- Step 2 states that: "If the FAQ tool has a valid answer, share it with the user."

Testing a Generative Playbook

- Test the new Generative Playbook in the Simulator to ensure that the natural language response provided by the Generative AI Agent contains the relevant data from the data store.

Mutually Exclusive Features

- Generative Playbooks cannot be used with Generators or Generative Fallback, these are mutually exclusive features.
- If you're inside of a Playbook, the LLM is in full control, and you'll be able to provide instructions that act as a Generative Fallback would.

Hybrid Flows

- Generative Playbooks can be used with traditional Flows.
- You can route to a Playbook seamlessly from a normal Dialogflow CX Flow by choosing the playbook option in the transition menu.

1.7.5. Quiz - Generative Playbooks quiz

1.7.5.1. Quiz 1.

> **Important**
> **What is a tool in the context of a playbook? (Select two options)**
>
> √ Tools enable you to use a mix-and-match approach to leverage both intent-based flows and generative AI use cases.
> √ Tools are API calls, defined by a schema in the OpenAPI 3.0 format.
> √ Tools can be configured for web, structured or unstructured data
> √ Tools are a collection of functions that are available to the playbook and the Dialog Manager.
> √ Tools are a prompt for an LLM that can execute on a single turn of a conversation, such as LLMs for Dialogflow CX fulfillment.

1.7.5.2. Quiz 2.

> **Important**
> **What are two of the limits of Generative Playbooks? (Select two options)**
>
> √ 10 LLM calls per conversation turn
> √ 15k input tokens and 500 output tokens summed across a conversation turn

- √ 13k input tokens and 500 output tokens summed across a conversation turn
- √ 15k input tokens and 300 output tokens summed across a conversation turn
- √ 5 LLM calls per conversation turn

1.7.5.3. Quiz 3.

Important

What are two powerful features of Generative Playbooks? (Select two options)

- √ Playbooks can completely replace manual flow-development.
- √ Playbooks enable LLMs to access APIs and tools.
- √ Playbooks can leverage LLMs to handle No Match errors.
- √ Playbooks can leverage LLMs for instruction-driven flow development.
- √ Playbooks will always increase intent matching accuracy and keep your conversations on an intended route.

1.7.5.4. Quiz 4.

Important

What are two generative AI related features of Dialogflow CX that can be used with Playbooks? (Select two options)

- √ Data stores
- √ Generative AI Agents
- √ Vertex AI Search
- √ Generators
- √ Generative Fallback

1.7.5.5. Quiz 5.

Important

What is the purpose of the ReAct pattern?

- √ To enable LLMs to generate both verbal reasoning traces and text actions to solve various language reasoning and decision making tasks.
- √ To enable planning and action through mapping text contexts to text actions
- √ To enable LLMs to use prompt chaining to break down a complex task into smaller, more manageable subtasks, and then chaining together prompts that can be used to complete each subtask.
- √ To enable Chain of Thought to solve multi-step reasoning problems.

1.7.5.6. Quiz 6.

Important

What can you do to improve the accuracy of the agent's behaviour?

- √ Write clear instructions combined with thorough training examples.

- √ Provide a representative one-shot example for each Playbook.
- √ Cover one happy path scenario with an example.
- √ Focus on the exact language when writing clear and descriptive instructions.

1.7.5.7. Quiz 7.

Important

What are two key steps in connecting a Generative Playbook with a data store?

- √ Configure a data store for use with Generative Playbooks
- √ Select the data store in the tool user interface and save the tool.
- √ Test the data store
- √ Create a Playbook tool for the Generative AI Agent
- √ Use the variable FLOW and specify a flow name.

1.8. Your Next Steps

1.8.1. Badge - Course Badge