

RFP Demo with Vector Databases and Prompt Builder copie

Demo Owner

Sharda Rao - Distinguished Technical Architect Data Cloud

Thank you

Matt Dykeman - Technical Architect Data Cloud
Reinier van Leuken - Product Manager Salesforce
Raveesh Raina - Principal Account SE
Jonathan Zhou - Distinguished Technical Architect

Business Problem

Sales representatives often find themselves grappling with the arduous task of responding to RFP questions, which can be both tedious and time-consuming.

However, a revolutionary solution has emerged with the integration of vector databases and RAG (Retrieval Augmented Generation). This innovation allows sales professionals to leverage their historical RFP responses to automatically generate content for their current inquiries.

This groundbreaking feature is now seamlessly integrated into Salesforce, along with the robust capabilities of a data cloud. Salesforce representatives can now effortlessly generate context-specific content directly within the system of engagement.

Furthermore, they have the flexibility to modify the generated response and update answers accordingly, leading to a significant boost in productivity and an enhanced adoption of Salesforce within the sales team.

By harnessing the power of vector databases and RAG technology, this transformative feature not only streamlines the RFP response process but also empowers sales reps to customize and optimize their answers within the Salesforce ecosystem. This not only saves time but also contributes to an overall increase in efficiency and effectiveness, ultimately driving improved sales performance.

Demo Script

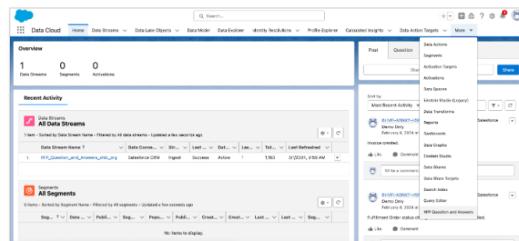
Demo Link - <https://clicktologin.herokuapp.com/?un=sharda@rag.demo&pw=salesforce1>

NOTE - The password has been changed. Please DM Sharda Rao for the new password.

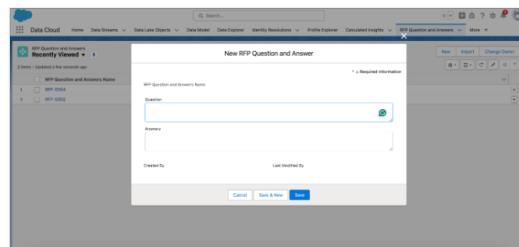
Video - https://drive.google.com/file/d/11tVJKI6Uj4aMG3_wt8LTPnh6QP3io1w3/view?usp=sharing

RFP Demo with Vector Databases and Prompt Builder copie

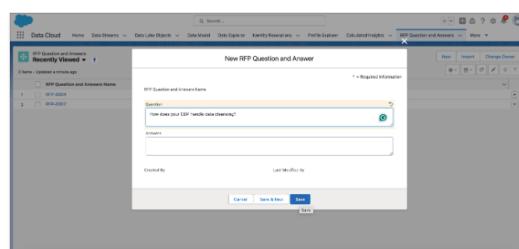
Navigate to Data Cloud → RFP Question and Answers



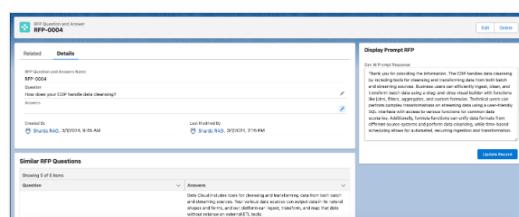
Click New



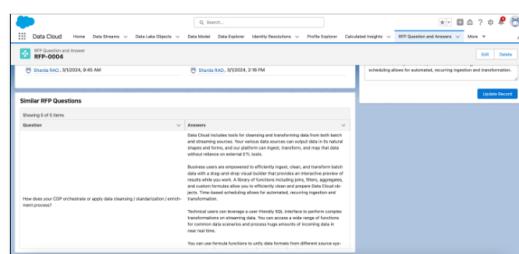
Enter the Question and Click Save



Expand the RFP text box on the right panel.
Highlight the text generated by prompt template



Scroll down and show the similar question and answer generated by vector search



Highlight how you can change the text generated by LLM and click save to update the

RFP Demo with Vector Databases and Prompt Builder copie

answer in Salesforce object

The screenshot shows the Salesforce Data Cloud interface. A modal window titled "RFP-0004" is open, displaying details about a related record. The modal has two tabs: "Related" and "Details". The "Details" tab contains information such as "RFP Question and Answers Name: RFP-0004", "How does your CDP handle data cleansing?", and "Answers". Below the modal, a section titled "Similar RFP Questions" lists "Showing 0 of 5 items". To the right of the modal, a larger panel titled "Display Prompt RFP" shows the full RFP record with its content.

Highlight how the record got updated after Sales rep reviews and updates the answer

This screenshot shows the same Salesforce Data Cloud interface after a review and update. The "Display Prompt RFP" panel now displays the updated content from the sales rep's review. The "Answers" section has been modified to reflect the changes made during the review process.

Navigate to Set up → Prompt Builder → RFP No Grounding → Choose RFP002 as a related record. Highlight how with no context the LLM response is not relevant for the Sales rep

The screenshot shows the Salesforce Prompt Builder interface. A modal window titled "Prompt Template Workspace" is open, showing a question template. The "Resolution" and "Response" sections both contain placeholder text that is not directly related to the question asked, indicating that without grounding, the LLM generates irrelevant responses.

Navigate to Set up → Prompt Builder → RFP with Grounding → Choose RFP002 as a related record. Highlight how we are now sending the context from our vector search of answers to similar questions on the left panel and show the grounded response on the right panel. Mention how the context is sent using an apex class search context.

This screenshot shows the same Prompt Builder interface but with "Grounding" selected. The "Resolution" and "Response" sections now display a more relevant and contextually appropriate response, generated by the LLM based on the provided ground truth context.

In essence, this demonstration effectively illustrates the seamless integration of vector search and the Retrieval Augmented Generator (RAG) within Salesforce DataCloud. The primary goal is to showcase to customers the effortless application of these technologies, specifically emphasizing their ability to anchor prompts within the natural workflow of Salesforce users. By doing so, the demo aims to highlight the enhanced ease with which users can navigate and leverage these tools, ultimately resulting in a significant boost in productivity. This showcase not only underscores the user-friendly nature of the system but also accentuates its capacity to streamline processes, making it an invaluable asset for the Salesforce user community.

WIP - PDF and knowledge articles vector databases

💡 How to build your own

1. Historical dataset to create the custom object. Note that you can use any unstructured data pertinent to your use case

FP Responses.csv

RFP Demo with Vector Databases and Prompt Builder copie

2. Ingest the data into DataCloud as a Data Stream
3. Map it to a custom DMO
4. Navigate to Data Cloud → Search Index

The screenshot shows the Data Cloud interface with the following details:

- Page Title:** RFP Question and Answer
RFP-0004
- Related Tab:** Details
- Question:** How does your CDP handle data cleansing?
- Answers:** A text area containing a response about Data Cloud's data transformation capabilities.
- Created By:** Sharda RAO, 3/1/2024, 9:45 AM
- Last Modified By:** Sharda RAO, 3/1/2024, 2:28 PM
- Similar RFP Questions:** Showing 5 of 5 items (Question and Answers)
- Display Prompt RFP:** A sidebar with a Gen AI Prompt Response message and a list of Data Action Targets (Delete, Data Actions, Segments, Activation Targets, Activations, Data Spaces, Einstein Studio (Legacy), Data Transforms, Reports, Dashboards, Data Graphs, Einstein Studio, Data Shares, Data Share Targets, Search Index, Query Editor).

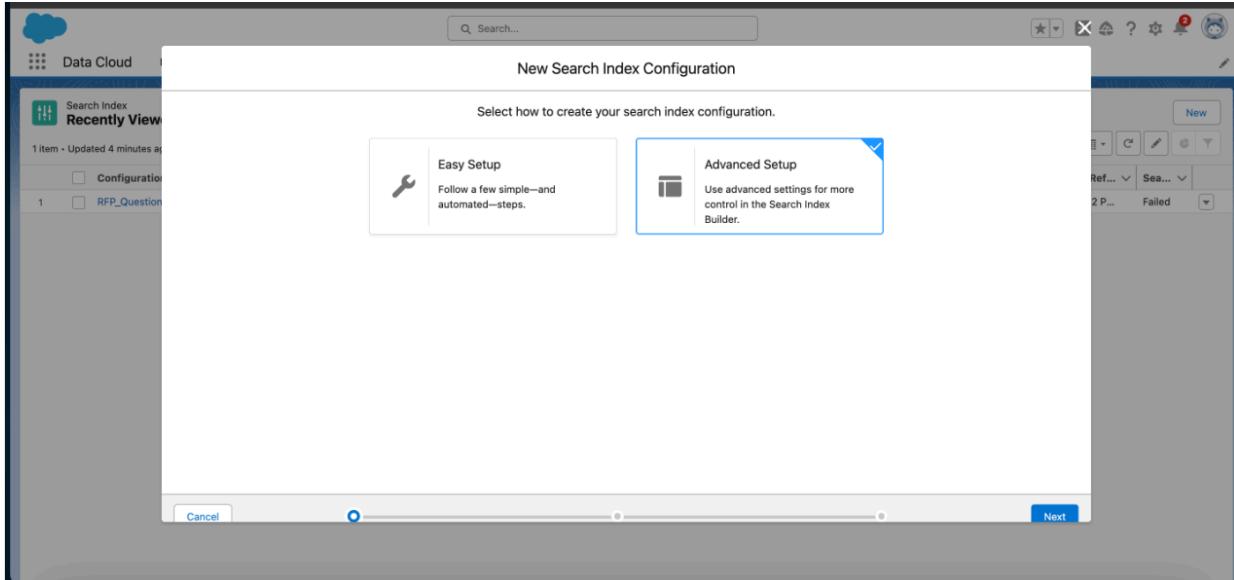
5. Create new

The screenshot shows the Data Cloud interface with the following details:

- Page Title:** Search Index
- Recently Viewed:** RFP_Question_and_Answers
- Table Headers:** Configuration Name, Data..., API Name, Source Data Model Obj..., Vector Data Model Obj..., Chunk Data Model Obj..., Last Modified Date, Search Index Ref..., Sea...
- Table Data:** One item listed: RFP_Question_and_Answers, defa..., RFP_Question_and_Answers, RFP_Question_and_Answers, RFP_Question_and_Answers, RFP_Question_and_Answers, 2/26/2024, 12:52 PM, 2/29/2024, 2:22 P..., Failed

6. Select Advanced and next

RFP Demo with Vector Databases and Prompt Builder copie



7. Choose DMO

Search Index Builder New Configuration

Guided Setup

Select source DMO

Select a data model object for Search Index configurations.

Data Space: default

Search data model objects...

Data Model Object Name: RFP_Question_and_Answers_sfdo_org

Object API Name: RFP_Question_and_Answers_sfdo_org_dim

Frequently Asked Questions

What Type of Object Should I Index?

Create search index configurations for objects with text blob fields. For example, object configurations such as Service Knowledge Articles or text documents stored in an external blob store, like Amazon S3.

8. Click Next

RFP Demo with Vector Databases and Prompt Builder copie

9. Click Manage Fields and select the fields for chunking and click save

10. Click Next

RFP Demo with Vector Databases and Prompt Builder copie

Select Fields to Chunk

Set the chunking strategy for each field in the index configuration. Click to add or remove fields as needed.

2/4 Fields selected

Field Label	Field API Name	Data Type	Chunking Strategy
Answers	Answers_c_c	Text	Passage Extraction
Question	Question_c_c	Text	Passage Extraction

Frequently Asked Questions

Which Fields Should I Chunk?

Chunk fields with dense, semantically rich content. For example, chunk the custom text fields of a Knowledge Article by HTML tags using the Passage Extraction strategy.

Which Chunking Strategy Should I Use?

Select your chunking strategy based on the content you're working with. For example, consider the fields that provide the most relevant context and semantic meaning for your application's purpose, and how you expect the retrieved results to be used in your Einstein application.

11. Click Next

Select a Vectorization Strategy

Select a vectorization strategy based on your embedding model. This strategy determines how Data Cloud measures your unstructured data for semantic relevance when building search results.

Embedding model: E5 Large V2 Embedding Model

Dimension: 1,024

Max Token Limit: 512

Index: HNSW

hnswEfConstruction: 8

M: 4

Similarity Metric: COSINE

Frequently Asked Questions

What is Vectorization?

Vectorization is the process of embedding your unstructured textual data into numerical representations of data that can be compared. Vector embeddings are used to measure the semantic closeness of different pieces of text to create accurate and relevant results in your prompts and searches.

12. If required add filters

RFP Demo with Vector Databases and Prompt Builder copie

13. Review and Save

14. Once indexing is completed you will see two new DMOs created

RFP Demo with Vector Databases and Prompt Builder copie

Object Label	Object API Name	Data Streams	Data Lake Objects	Data Space	Type	Status
1 RFP_Question_and_Answers_sfdc_org	RFP_Question_and_Answers...	RFP_Question_and_Answers_sf...	RFP_Question_and_Answers_sf...	default	Custom	Ready
2 RFP_Question_and_Answers_sfdc_org chunk	RFP_Question_and_Answers...	RFP_Question_and_Answers_sf...	RFP_Question_and_Answers_sf...	default	Semantic	Ready
3 RFP_Question_and_Answers_sfdc_org vector	RFP_Question_and_Answers...	RFP_Question_and_Answers_sf...	RFP_Question_and_Answers_sf...	default	Semantic	Ready

15. Navigate to the query editor and enter the following query

```
select v.Score__c Score__c, r.Id__c Id__c, r.Question_c__c Question_c__c, r.Answers_c__c Answers_c__c
FROM vector_search(table(RFP_Question_and_Answers_sfdc_org_vector__dlm),
'How does your CDP handle data cleansing?', '', 2000) v
JOIN RFP_Question_and_Answers_sfdc_org_chunk__dlm rc
ON v.RecordId__c = rc.RecordId__c
JOIN RFP_Question_and_Answers_sfdc_org__dlm r ON
rc.SourceRecordId__c = r.Id__c ORDER BY Score__c DESC LIMIT 5
```

This query fetches all the questions and answers which are very similar to the question 'How does your CDP handle data cleansing?'

16. Create apex class

```

public class SearchContext {

    @InvocableMethod(CapabilityType = 'PromptTemplateType://einstein_gpt__fieldCompleter')
    public static List<Response> searchforContext(List<Request> requests) {
        List<Response> responses = new List<Response>();
        Response response = new Response();

        String topic = requests[0].RelatedEntity.Question__c;

        ConnectApi.CdpQueryInput input = new ConnectApi.CdpQueryInput();
        input.sql = 'select v.score__c Score__c, r.Id__c Id__c, r.Answers_c__c Answers_c__c FF
        ConnectApi.CdpQueryOutput output = ConnectApi.CdpQuery.queryANSISql(input);
        List<Object> data = output.data;
        String scs = '';
        List<List<Results>> allResults = new List<List<Results>>();
        List<Results> returnResult = new List<Results>();
        for (Object searchRecord : data) {
            Map<String, Object> myMap = (Map<String, Object>) JSON.deserializeUntyped(searchRecord);
            // check for access of case record for the current user
            if (SearchContext.getUserRecordAccess((String) myMap.get('Id__c'))) {
                Map<String, String> sc = new Map<String, String>();
                //sc.put('Id', (String) myMap.get('Id__c'));
                //sc.put('Similar_Case__c', (String) myMap.get('Id__c'));
                // sc.put('Summary__c', (String) myMap.get('Summary__c'));
                sc.put('Answers_c__c', String.valueOf(myMap.get('Answers_c__c')));
                //sc.put('Summary_c', String.valueOf(myMap.get('Summary__c')));
                scs = scs + JSON.serialize(sc);
            }
        }
        scs = '[' + scs + ']';

        response.Prompt = scs;
        responses.add(response);
        return responses;
    }

    public static boolean getUserRecordAccess(String recordId) {
        return true;
    }

    // All inputs will be passed into the IA
    public class Request {
        @InvocableVariable
        public RFP_Question_and_Answers__c RelatedEntity;
    }

    public class Response {
        @InvocableVariable
        public String Prompt;
    }

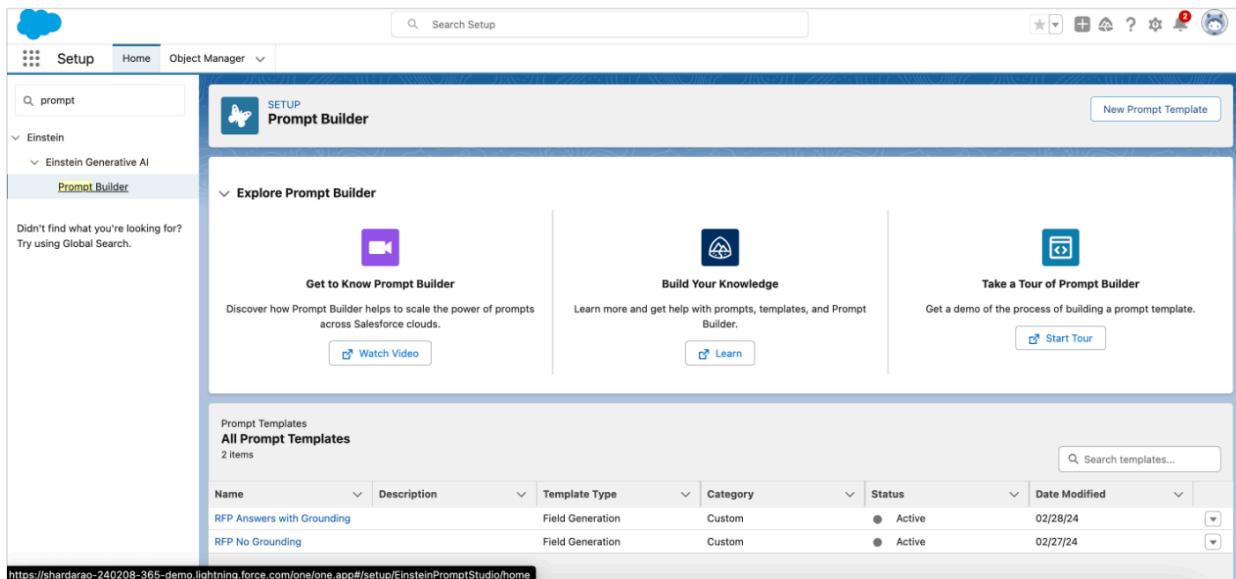
    public class Results {
        @InvocableVariable
    }
}

```

RFP Demo with Vector Databases and Prompt Builder copie

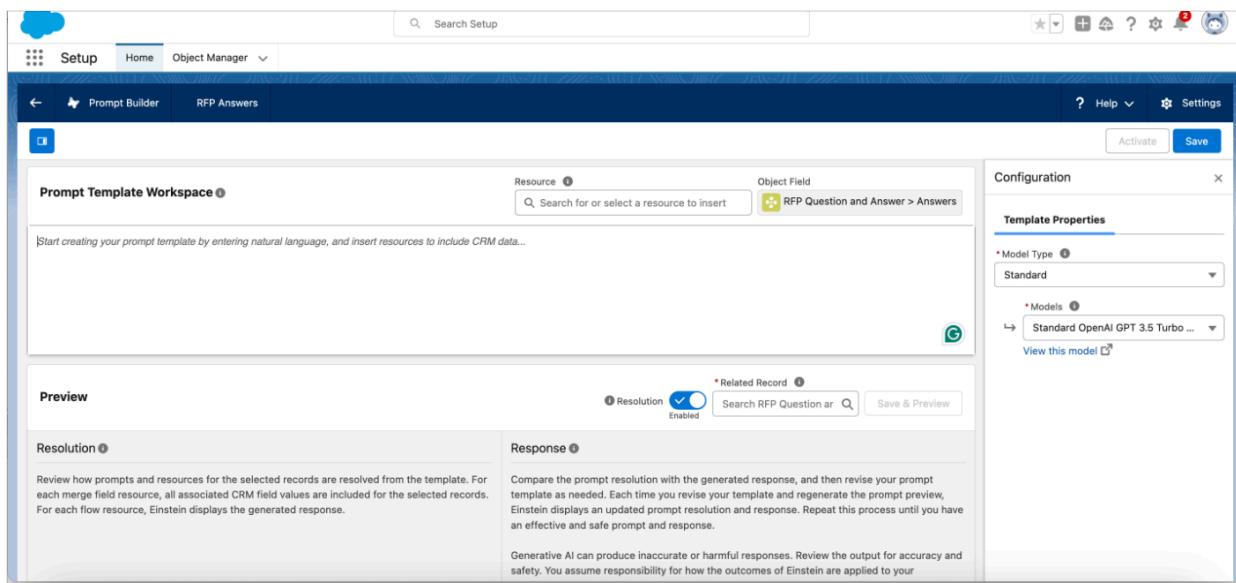
```
public String Answer;  
@InvocableVariable  
public String Question;  
  
}  
  
}
```

16. Navigate to setup → prompt builder → new prompt template



The screenshot shows the Salesforce Setup interface. In the left sidebar, under 'Einstein', 'Prompt Builder' is selected. The main content area is titled 'SETUP Prompt Builder' and contains three sections: 'Explore Prompt Builder' (with links to 'Get to Know Prompt Builder', 'Build Your Knowledge', and 'Take a Tour of Prompt Builder'), 'Prompt Templates' (listing 'All Prompt Templates' with two items: 'RFP Answers with Grounding' and 'RFP No Grounding'), and a search bar for templates.

17. Enter the name, primary object, and the field for which you would like to generate content



The screenshot shows the 'Prompt Template Workspace' in the Einstein Prompt Studio. It includes a 'Resource' search bar, an 'Object Field' dropdown set to 'RFP Question and Answer > Answers', and a 'Configuration' panel on the right with 'Template Properties' (Model Type: Standard, Model: Standard OpenAI GPT 3.5 Turbo). The workspace itself has a 'Preview' section showing a table with 'Resolution' and 'Response' columns, and a note about generating AI responses.

18. Draft prompt and specify resource

RFP Demo with Vector Databases and Prompt Builder copie

The screenshot shows the Salesforce Prompt Builder workspace. In the top right corner, there are standard navigation and configuration buttons: Help, Settings, Activate, and Save. The main area is divided into several sections:

- Prompt Template Workspace**: A text input field containing the question input: `Input:RFP_Question_and_Answers__c.Question__c`. Below it is a note: "Please make use of this information while answering the question `Apex:SearchContext`".
- Resources**: A list of available resources: Apex, Current Organization, Current User, and RFP Question and Answer.
- Object Field**: A dropdown menu set to "RFP Question and Answer > Answers".
- Configuration**: A sidebar on the right with "Template Properties" settings:
 - Model Type**: Standard
 - Models**: Standard OpenAI GPT 3.5 Turbo ...
- Preview**: A section showing the generated prompt. It includes fields for "Resolution" (Enabled) and "Related Record" (set to "RFP-0004"). The preview text is identical to the workspace input.
- Notes**: A note at the bottom of the preview section: "Generative AI can produce inaccurate or harmful responses. Review the output for accuracy and safety. You assume responsibility for how the outcomes of Einstein are applied to your organization."

19. Save template, test prompt by selecting sample related record.

This screenshot is nearly identical to the previous one, showing the same workspace, resources, and configuration settings. The primary difference is in the "Preview" section, where the "Related Record" dropdown now shows "RFP-0004" instead of "Enabled". This indicates that the template has been saved with a specific record selected.

20. Activate template

RFP Demo with Vector Databases and Prompt Builder copie

The screenshot shows the Salesforce Prompt Builder interface. In the top navigation bar, there are tabs for 'Setup', 'Home', 'Object Manager', 'Prompt Builder' (which is selected), 'RFP Answers', and 'Version 1'. The main area is titled 'Prompt Template Workspace'. It contains a message: 'I want you to answer this question Input: `Input:RFP_Question_and_Answers__c.Question__c`. Please make use of this information while answering the question `Apex: SearchContext`'. Below this is a 'Resource' search bar and a 'Search for or select a resource to insert' button. To the right is a 'Configuration' panel with 'Activate Configuration' and 'Template Properties' sections. The 'Model Type' is set to 'Standard'. Under 'Models', it lists 'Standard OpenAI GPT 3.5 Turbo ...'. A 'Preview' section shows a resolution and response. The resolution includes a note about CDP handling data cleansing. The response is a thank you message about the CDP's capabilities.

21. Create flow to surface Gen AI responses

The screenshot shows the Salesforce Flow Builder interface. The flow is titled 'Display Prompt RFP - V15'. It consists of the following steps: 'Screen Flow Start', 'getRecords' (Get Records from Salesforce CRM), 'prompt Action', 'Display Screen', 'New Annotated Response Assignment', 'Update Answer Update Records', and 'End'. On the right side, there is a modal window titled 'Edit RFP Answers with Grounding prompt (prompt)'. It contains a sub-section 'Set Input Values for the Selected Action' with a field 'RelatedEntity' containing '{recordId}' and a toggle switch 'Don't Include' which is turned off.

21. Create an apex class to find similar question and answers

```
global with sharing class SearchSimilarQandA {
    @InvocableMethod(label='Search Similar Questions' callout=true description='Perform a search on vector database')
    public static List<Results> query(List<Requests> requests) {
        List<Results> results = new List<Results>();
        Results result = new Results();

        String topic = requests[0].topicOne;
        System.debug('Searching Vector Database for topic: ' + topic);

        ConnectApi.CdpQueryInput input = new ConnectApi.CdpQueryInput();
```

RFP Demo with Vector Databases and Prompt Builder copie

```
input.sql = 'select v.score__c Score__c, r.Id__c Id__c, r.Question_c__c Question_c__c, r.Answers_c__c Answers_c__c from RFP_Question_and_Answers__c r inner join vector_index v on r.Id__c = v.id where r.Question_c__c like :topicOne';

ConnectApi.CdpQueryOutput output = ConnectApi.CdpQuery.queryANSISql(input);
List<RFP_Question_and_Answers__c> questionAnswers = new List<RFP_Question_and_Answers__c>();

for (Object searchRecord : output.data) {
    Map<String, Object> myMap = (Map<String, Object>) JSON.deserializeUntyped(searchRecord);

    // check for access of case record for the current user
    if (getUserRecordAccess((String) myMap.get('Id__c'))) {
        RFP_Question_and_Answers__c qa = new RFP_Question_and_Answers__c();
        qa.Question__c = String.valueOf(myMap.get('Question_c__c'));
        qa.Answers__c = String.valueOf(myMap.get('Answers_c__c'));
        questionAnswers.add(qa);
    }
}

result.searchResults = questionAnswers;
results.add(result);
return results;
}

public static boolean getUserRecordAccess(String recordId) {
    // Implement your logic for checking user access
    return true;
}

global class Requests {
    @InvocableVariable(label='Topic' description='Topic to search for' required=true)
    global String topicOne;
}

global class Results {
    @InvocableVariable(label='Search results' description='Search results of your query')
    global List<RFP_Question_and_Answers__c> searchResults;
}
}
```

21. Create flow to surface the similar question and answers

RFP Demo with Vector Databases and Prompt Builder copie

