Einstein Prompt Builder Hands-on Workshop

See What You Can Do With an LLM

Attendee Guide

Version 022724

http://sfdc.co/TDX24PromptBuilderWorkbook

Disclaimer

The Salesforce products and features referenced in this workshop guide are subject to change at the discretion of Salesforce.com. Mention of specific products or features is intended for discussion purposes only and does not represent a guarantee of their availability. Please refer to Salesforce.com for the most current information on available products, editions, and functionality. We apologize for any confusion arising from discrepancies between products mentioned in this guide and the latest official Salesforce information. Please reach out to us directly with any questions.

Table of Contents

Disclaimer ---- 1
Table of Contents ---- 2
Scenario ---- 3
Exercise 1 - The Simple Prompt ---- 4
Overview ---- 4
Step 1: Turn on Einstein ---- 4
Step 2: Create A Prompt Builder Template ---- 5
Step 3: Test the Sales Check In Prompt Template ---- 9
Exercise 2 - Einstein AI Case Response ---- 11
Overview ---- 11
Step 1: Turn on Einstein ---- 12
Step 2: Create a Template-Triggered Prompt Flow ---- 13
Step 3: Create a Prompt Template ---- 18
Step 4: Associate the Prompt Template to the AI Analysis field ---- 21
Step 5: Test Your AI Case Response ---- 23
Exercise 3 - Testing Sentiment ---- 25
Overview ---- 26
Step 1: Turn on Einstein ---- 27
Step 2: Create the initial Prompt ---- 28
Step 3: Create the Flow ---- 31
Step 4: Add the Flow to our Prompt ---- 36
Step 5: Use the Results for Automation (optional) ---- 38
Working with Variables ---- 43
Adding Logic to Our Flow ---- 44
Display the Results ---- 47
Test and Debug ---- 49
Add to the Interface ---- 50
What if I Get the Activation Dialog? ---- 52
Extra Credit: Expand the Flow ---- 54
Want to learn more? ---- 55
Trailhead ---- 55
Documentation ---- 55
Where Do We Go From Here? ---- 56

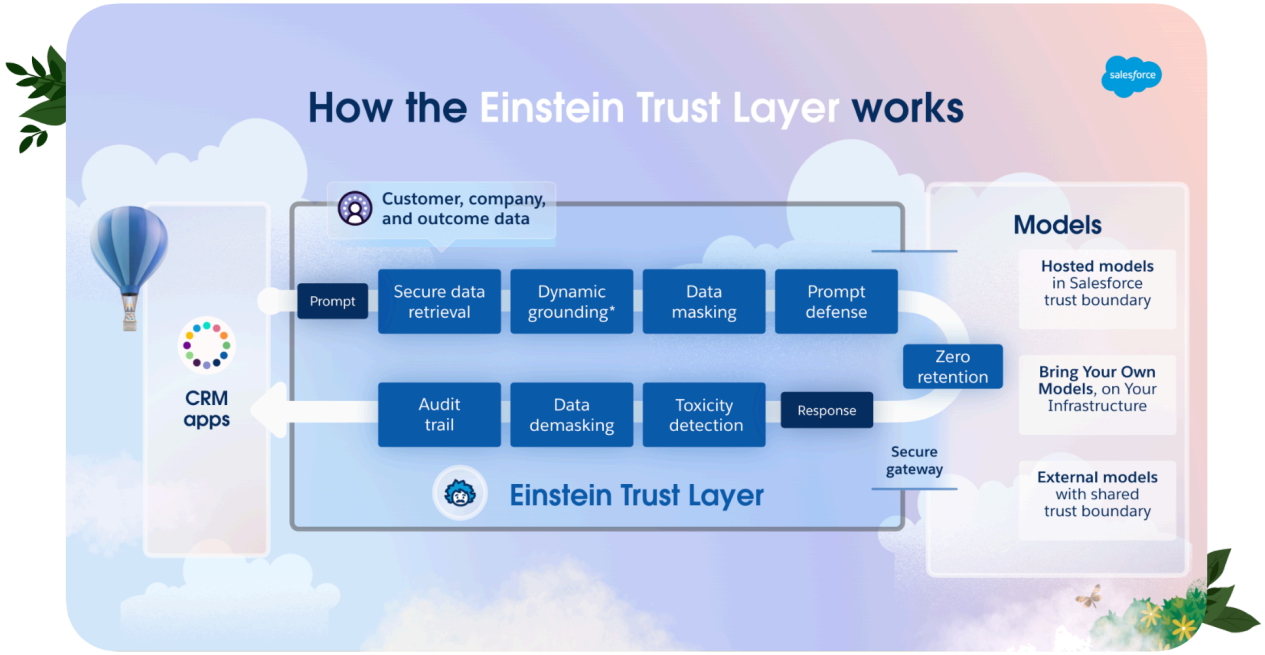
Scenario

Welcome to our introductory workshop on using Salesforce's new **Prompt Builder** feature. In this hands-on session, we will walk through how to leverage Prompt Builder to automatically create dynamic prompts that can be sent to large language models (LLMs).

As administrators at the fictional Sprungfield Nuclear Power Plant, we will use examples relevant to our duties like looking up employee records, scheduling, and plant maintenance. You will get to try building sample prompts and see firsthand how Prompt Builder allows you to tap into AI without hard-coding responses. Our goal is that by the end of this workshop, you will feel confident using Prompt Builder to improve efficiencies and operations at your own organization through natural language interactions.

The exercises today will provide real-world practice with this innovative tool to make AI more accessible. As you review the image below, we will be focusing on **Secure Data Retrieval** and **Dynamic Grounding**.

We hope you enjoy this hands-on experience with Prompt Builder!



Exercise 1 - The Simple Prompt

Salesforce's Einstein Prompt Builder is an AI-powered tool that enables the easy creation of customized text prompts without needing data science expertise. You simply describe in plain language the desired content such as customer success stories, FAQs or marketing messages. Einstein then suggests appropriate tones, length and other attributes to craft impactful prompts.

It's part of Salesforce's broader Einstein AI capabilities woven throughout its products to automate and enhance through predictive insights. With Prompt Builder, you can boost their content with the accessibility and intelligence of AI, no advanced data skills required. It makes language AI achievable and actionable across functions.

Overview

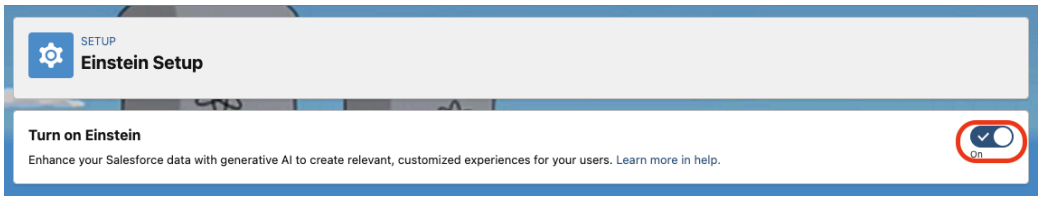
As a Sprungfield Nuclear Power Sales Rep you've just reviewed your Monthly Account Check-Ins report. You noticed Homer Sampson's account is past due and clicked into the record for further review.

As you review Homer's account you decide to use Einstein AI to generate an email to Homer with payment options and also upsell products.

Step 1: Turn on Einstein

Before you can use Einstein **Prompt Builder** in Salesforce, make sure Einstein is activated for your organization. Prompt Builder relies on AI capabilities that are only enabled once Einstein is turned on. Use the following steps to activate Einstein for your organization.

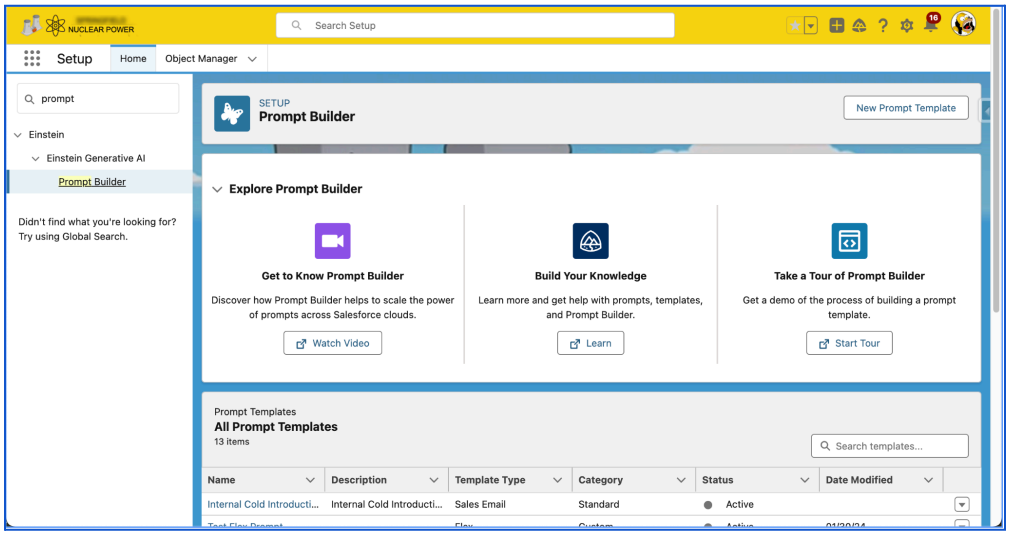
- ☐ Click the **Setup** icon and select **Setup**.
  This will take you to the Setup page for your org
- ☐ In the **Quick Find**, type **Einstein Setup** then select **Einstein Setup**.
- ☐ Toggle the **Turn on Einstein** switch.



Step 2: Create A Prompt Builder Template

- ☐ Click the **Setup** icon and select **Setup**
- ☐ Type **Prompt** in the **Quick Find**
- ☐ Select **Prompt Builder** under the **Einstein Generative AI** group

Salesforce displays the Prompt Builder list including options to explore this feature



- ☐ Click the New Prompt Template button to create a new **Prompt Template**

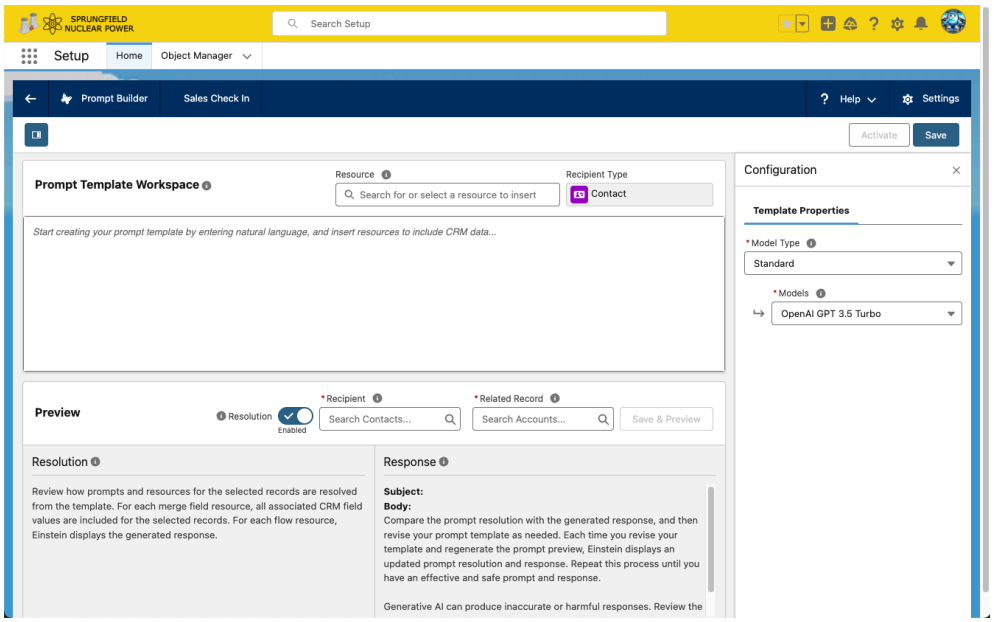
Salesforce opens the New Prompt Template dialog

- ☐ Enter the following:

Field	Value	Reason
Prompt Template Type	Sales Email	This prompt is going to use a Flow as a data source.
Prompt Template Name	Sales Check In	Required field
API Name		Automatically generated.
Template Description	Prompt to check in with account	Descriptions help
Define Resources		
Recipient	Contact	
Related Object	Account	

- ☐ Click the Next button

Salesforce automatically displays the Prompt Builder interface



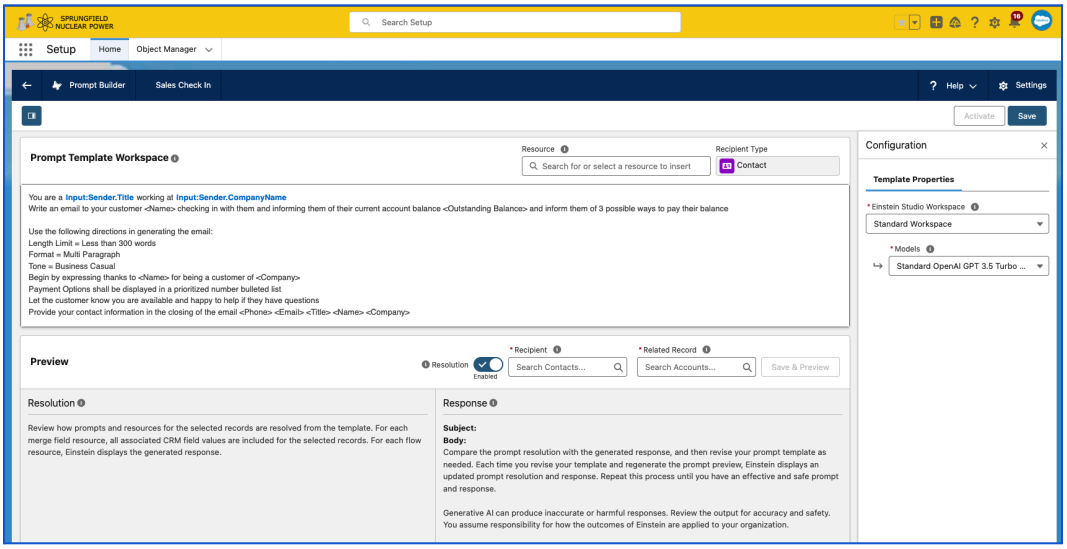
☐ Enter the following prompt in the **Prompt Editor**

```
You are a <Title> working at <Company>
Write an email to your customer <Name> checking in with them
and informing them of their current account balance
<Outstanding Balance> and inform them of 3 possible ways to
pay their balance

Use the following directions in generating the email:
Length Limit = Less than 300 words
Format = Multi Paragraph
Tone = Business Casual
Begin by expressing thanks to <Name> for being a customer of
<Company>
Payment Options shall be displayed in a prioritized number
bulleted list
Let the customer know you are available and happy to help if
they have questions
Provide your contact information in the closing of the email
<Name> <Title> <Company> <Email> <Phone>
```

☐ Now we will substitute dynamic values for the <placeholder> keywords

☐ Highlight **<Title>**, delete the text and click **Resource > Sender > Title**

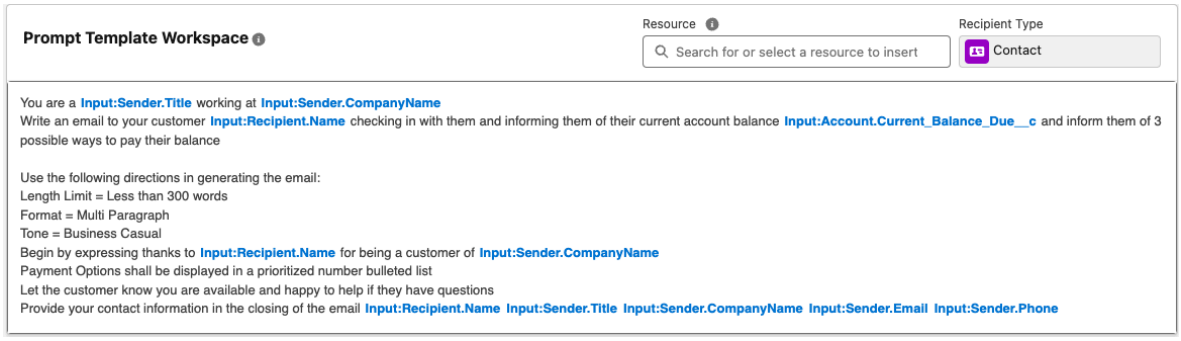
☐ Do the same for **<Company>** replacing it with **Sender > CompanyName**



☐ Highlight **<Name>**, delete the placeholder and click **Resource > Recipient > Full Name**

☐ Replace **<Outstanding Balance>** with **Resource > Account > Current Balance Due**

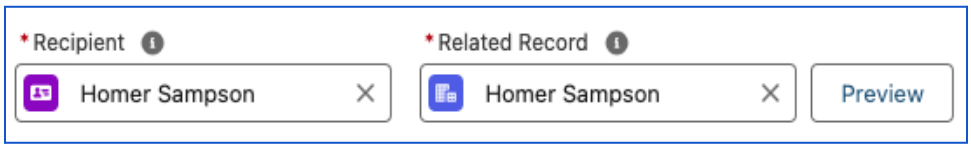
☐ Continue replacing the placeholders with dynamic values from Sender, Recipient and Account; e.g. "...your contact information..." will pull from **Resource > Sender >**



Tip: After you select a resource e.g. Recipient, you can enter the name of the desired field in the search box to locate and insert it into the template. Helpful when objects have many fields

☐ Click the Save button
Salesforce saves the current prompt setup and deactivates the Save button

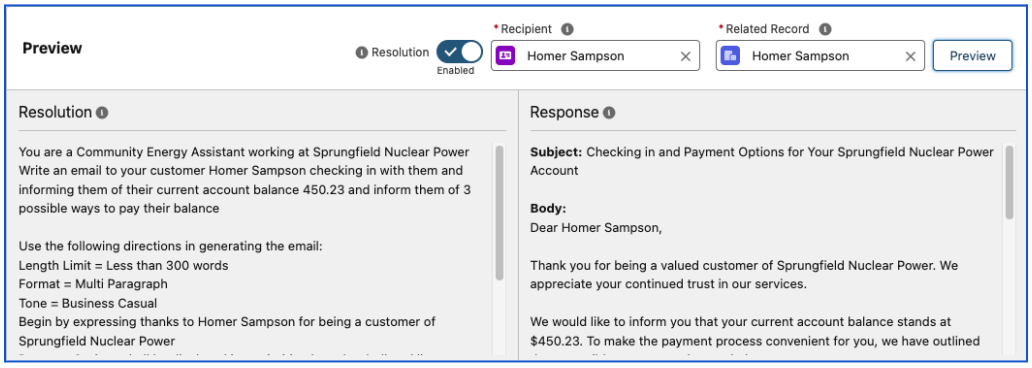
☐ Test your prompt by entering **Homer** in the **Recipient** search box, selecting Homer, entering **Homer** in the Related Record and selecting Homer



Note: Homer is a **person account**, thus requiring entering Homer in both fields

☐ Click the Preview button

☐ Review the **Resolution**, which is the dynamically grounded prompt that will be used to generate the response, which is the email that will be sent to the customer



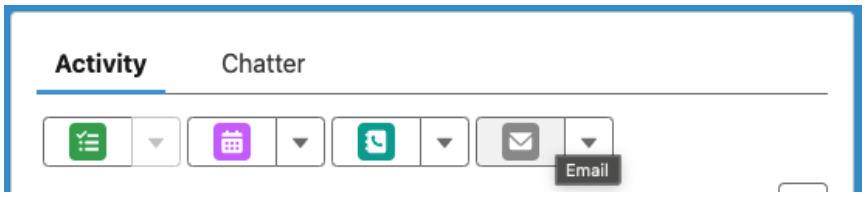
☐

Notice: In the Resolution, the resources you inserted are replaced (dynamically grounded) with actual values from Salesforce

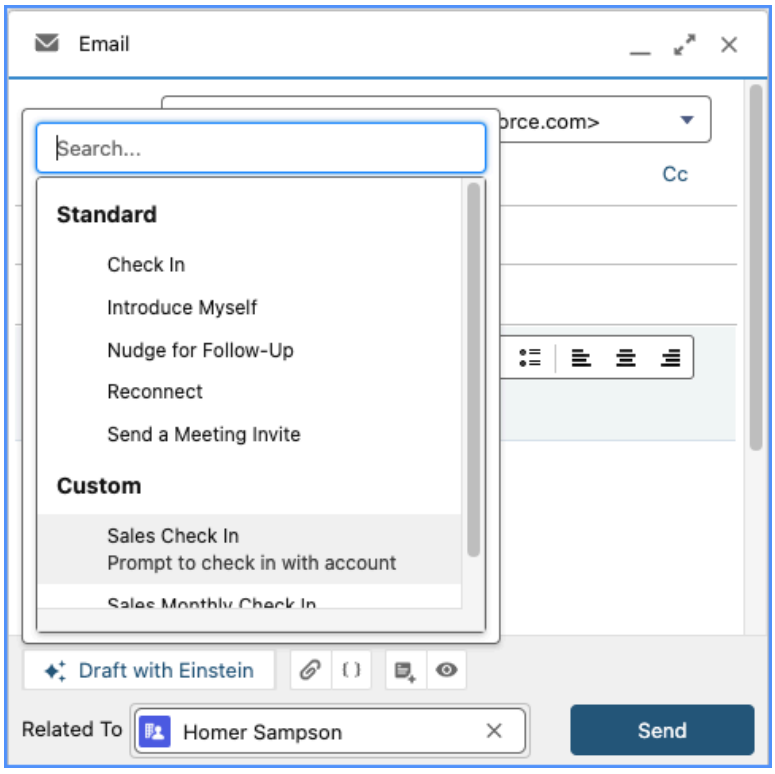
☐ Click the Activate button to activate the new prompt
Salesforce activates the prompt and makes it available for use.

Step 3: Test the Sales Check In Prompt Template

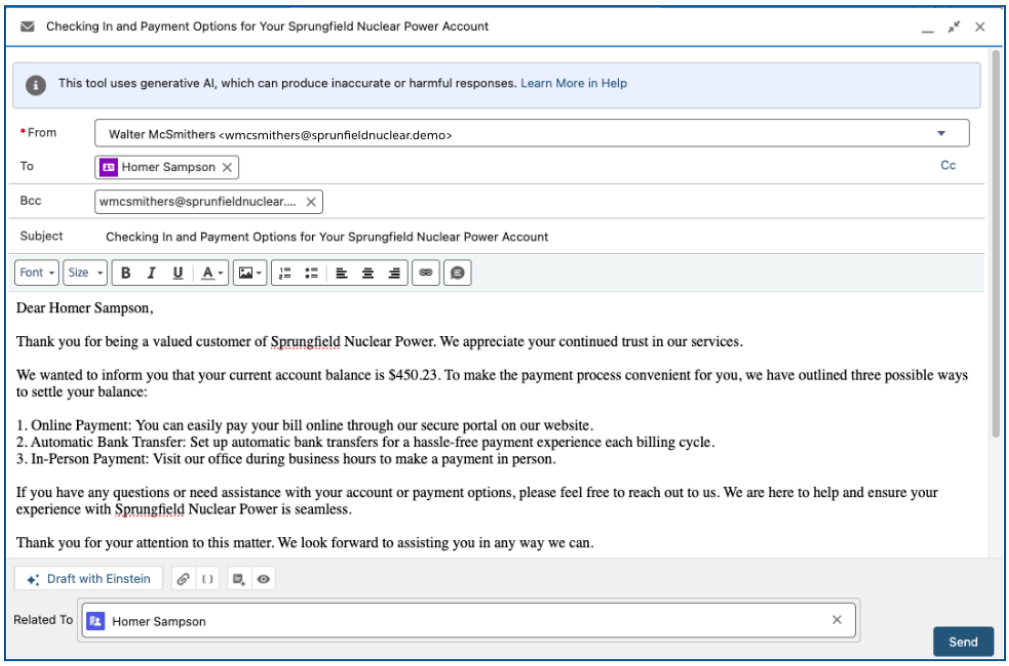
- ☐ In **Sales Lightning** app, open Homer's record
- ☐ In **Activity**, click **Email** to create an email to Homer



- ☐ Click **Draft with Einstein** and select **Sales Check In** from the **Custom** section



- ☐ Select **Homer Sampson** for the Account and click Continue
- ☐ Click the  to pop out the email message for easier reading



- ☐ Notice the email has everything we described in our template:
  - ☐ Thanking the customer for their business
  - ☐ The dynamically grounded data
  - ☐ The payment options in list form and more.

Exercise 2 - Einstein AI Case Response

This module demonstrates how Prompt Builder provides us a simple way to generate required information for a user that greatly enhances both the user's experience and the customer experience.

Overview

A Sprungfield customer calls or sends emails regarding a high electric bill.

The agent requires a callout to Sprungfield power station's SAP system to get a billing summary. This system was last updated in the 80's and requires complex JSON.

The agent also requires seeing weather data coming from a 3rd-party weather system.

We will utilize Einstein's generative AI capabilities to bring together the needed information quickly and accurately.

Perform the steps on the following pages.

Assumptions. These items exist within the org used for the workshop:

- ☐ The **AI Analysis** Long Text Area field has been created on the **Case** object.
- ☐ **Dynamic Actions** has been enabled on the Case page layout.
- ☐ **Get Weather Data** External Service has been set up, including credentialing.
- ☐ **Make API Callout** Apex Action has been set up.
- ☐ **Case 00001006** has been created.

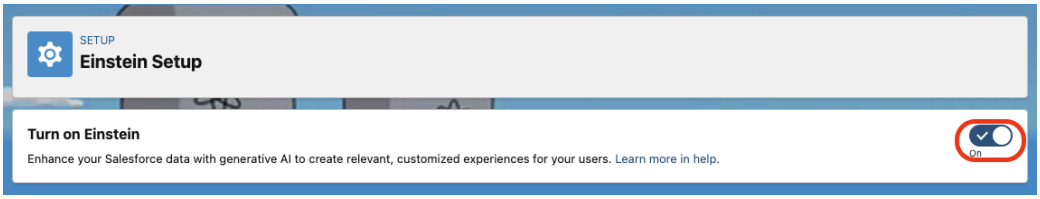
Complete the following steps to setup the AI Case response prompt:

Step 1: Turn on Einstein

Before you can use Einstein **Prompt Builder** in Salesforce, make sure Einstein is activated for your organization. Prompt Builder relies on AI capabilities that are only enabled once Einstein is turned on. Use the following steps to activate Einstein for your organization.

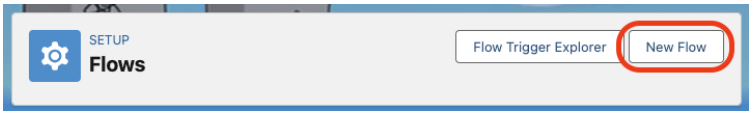
Note: This may already be done by finishing previous exercises.

- ☐ Click the **Setup** icon and select**Setup**.
  This will take you to the Setup page for your org
- ☐ In the **Quick Find**, type **Einstein Setup** then select **Einstein Setup**.
- ☐ Toggle the **Turn on Einstein** switch.

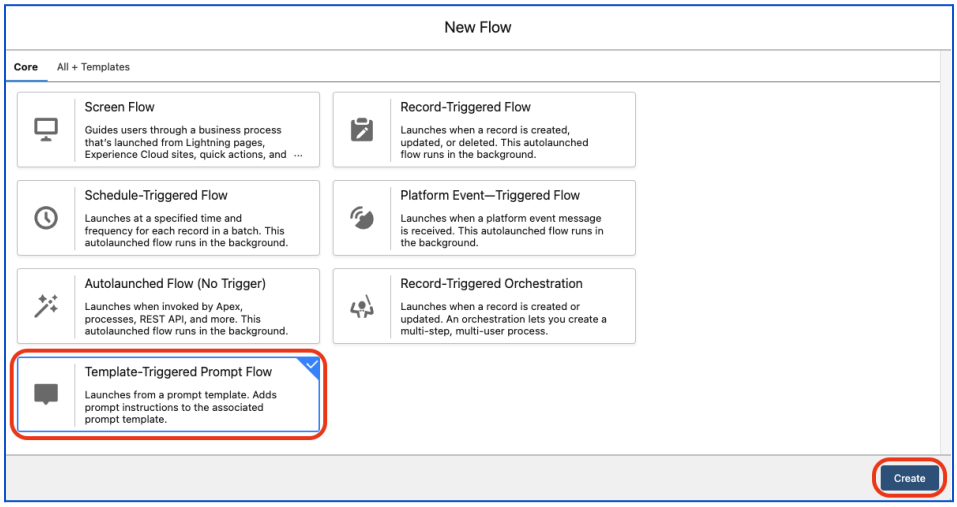


Step 2: Create a Template-Triggered Prompt Flow

- ☐ Click the **Setup** icon and select **Setup**.
  This will take you to the Setup page for your org
- ☐ In the **Quick Find**, type **Flows** and select **Flows**.
- ☐ Click **New Flow**.

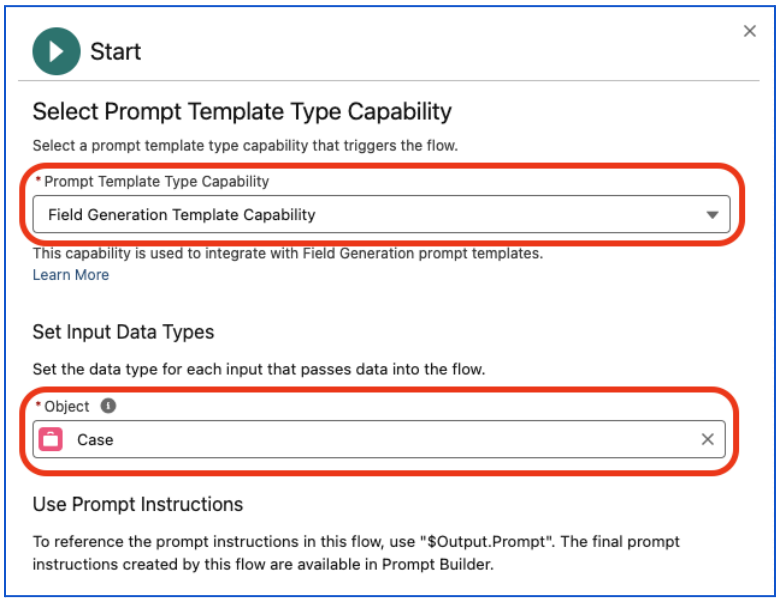


- ☐ Select **Template-Triggered Prompt Flow** and click **Create**.

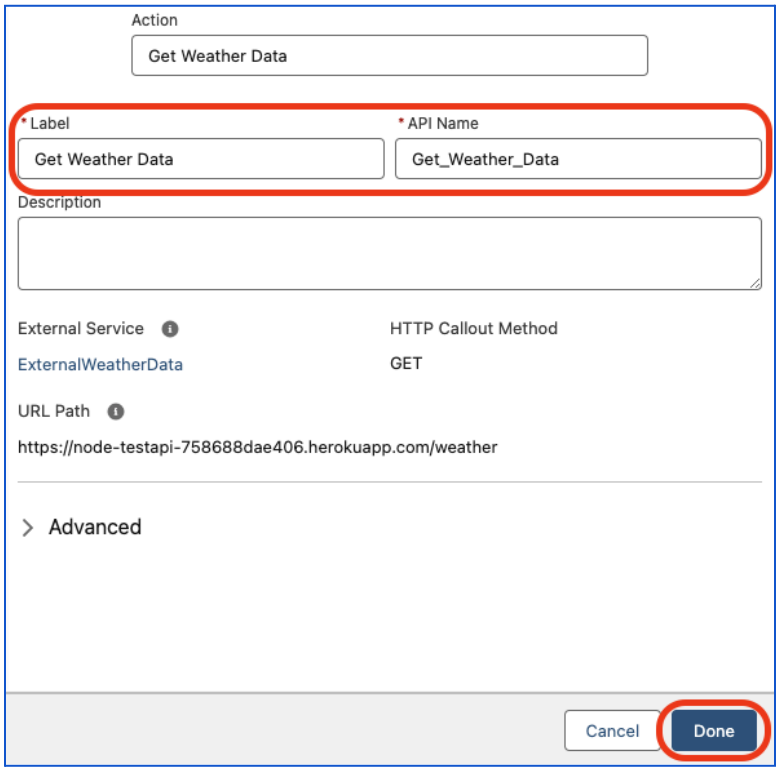


- ☐ For the **Start** > **Prompt Template Type Capability**, select the following:

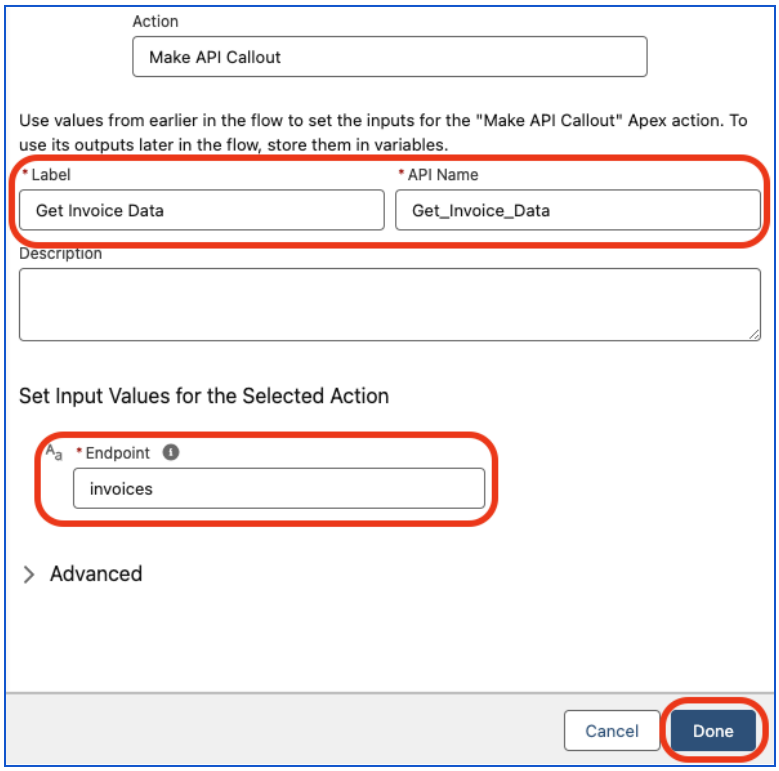
Setting	Value
Prompt Template Type Capability	**Field Generation Template Capability**
Object	**Case**



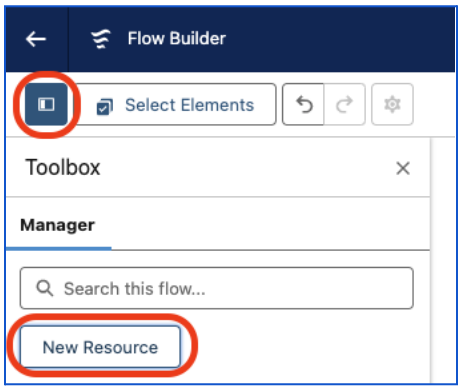
- ☐ Click the  icon to add an element *after* Start and select **Action**.
- ☐ In the **Search all actions...** under **Action**, type **Weather** and select **Get Weather Data**.
  This will select an External Service that has been setup to get weather information.
- ☐ For the action's **Label** enter **Get Weather Data**, then tab to automatically fill in the API Name.
- ☐ Click **Done**.



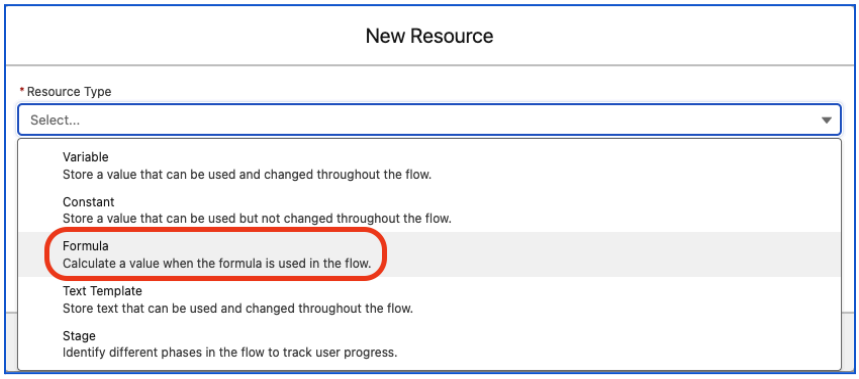
- ☐ Click the ⊕ icon to add an element *after* **Get Weather Data** and select **Action**.
- ☐ In the **Search all actions...** under **Action**, type **API** and select **Make API Callout**.
  This will select an Apex class that has been setup to get invoice data.
- ☐ For the action's **Label** enter **Get Invoice Data** then tab to automatically fill in the API Name. In the **Endpoint** field, enter **invoices**.
  'invoices' is a string passed to the endpoint, no selection box will appear
- ☐ Click **Done**.



- ☐ Click the **Toolbox** icon (in the top left of the Flow Builder), then click **New Resource**.

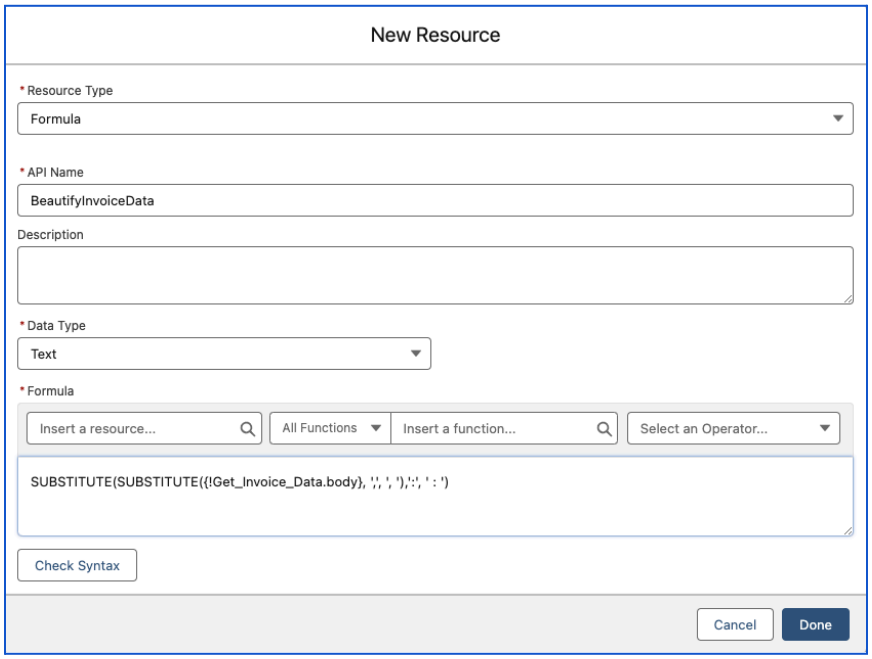


- ☐ In the **Resource Type**, select **Formula**.



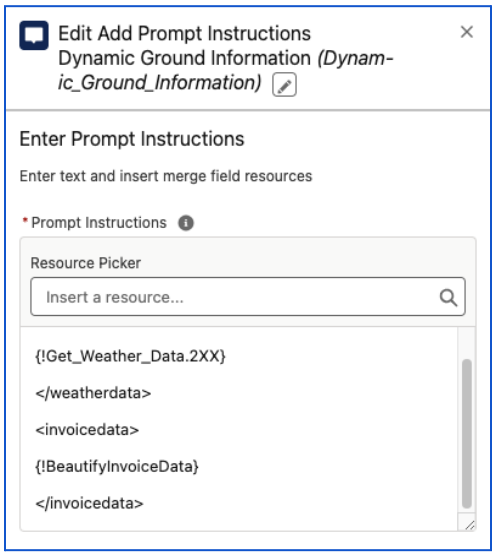
- [ ] In the **API Name** field, enter **BeautifyInvoiceData**. For the **Data Type**, select **Text**. In the **Formula** field, enter the following formula:

```
Unset
SUBSTITUTE(SUBSTITUTE({!Get_Invoice_Data.body}, ',', ', '),':', ' : ')
```

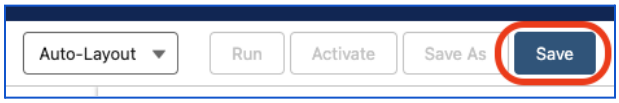


- [ ] Click **Done.**
- [ ] Click the  icon to add an element *after* Get Invoice Data and select **Add Prompt Instructions**.
- [ ] In the **Label** enter **Dynamic Ground Information** then tab to automatically fill in the API Name. In the **Prompt Instructions** field, enter the following:

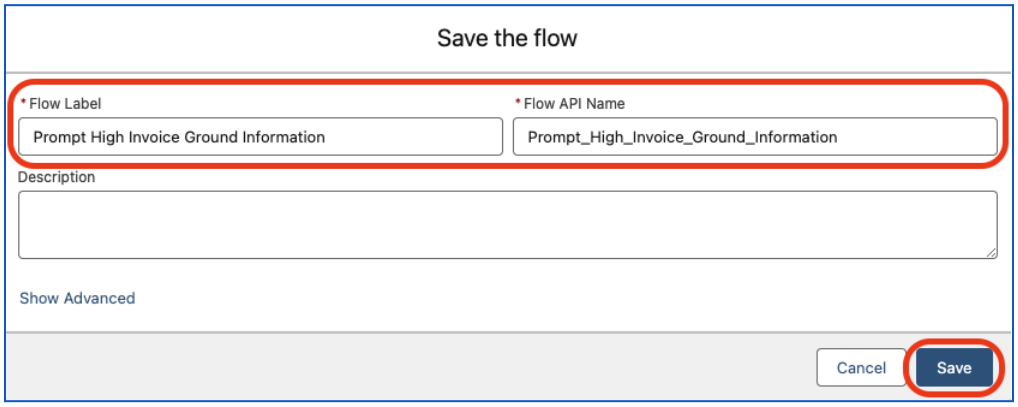
```
Unset
                    <weatherdata>{!Get_Weather_Data.2XX}
                    </weatherdata>
                    <invoicedata>
                    {!BeautifyInvoiceData}
                    </invoicedata>
```



☐ Click **Save** to save the Flow.



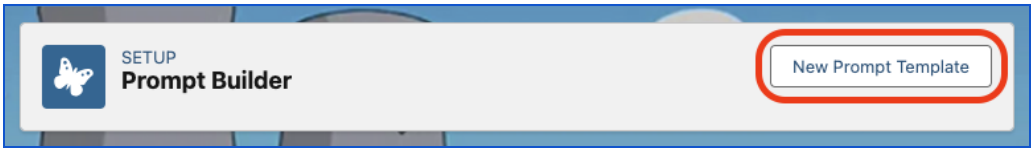
☐ For the **Flow Label**, enter **Prompt High Invoice Ground Information** then tab to automatically fill in the API Name. Click **Save**.



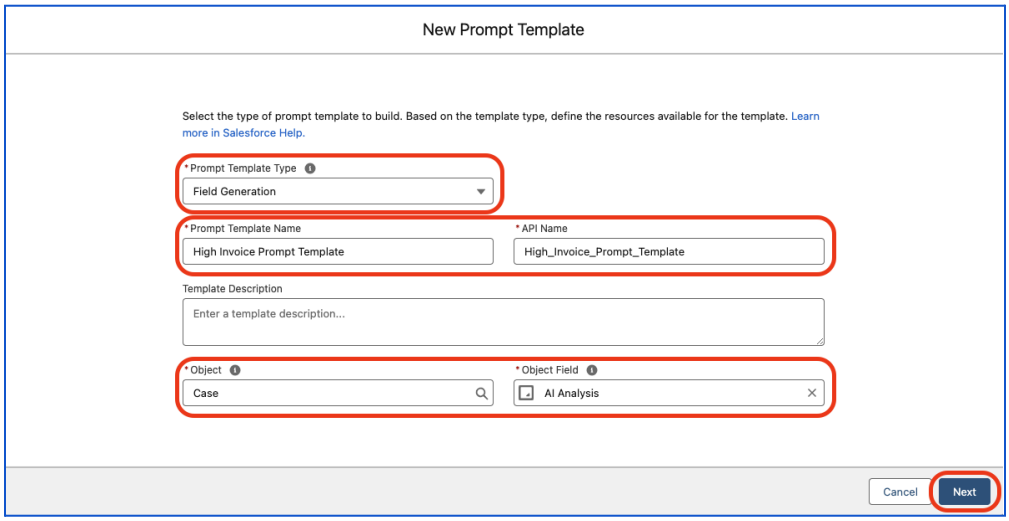
☐ Click the Activate button to activate flow.

Step 3: Create a Prompt Template

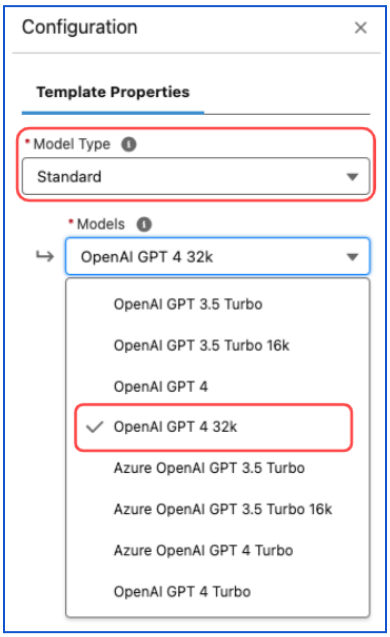
- ☐ Click the **Setup** icon on a previous browser tab and select **Setup**.
  This will take you to the Setup page for your org. You may also already be at the Setup page from a previous step.
- ☐ In the **Quick Find**, type **Prompt** and select **Prompt Builder**.
- ☐ Click **New Prompt Template**.



- ☐ In the **Prompt Template Type**, select **Field Generation**.
- ☐ For the **Prompt Template Name**, enter **High Invoice Prompt Template** then tab to automatically fill in the API Name.
- ☐ In the **Object** field, search for **Case** then select **Case**. *(Note that it may take a few moments for the Object field to be enabled.)*
- ☐ In the **Object Field** field, select **AI Analysis**.



- ☐ Click Next
- ☐ Under **Configuration** > **Template Properties** *(on the right side of the window)*, ensure **Standard** is selected for the **Model Type**.
- ☐ In the **Models** field, select **OpenAI GPT 4 32k**.

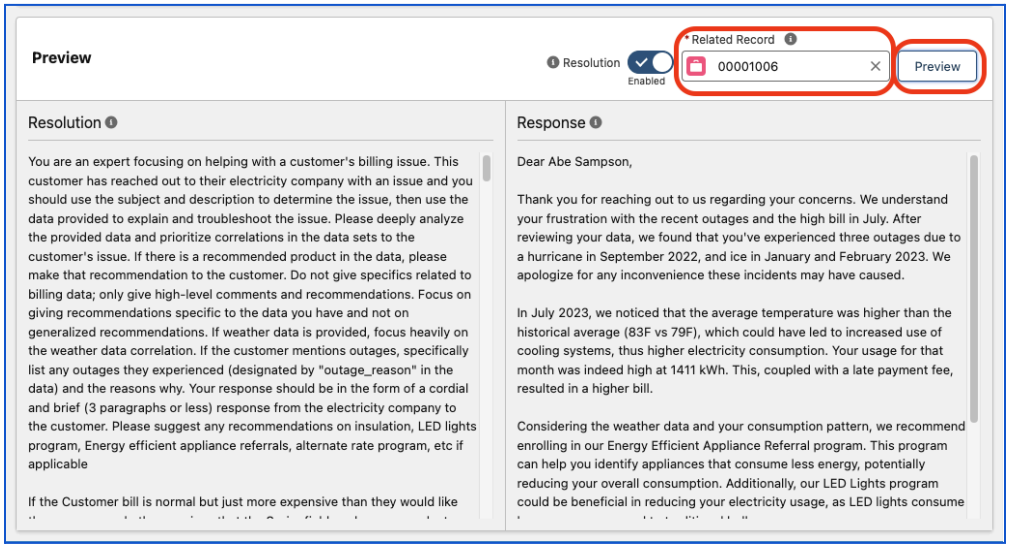


☐ In the **Prompt Template Workspace**, enter the following:

```
Unset
You are an expert focusing on helping with a customer's billing issue. This
customer has reached out to their electricity company with an issue and you
should use the subject and description to determine the issue, then use the
data provided to explain and troubleshoot the issue.

Please deeply analyze the provided data and prioritize correlations in the
data sets to the customer's issue.

If there is a recommended product in the data, please make that
recommendation to the customer. Do not give specifics related to billing
data; only give high-level comments and recommendations. Focus on giving
recommendations specific to the data you have and not on generalized
recommendations.

If weather data is provided, focus heavily on the weather data correlation.

If the customer mentions outages, specifically list any outages they
experienced (designated by "outage_reason" in the data) and the reasons why.
Your response should be in the form of a cordial and brief (3 paragraphs or
less) response from the electricity company to the customer. Please suggest
any recommendations on insulation, LED lights program, Energy efficient
appliance referrals, alternate rate program, etc if applicable
```

```
If the Customer bill is normal but just more expensive than they would like
then recommend other services that the Sprungfield nuclear power plant can
offer (solar power, RTG or RITEG (made with genuine Sprungfield nuclear
waste), district heating service)

Case Subject: {!$Input:Case.Subject}

Description: {!$Input:Case.Description}

Internal comments: {!$Input:Case.Comments}

Customer name: {!$Input:Case.Contact.FirstName}
{!$Input:Case.Contact.LastName}

Name of the agent sending response: {!$User.FirstName} {!$User.LastName}

<data>
{!$Flow:Prompt_High_Invoice_Ground_Information.Prompt}
</data>
```

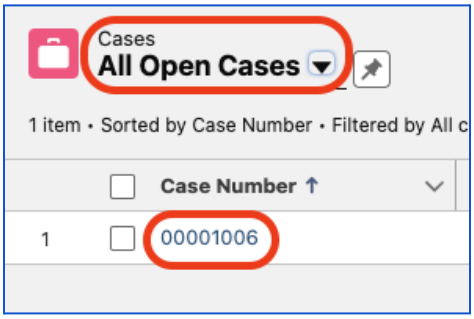
- ☐ Click the **Save** button
  Salesforce saves the current prompt setup and deactivates the Save button
- ☐ In the **Related Record** field, search for **1006** then select **00001006**.
- ☐ Click the **Preview** button
  You should see a response from the Large Language Model (LLM).



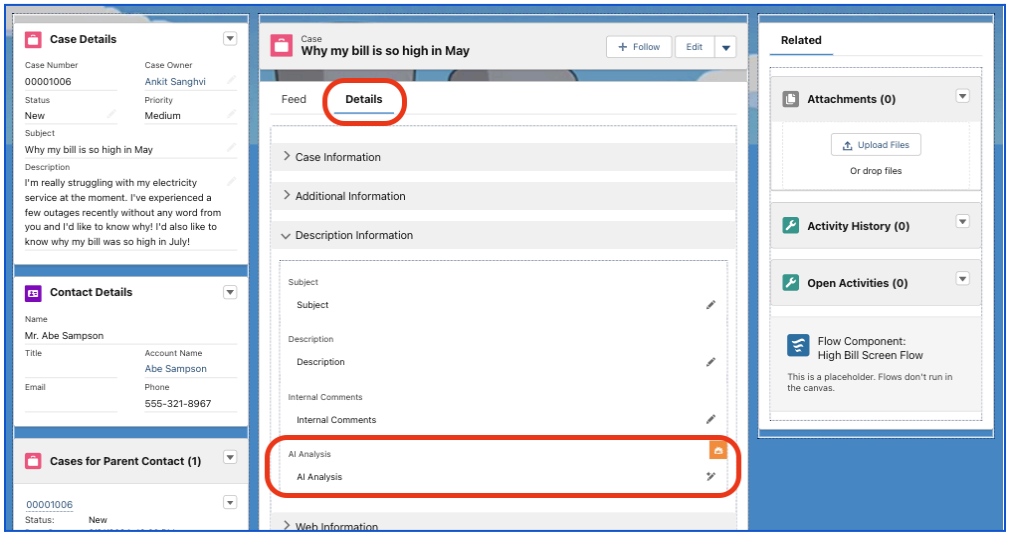
- [ ] Click the Activate button to activate the new prompt
  Salesforce saves and activates the prompt and makes it available for use.

Step 4: Associate the Prompt Template to the AI Analysis field

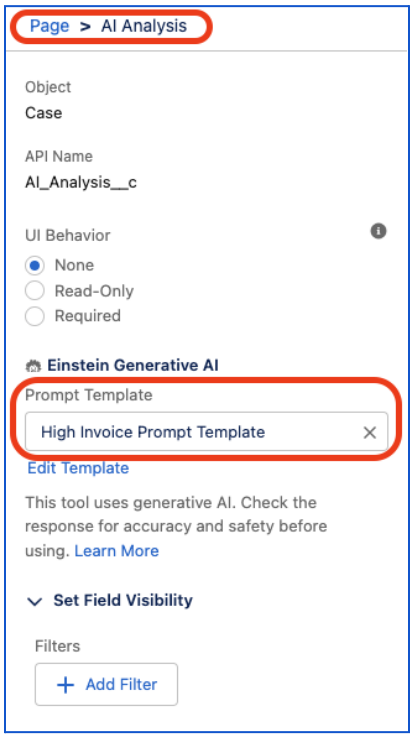
- [ ] Click the **App Launcher** icon and search for **Cases**, then select **Cases**.
- [ ] Change the list view from **Recently Viewed Cases** to **All Open Cases** then select **Case Number 00001006**.



- [ ] Click the **Setup** icon and select **Edit Page**.
  This will take you to the Case Record Page layout
- [ ] Click the **Details** tab then click the **AI Analysis** field.



- [ ] In the **Prompt Template** field under the **Page > AI Analysis** select **High Invoice Prompt Template**.



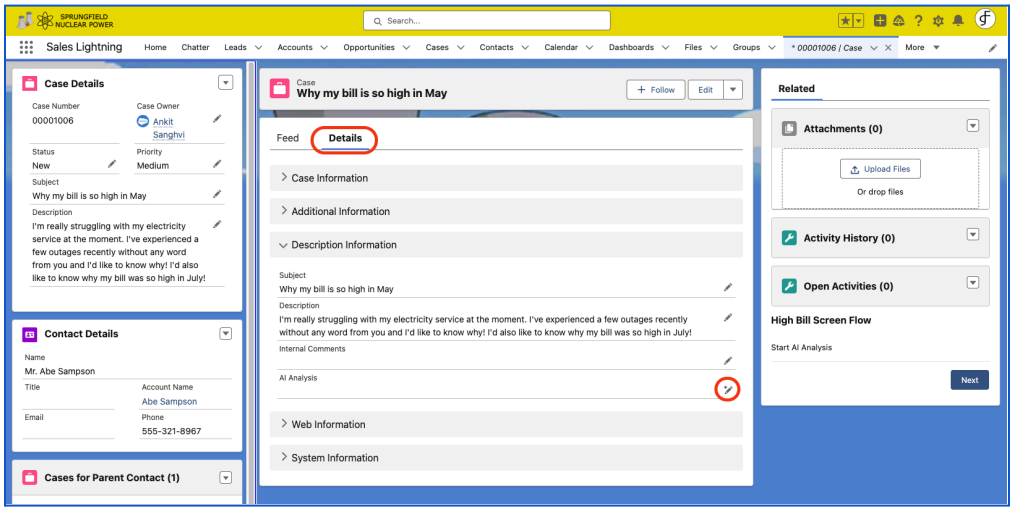
☐ Click the Save button

☐ Click the **Lightning App Builder** ← **Back** icon.

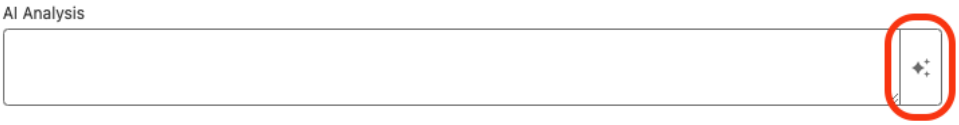
Step 5: Test Your AI Case Response

- ☐ Click the **App Launcher** icon and search for **Cases**, then select **Cases**.
- ☐ Change the list view from Recently Viewed Cases to **All Open Cases** then select **Case Number 00001006**.
  Note: You may already be on Case Number 00001006 from a previous step.

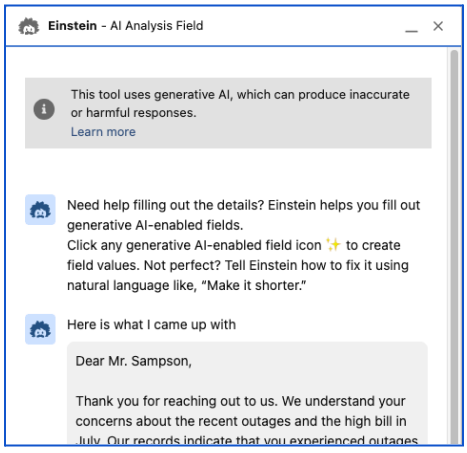
- ☐ Click the **Edit AI Analysis** icon on the *Details > AI Analysis* field.



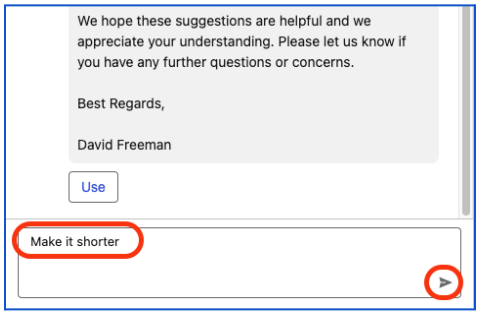
- ☐ Click the **Get Einstein's help in creating this field value.** icon.



Note: This will generate a response in the ' **Einstein - AI Analysis Field**' pop-up.



- [ ] **Type Make it shorter** in the response field and press Enter/Return.
  This will generate a shorter response



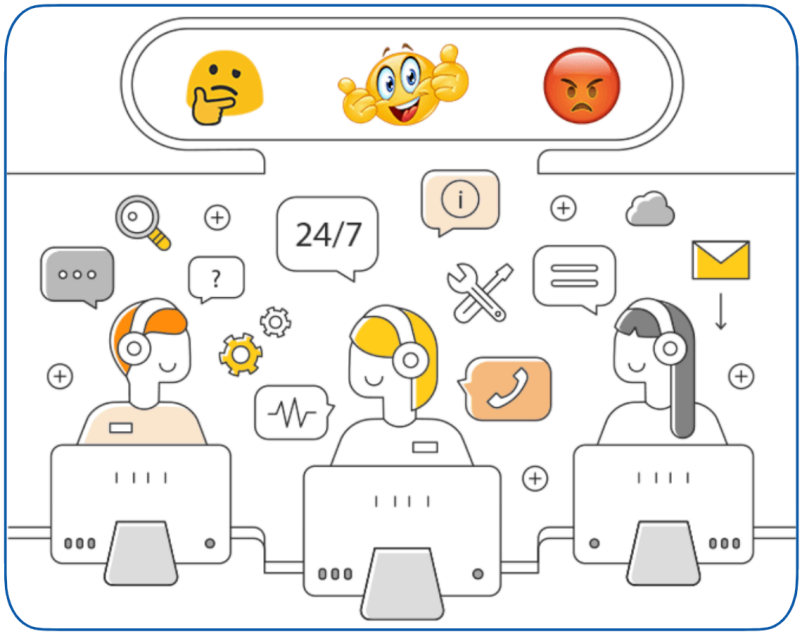
- [ ] *(optional)* Try typing additional responses; e.g. Make it more formal.
- [ ] **Click** Use to accept this response
- [ ] Click the Save button on the case record.

Exercise 3 - Testing Sentiment

Salesforce has offered Natural Language Processing (NLP) capabilities for years, allowing users to interact conversationally with its CRM platform. However, the recent integration of large language models like GPT-3 has greatly expanded what Salesforce can understand and how it responds.

In the past, users were limited to fairly simple, predefined commands when talking to Salesforce AI. Now, with large language models, users can have more free-flowing, natural conversations and ask more complex questions. The AI can parse nuanced prompts and return insightful answers without having every possible response rigidly coded. This allows for more dynamic interactions that feel tailored to the user.

Salesforce Prompt Builder gives administrators an unprecedented set of tools to leverage large language models to determine sentiment with better accuracy than ever before. With LLM advancements and Prompt Builder, Salesforce continues leading the way in making enterprise AI conversational, useful, and intelligent.



We are now going to experiment with designing a prompt that will determine sentiment from a block of text.

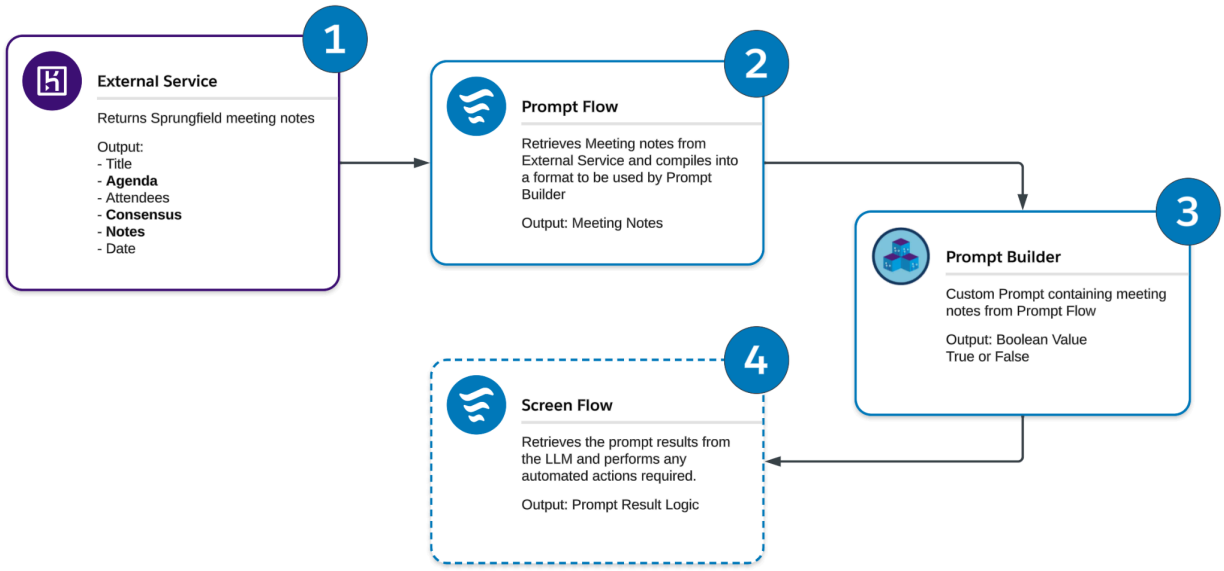
Overview

In this exercise, we will demonstrate how Salesforce's Prompt Builder can be used with an LLM to analyze text and detect negative statements about an individual.

Our example will utilize Sprungfield Committee meeting notes sourced from the web. We will design a prompt asking the AI to review these notes and identify if anything negative is stated about Montogomery Berns, our fictional owner of the Sprungfield Nuclear Power Plant.

The AI will scan the text, understand the context, and return either a True or False result depending on if it finds negative remarks about Mr. Berns.

This showcases how Prompt Builder gives us an easy way to leverage the linguistic skills and textual comprehension of an LLM. By translating our request into a natural language prompt, the AI can quickly parse the notes and alert us if they contain any concerning negativity about Mr. Berns.



In this exercise, we are going to create a text string within a Prompt Flow (2) by retrieving the meeting notes from the External Service (1). This text is going to be added to an existing prompt within Prompt Builder (3). An optional task is to create an Autolaunched Flow (4) that tests the results of the Prompt Builder (3).

Perform the following steps:

Assumptions:

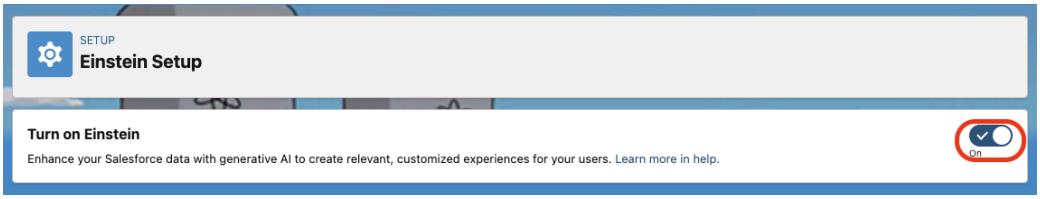
- ☑ The **GetMeetingMinutes** External Service is configured within your Salesforce org

Step 1: Turn on Einstein

Before you can use Einstein **Prompt Builder** in Salesforce, make sure Einstein is activated for your organization. Prompt Builder relies on AI capabilities that are only enabled once Einstein is turned on. Use the following steps to activate Einstein for your organization.

Note: This may already be done by finishing previous exercises.

- ☐ Click the **Setup** icon and select **Setup**.
  This will take you to the Setup page for your org
- ☐ In the **Quick Find**, type **Einstein Setup** then select **Einstein Setup**.
- ☐ Toggle the **Turn on Einstein** switch.

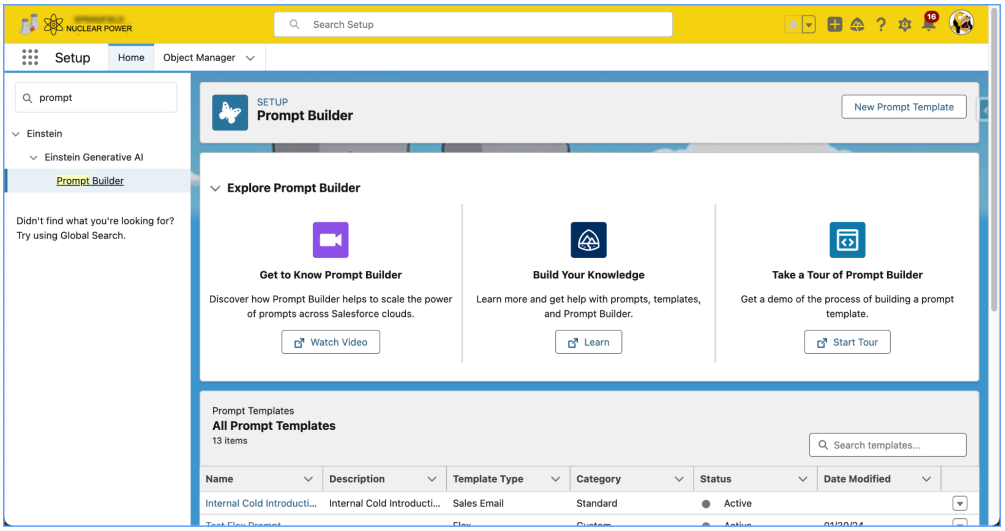


Step 2: Create the initial Prompt

The first thing we need to do is create the prompt in Prompt Builder. This is a requirement for a Prompt Flow to properly be linked.

Important: Without knowing the prompt, the flow won't be able to properly communicate.

- ☐ Click the **Setup** icon and select **Setup**
  This will take you to the Setup page for your org
- ☐ Enter **Prompt** in the **Quick Find**
  This will filter the list of options to show the Prompt Builder option
- ☐ Select **Prompt Builder** under the **Einstein Generative AI** group
  Salesforce displays the Prompt Builder list including options to explore this feature



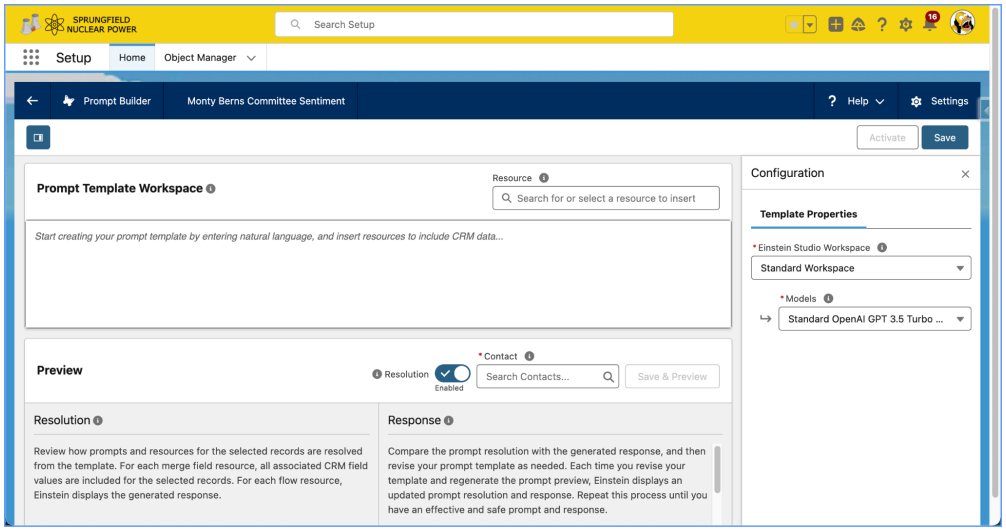
- ☐ Click the New Prompt Template button to create a new **Prompt Template**
  *Salesforce opens the **New Prompt Template** dialog*
- ☐ Enter the following:

Field	Value	Reason
Prompt Template Type	Flex	This prompt is going to use a Flow as a data source.
Prompt Template Name	Monty Berns Committee Sentiment Prompt	Required field
API Name		Automatically generated.
Template Description	Prompt to determine committee sentiment for Montgomery Berns	Descriptions help.

Define Resources		
Name	Contact	This is required and Contact is a logical choice
API Name		Automatically generated
Object	Contact	We said it was going to be Contact

☐ Click the Next button

Salesforce automatically displays the Prompt Builder interface

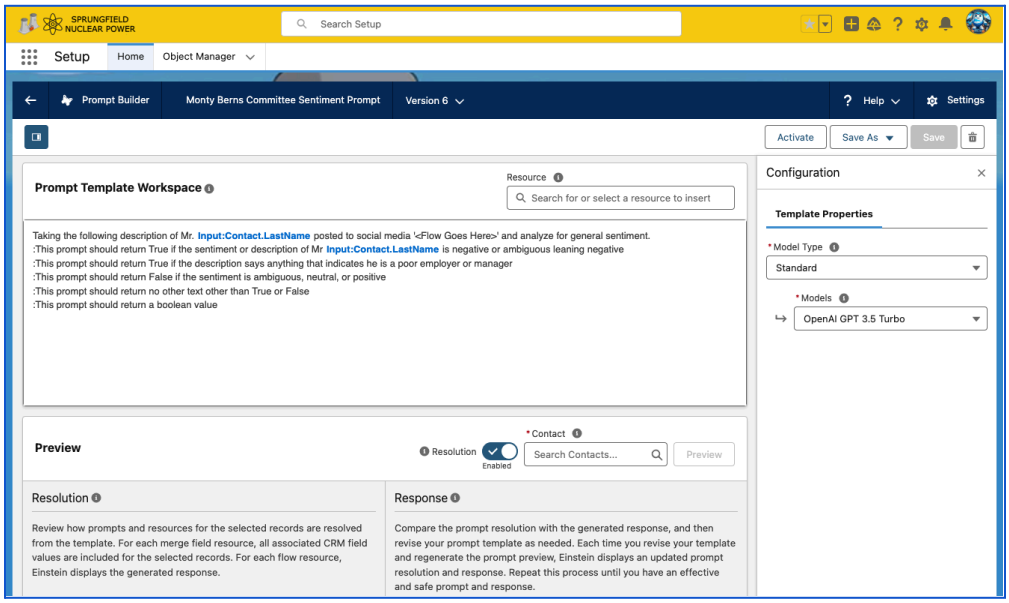


☐ Enter the following prompt in the **Prompt Editor**

```
Unset
Taking the following description of Mr Berns posted to social media '<Flow Goes Here>' and analyze for general sentiment.
:This prompt should return True if the sentiment or description of Mr Berns is negative or ambiguous leaning negative
:This prompt should return True if the description says anything that indicates he is a poor employer or manager
:This prompt should return False if the sentiment is ambiguous, neutral, or positive
:This prompt should return no other text other than True or False
:This prompt should return a  boolean value
```

We don't like having hard-coded names in the prompt. We selected the Contact object for an input for a reason. We want to be able to test for anyone the town committee is concerned about. In our case, we are looking for the name Berns to replace with the selected contact.

- [ ] Double-click the first location where you find the name **Berns** in the prompt.
  This is just so we know exactly where our cursor is.
- [ ] Click the **Resource** textbox at the top of the editor
- [ ] Select **Contact > Last Name**
  Prompt Builder will add the link to the prompt
- [ ] Delete the text **Berns** after the new entry
- [ ] Repeat these steps for the second instance of **Berns** found in the second line of the prompt.



- [ ] Click the Save button
  Salesforce saves the current prompt setup and deactivates the Save button
- [ ] Click the Activate button to activate the new prompt
  Salesforce activates the prompt and makes it available for use.

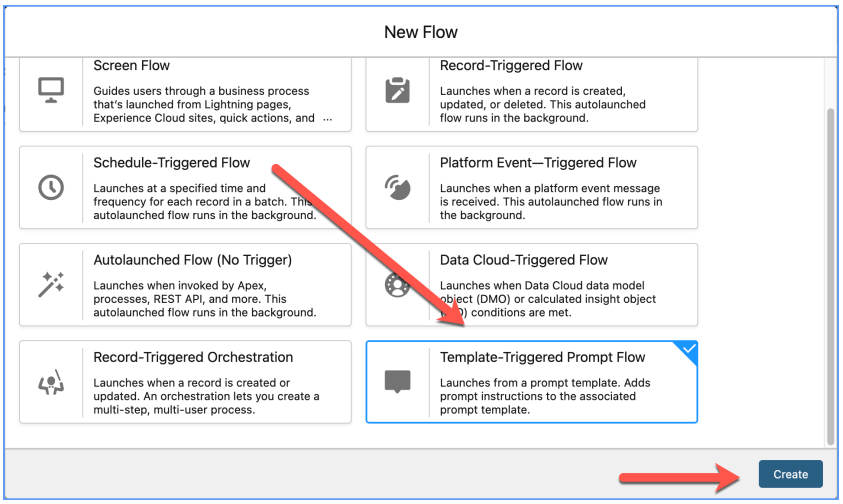
Note: Even though the new prompt is activated, it is not 100% complete. We could attempt to test it, but without the meeting notes, the LLM will be confused and return an invalid response.

Step 3: Create the Flow

Now that the basic prompt is activated, it becomes a known commodity for the Prompt Flow. We are now going to create this flow to pull the Committee Notes that will then be used by the Prompt.

Complete the following steps:

- ☐ Click the **Setup** icon and select **Setup**
  This will take you to the Setup page for your org
- ☐ Enter **Flows** in the **Quick Find**
  *This will filter the list of options to show the Flow option under the **Process Automation** group*
- ☐ Click the **Flows** link
  This opens the Flows list and displays all Flows currently created
- ☐ Click the New Flow button on the top-right corner of the interface
  *Salesforce opens a new tab and the **New Flow** dialog*
- ☐ Select the **Template-Triggered Prompt Flow** option

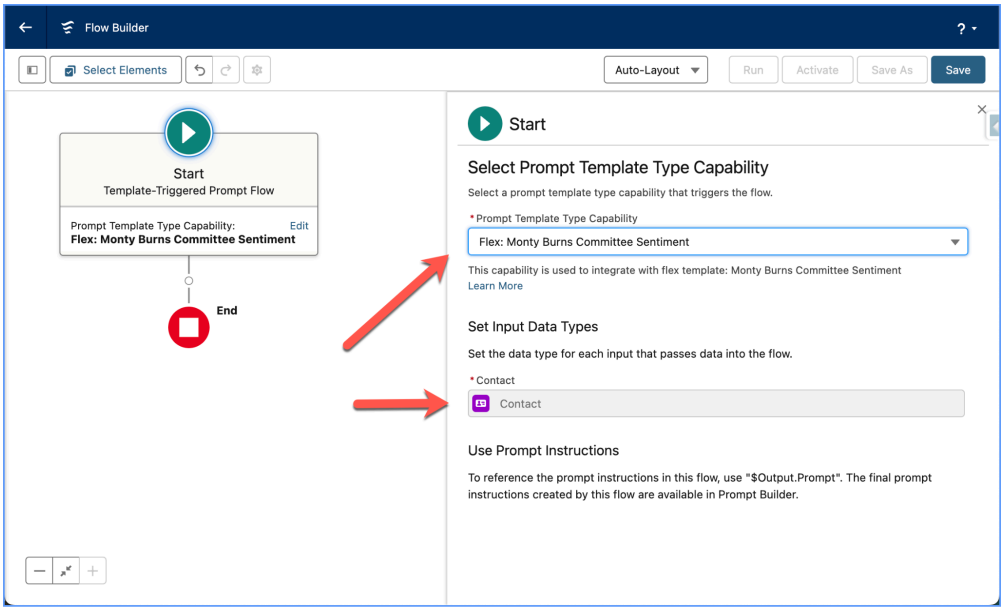


- ☐ Click the Create button
  Salesforce opens the Flow Builder for the new Template-Triggered Prompt Flow

Tip: The first thing a Template-Triggered Prompt Flow requires is the Prompt it will be communicating with.

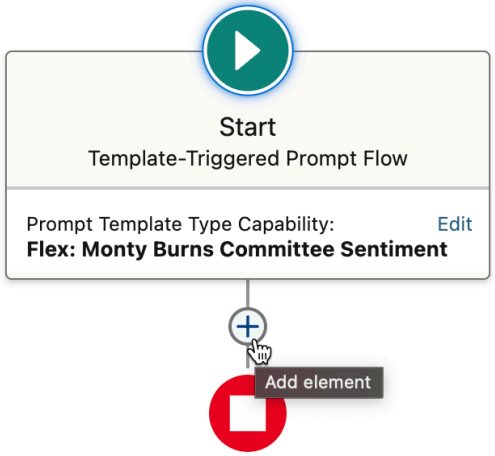
- ☐ Select the **Flex: Monty Berns Committee Sentiment Prompt** from the **Prompt Template Type Capability** picklist
  Flex identifies the type of Prompt. The rest identifies the name we gave the prompt.

Once selected, Salesforce automatically identifies the input(s) we identified for the Prompt. In this case it is the Contact.



☐ Click the **Add element** dot below the Start brick to add a new element

The dot will automatically change to a **+**



☐ Select the **Action** element

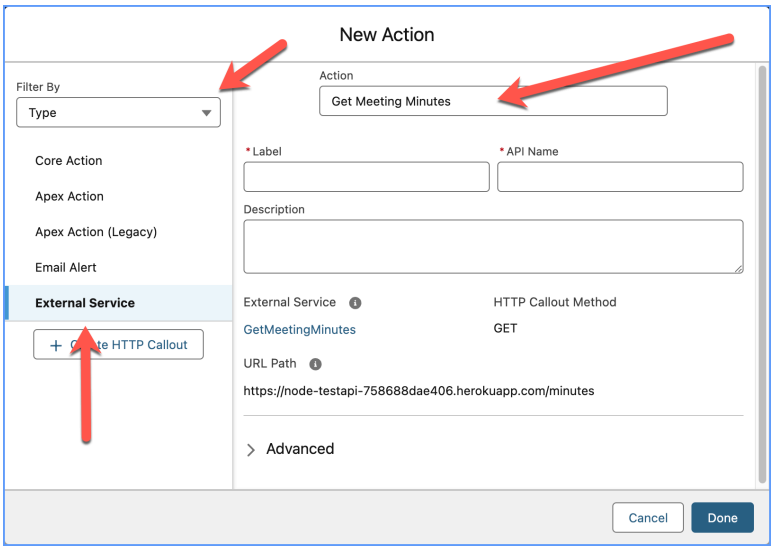
Salesforce opens the ***New Action*** *dialog*

☐ Select **Type** in the **Filter By** picklist

The list of options changes

☐ Select **External Service**

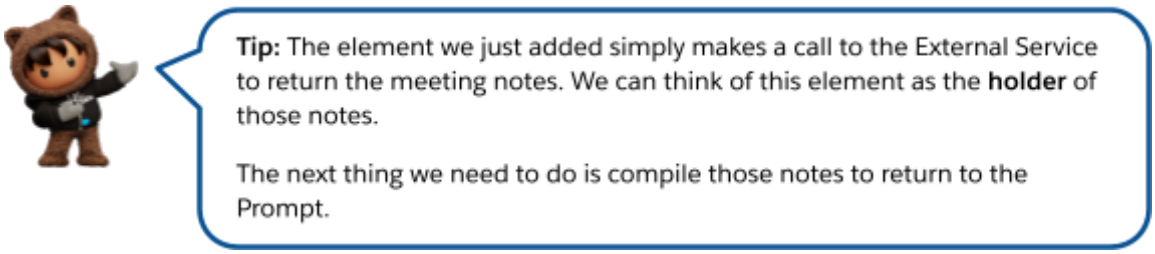
☐ Click in the **Action** textbox and select **Get Meeting Minutes** from the list
Salesforce changes the dialog to allow you to enter details about the External Service.



☐ Enter the following:

Field	Value	Reason
Label	Get Meeting Minutes	Required field
API Name		Automatically generated.
Description	Retrieve the newest Sprungfield committee meeting minutes.	Descriptions help.

☐ Click the Done button
The new Action element is added to the flow

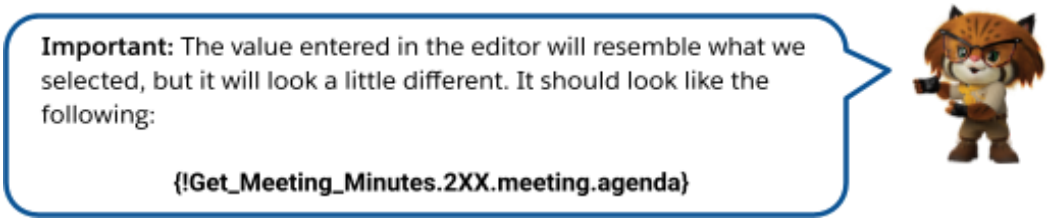


☐ Click the **Add element** dot below the new **Get Meeting Minutes** element
☐ Select the **Add Prompt Instructions** element from the list
*Salesforce adds the new element and opens the **New Add Prompt Instructions** panel directly to the right.*
☐ Enter the following:

Field	Value	Reason
Label	Return Meeting Minutes	Required field
API Name		Automatically generated.
Description	Build the prompt for the meeting minutes	Descriptions help.

- ☐ Click in the **Resource Picker** at the bottom of the panel
  *Salesforce opens the list of available options. In this case we will be focusing on the **Get Meeting Minutes** element.*
- ☐ Select the following:

  Outputs from Get_Meeting_Minutes > 2XX > meeting > agenda



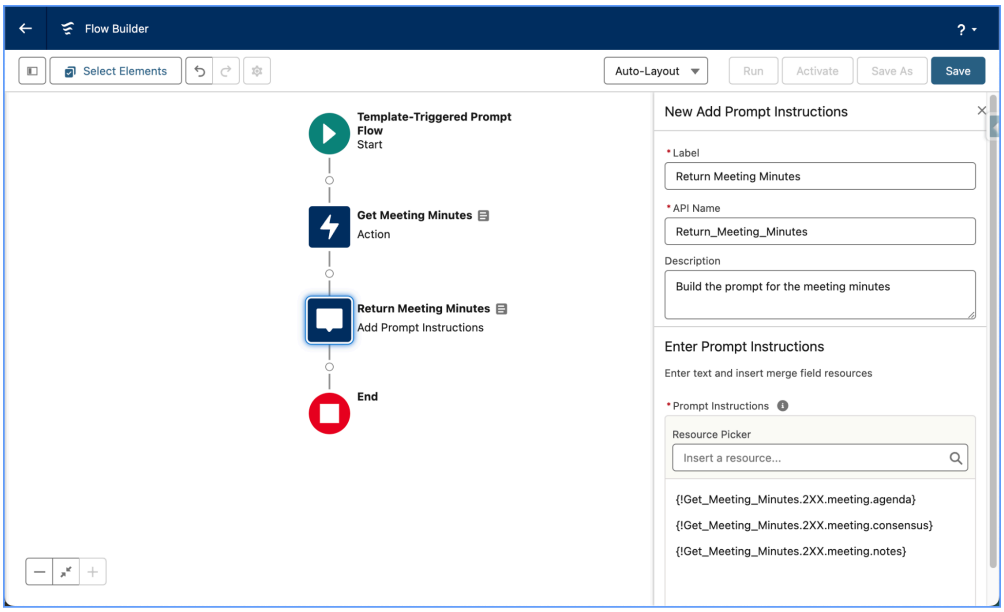
- ☐ Add a line *after* the value entered in the editor so the next thing added will be directly below the agenda.
- ☐ Click in the Resource Picker and select the following:

  Outputs from Get_Meeting_Minutes > 2XX > meeting > consensus

- ☐ Add a line *after* the value entered in the editor so the next thing added will be directly below the agenda.
- ☐ Click in the Resource Picker and select the following:

  Outputs from Get_Meeting_Minutes > 2XX > meeting > notes

- ☐ The screen should look like the following



☐ Click the Save button
*Salesforce opens the **Save the flow** dialog*

☐ Enter the following:

Field	Value	Reason
Flow Label	Get Committee Notes Flow	Required field
Flow API Name		Automatically generated.
Description	Flow to automatically pull and return the Agenda, Consensus, and Notes from the latest committee meeting.	Descriptions help.

☐ Click the Save button
*Salesforce opens the **Save the flow** dialog*

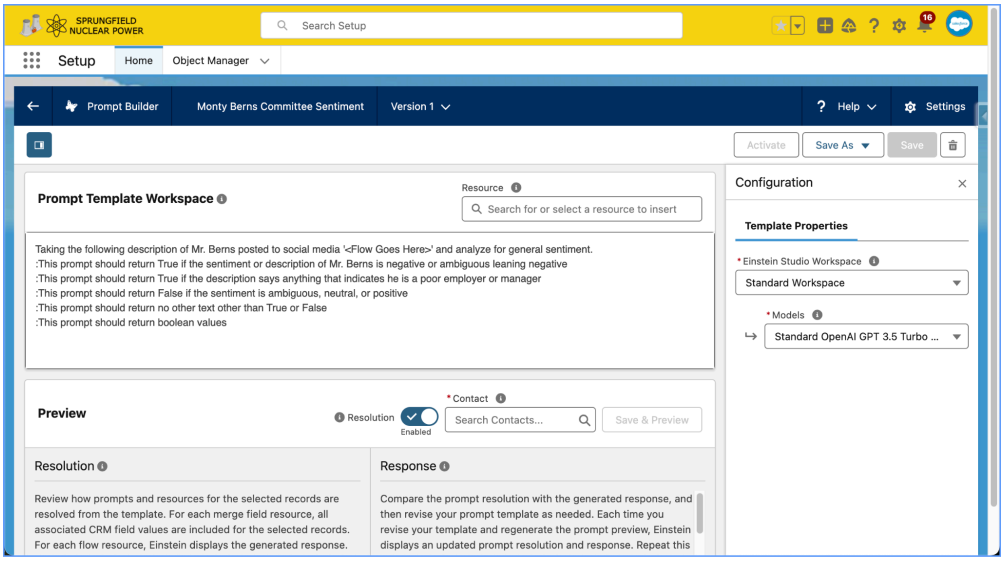
☐ Click the Activate button to activate the new flow
Salesforce activates the flow and makes it available for use within the prompt.

Notes: Now that the flow has been activated, Salesforce automatically added a Flows option to the available prompt inputs. Our final step is to update the prompt to use the results of the Flow within its **Grounding** process.

Step 4: Add the Flow to our Prompt

We have two of the three steps completed for our prompt creation. We started with creating the basic prompt. We then created a flow that automatically pulls meeting notes which we want to input into our prompt for it to test the overall sentiment.

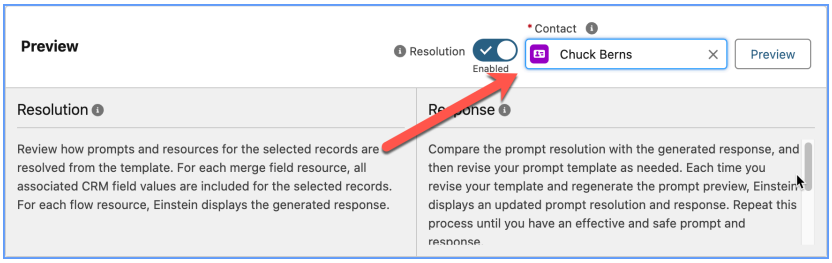
Our final step is to marry the two together. It's actually quite simple. Just complete the following steps:



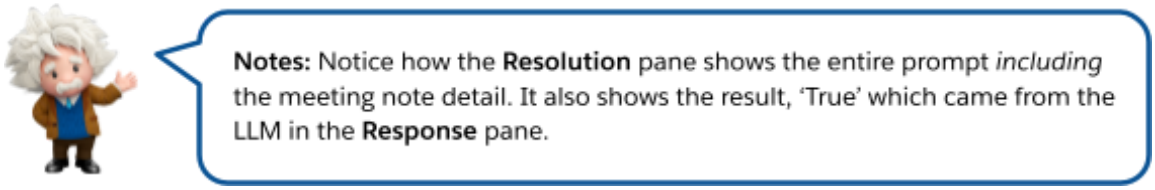
- ☐ Return to the browser tab with our prompt and **refresh your browser window**.
  or
  Open the **Monty Berns Committee Sentiment** prompt from the prompt list
- ☐ Place the cursor at the start of the **<Flow Goes Here>** text in the first line of the prompt.

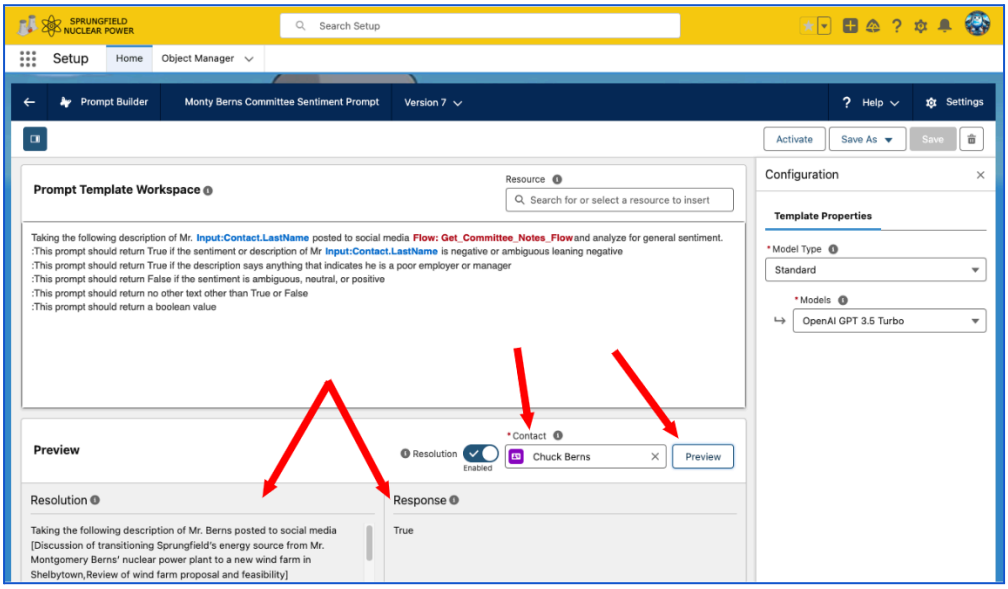
- ☐ Click in the **Resource** textbox
- ☐ Select **Flows** > **Get_Committee_Notes_Flow** from the list
  *Prompt Builder automatically adds the link to the flow in your template. The entry will be a bold, red entry, starting with **Flow:** followed by the name.*
- ☐ Delete the text <**Flow Goes Here**>
  The text was simply a visual cue. It's not needed for the prompt.
- ☐ Click the Save As button and select **Save as a New Version** from the list
  Prompt Builder saves the changes and creates version 2

- [ ] Click the **Contact** text box in the **Preview** section
- [ ] Search and select **Chuck Berns**
  *The **Preview** button will activate*



- [ ] Click the Preview button
  Prompt Builder automatically runs the Flow, pulling the meeting notes. Enters those notes into the prompt. Sends it to the LLM, and returns the result





Now that the prompt is working as expected, we can Activate the prompt so it's available throughout our org.

- [ ] Click the Activate button to activate the new version of our prompt.
  Prompt Builder changes the button and makes the prompt available to the entire org.

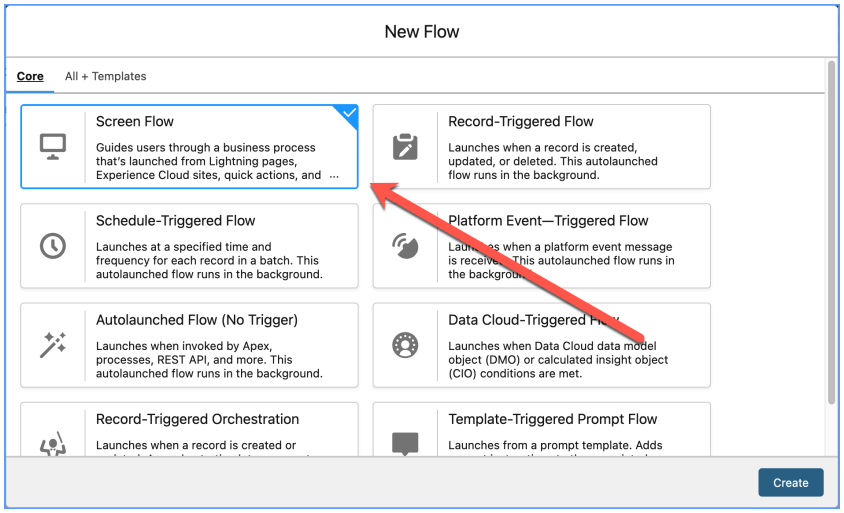
Step 5: Use the Results for Automation (optional)

The true power of Salesforce is being able to give administrators and users the ability to create automation without writing code. Now that we can determine the sentiment within notes, we can use that result and do something with it.

If the sentiment from the notes is deemed to be negative, Mr. Berns wants to be notified so he can launch an advertising campaign to boost his image. Our automation will not actually create the campaign, but we can easily notify him.

To make this process more interactive, we're going to create a Screen Flow so Mr. Berns can check his public image anytime he wishes.

- ☐ Click the **Setup** icon and select **Setup**
  This will take you to the Setup page for your org
- ☐ Enter **Flows** in the **Quick Find**
  This will filter the list of options to show the Flows option
- ☐ Click the **Flows** link
  This opens the Flows list and displays all Flows currently created
- ☐ Click the New Flow button on the top-right corner of the interface
  *Salesforce opens a new tab and the **New Flow** dialog*
- ☐ Make sure the **Screen Flow** option is selected.



- ☐ Click the Create button
  Salesforce opens the Flow Builder for the new Screen Flow

Tip: The **Screen Flow** is fun because it is interactive and can be placed anywhere on the Lightning interface.

- [ ] Click the **Add element** dot below the Start brick to add a new element

  The dot will automatically change to a **+**

- [ ] Select the **Screen** element

  *Salesforce opens the **New Screen** dialog giving you access to all of the different Lightning components available for use. It also prompts you to enter a label for your screen.*

- [ ] Enter the following:

Field	Value	Reason
Label	Get Town Sentiment	Required field
API Name		Automatically generated.
Description	Initial screen for testing town sentiment.	Descriptions help.

- [ ] Click the **Search Components...** textbox
- [ ] Enter **display**

  Flow Builder filters the list of components to only those with display in the name. It should consist of the Display Text and Display Image components

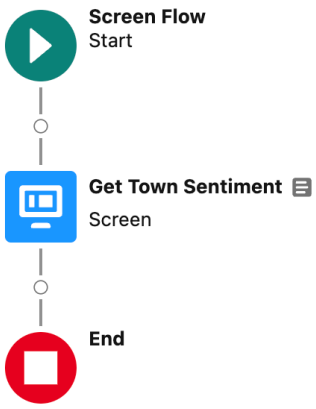
- [ ] Click the **Display Text** component

  Flow Builder automatically places a copy of the component on the screen

- [ ] Enter the following:

Field	Value	Reason
API Name	lblInstructions	We need to give this component a name. In this case, we prefix the name with 'lbl' to let us know that this is a label and isn't editable.
Resource Picker edit area	The current town sentiment towards you is '**Unknown**'. Click the **Next** button to update.	This is what the user will see. You can change the font as well as make text different colors or formatting.

☐ Click the Done button
Flow Builder closes the dialog and adds the screen to the flow



☐ Click the Save button on the top-right corner of the interface
*The **Save the flow** dialog is shown*

☐ Enter the following:

Field	Value	Reason
Flow Label	Screen Flow for Town Sentiment Flow	Required field
Flow API Name		Automatically generated.
Description	Automated screen flow to display the results from the automated prompt.	Descriptions help.

☐ Click the Save button

Tip: It is important to Save early and often

- ☐ Click the **Add element** dot below the new **Get Town Sentiment** screen element
- ☐ Select the **Get Records** element
- ☐ Enter the following:

Field	Value	Reason
Label	Get Mr. Berns	Required field
API Name		Automatically generated.
Description	Retrieve Mr. Berns record	Descriptions help.
Object	Contact	Data is found in the Contact object
Filter Contact Records		
Condition Requirements	All Conditions Are Met (AND)	Leave it to the default condition.
Field	LastName	This is the easiest
Operator	Equals	
Value	Berns	
Sort Contact Records		
Sort Order	Not Sorted	
Ancillary Settings		
How Many Records to Store	Only the first record	
How to Store Record Data	Automatically store all fields	

- ☐ Click the **Add element** dot below the new **Get Mr. Berns** Get Records element
- ☐ Select the **Action** element

*Flow Builder opens the **New Action** dialog*

Information: The **Action** element is the most versatile element in our arsenal. It gives us access to *almost* every facet of our Salesforce org.

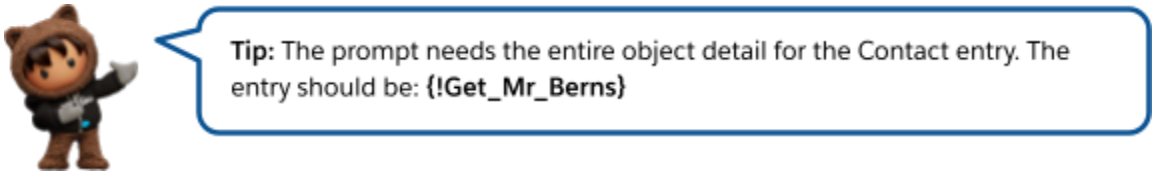
In this case, we are going to use it to access the **Monty Berns Committee Sentiment** prompt.

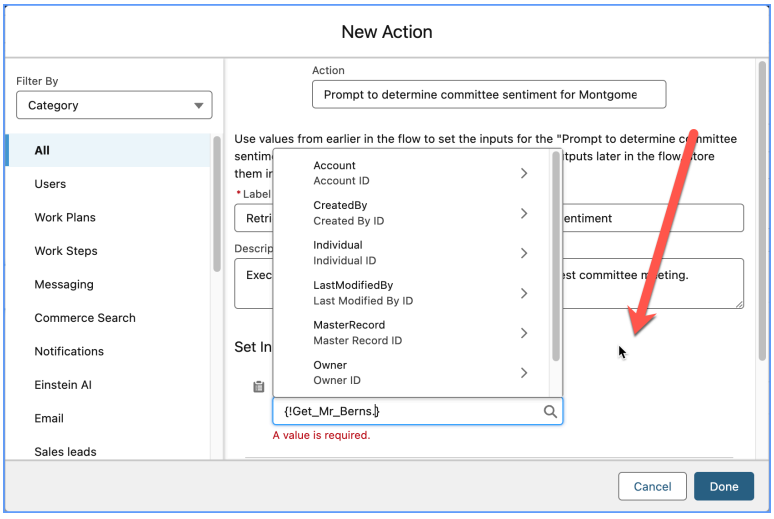
- ☐ Click in the **Search all actions** text box

- ☐ Enter **sentiment** in the text box
  Flow Builder will filter everything in the org for anything containing the word sentiment.
- ☐ Select the option **Monty Berns Committee Sentiment Prompt**
  This is the description for the prompt we created in Step 1 and completed in Step 3.
- ☐ Enter the following:

Field	Value	Reason
Label	Retrieve Sentiment	Required field
API Name		Automatically generated.
Description	Execute the prompt to retrieve the sentiment from the latest committee meeting.	Descriptions help.

- ☐ Click the **Contact** text box
  Flow Builder opens the list of available options
- ☐ Select the **Contact from Get_Mr_Berns** in the RECORD (SINGLE) VARIABLES group
  Flow Builder enters the beginning and advances to the submenu for the object.
- ☐ Click **anywhere** outside of the flyout options
  This is a shortcut to have Flow Builder automatically enter only the API name for the element.





- ☐ Click the Done button
  The new Action element is added to the flow

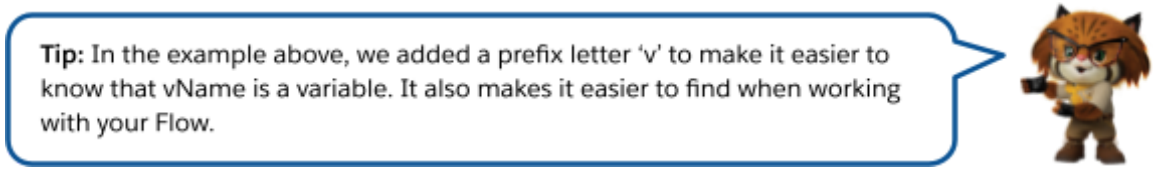
Working with Variables

We aren't finished with this **Action** element. The prompt we are calling will return a boolean value:

- **True** if the sentiment is negative
- **False** if the sentiment is positive

The element needs to be able to store the value in a holding area for us to use later in our flow. This holding area is called a **variable**.

A variable in programming is like a container that stores a value. You can think of it as a box that you put something in. They are given descriptive names to represent what value they contain. Like **vName** contains a name. What makes them handy is that the value stored in a variable can be changed.



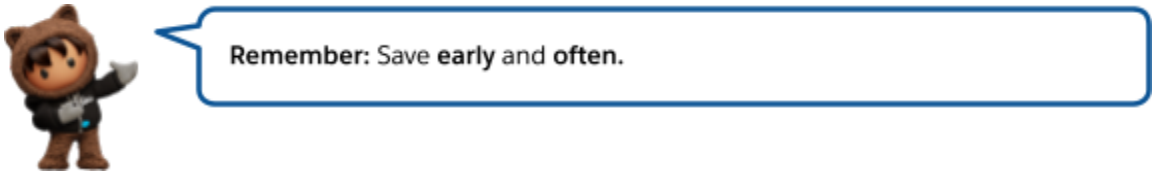
Fortunately, creating a variable is easy when working with Salesforce Flow Builder.

- ☐ Click the **Toggle Toolbox** button on the top-left corner of the interface
  Flow Builder opens the toolbox panel on the left side. This is where you can see everything we've created in this flow. It is also a handy area for us to create additional resources for our flow.
- ☐ Click the New Resource button in the Toolbox panel
  *Flow Builder opens the **New Resource** dialog*
- ☐ Select **Variable** in the **Resource Type**
  The required fields and options for a variable are displayed
- ☐ Enter the following:

Field	Value	Reason
API Name	vSentimentResult	Automatically generated.
Description	Variable to hold the sentiment result from the Prompt	Descriptions help.

Data Type	Text	

- ☐ Click the Done button
  The new variable is added to the flow
- ☐ Double-click the **Retrieve Sentiment** element in the flow
  Flow Builder opens the edit pane for the element
- ☐ Click the **Advanced** link at the bottom of the pane
  Flow Builder expands the list of available options
- ☐ Add a check to the **Manually assign variables** option
  *You will now see the **Prompt Response** text box*
- ☐ Click in the **Prompt Response** text box and select the **vSentimentResult** variable
- ☐ Click the **X** in the top-right corner of the pane to close it
- ☐ Click the Save button



Adding Logic to Our Flow

Our flow will display a screen informing Mr. Berns (the person who wants to use this flow). When he clicks the **Next** button, it automatically retrieves his record and executes the prompt we created earlier. The results of this prompt are stored in the variable vSentimentResult.

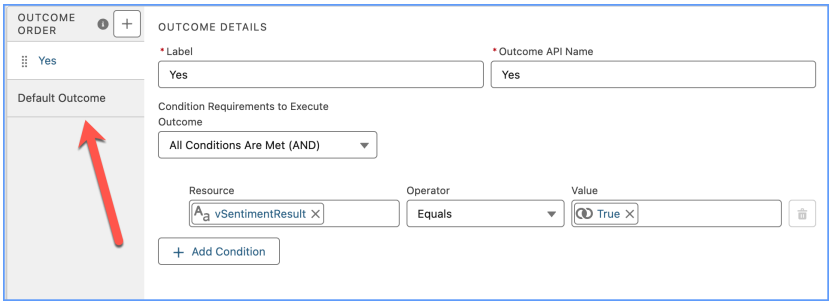
Our next step is to test the result and do something. To make it easier (and interactive), we're going to add a decision that tests the variable and updates the message that we'll show on a final screen. Once again, we're going to leverage variables.

- ☐ Click the **Add element** dot below the new **Retrieve Sentiment** action element
- ☐ Select the **Decision** element
  Flow Builder adds the element and creates a divided path in your flow

- [ ] Enter the following:

Field	Value	Reason
Label	Negative Sentiment?	Required field
API Name		Automatically generated.
Description	Was negative sentiment found in the meeting notes?	Descriptions help.
New Outcome Settings		
Label	Yes	This is the easiest
Outcome API Name		Automatically generated.
Resource	vSentimentResult	Select the variable holding the results of the automated prompt
Operator	Equals	
Value	{!$GlobalConstant.True}	Select the $GlobalConstant.True option from the list of available entries supplied

- [ ] Click the **Default Outcome** tab directly below the **Yes** outcome we just configured
*We want our Decision to make sense. The default outcome is anything that is found to be **not** True.*



- [ ] Change the label to **No**
Notice that when you click away, the label in the flow changes
- [ ] Click the **X** in the top-right corner of the pane to close it

This next step is where we are going to configure the message that will display on the final screen. We are going to use some developer magic to do this.

Our magic is going to consist of setting a Text variable that will hold the message. But, what makes this a little more magical is that we only need to configure it in one of the two Decision outcomes. The way we're going to do this is:

- Determine which path we want to use. In this case, we're going to reset the variable in the 'Yes' path.
- We are going to create a new variable but set the default value to a message where there were no negative comments made.

So, we'll have a variable in our flow that is preconfigured for a response where nothing negative is stated. If the prompt returns False, then the decision will follow that path and we won't have to do anything to the variable. If, however, the prompt returns True, we will follow that path and change the variable to a message stating that the committee has a negative outlook on the person.

In this part of the assignment, we're going to create the variable on the fly.

Let's get going:

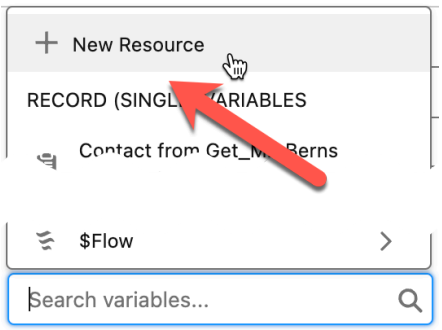
- ☐ Click the **Add element** dot below the **Yes** label under the **Decision** element
- ☐ Select the **Assignment** element
- ☐ Enter the following:

Field	Value	Reason
Label	Set Negative Message	Required field
API Name		Automatically generated.
Description	Update the variable to tell the user that the committee has a negative outlook on him/her.	Descriptions help.

- ☐ Click the **Variable** textbox in the **Set Variable Values** section
  Flow Builder opens the list of available options

Note: In this instance, we don't have a variable for the final screen yet. Flow Builder allows us to create new resources on the fly. That is what we're going to do now.

- ☐ Click the **+ New Resource** option at the top of the list



- [ ] Select **Variable** in the **Resource Type** dropdown
- [ ] Enter the following:

Field	Value	Reason
API Name	vScreenMessage	The lowercase 'v' reminds us that this is a variable
Description	Variable to hold the message shown on the last screen.	Descriptions help.
Data Type	Text	
Default Value	Nothing derogatory was found in the committee notes. You are liked.	

- [ ] Click the Done button

*The new variable is created and automatically entered in the **Variable** field for the Assignment element.*

- [ ] Click the **Value** textbox and enter the following:

> It appears that the committee does not appreciate everything you've done for the town of Sprungfield. It is time to start your 'Positive Outlook' campaign.

- [ ] Click the **X** in the top-right corner of the pane to close it

Display the Results

Our variable is set and we are ready to display the results to the user. All we need to do is add the final screen and post those results.

- [ ] Click the **Add element** dot above the **End** element
- [ ] Select the **Screen** element

- [ ] Enter the following:

Field	Value	Reason
Label	Town Sentiment Results	Required field
API Name		Automatically generated.
Description	Screen to display the prompt results	Descriptions help.

- [ ] Click the **Search Components** textbox
- [ ] Enter **display**
  Flow Builder filters the list of components to only those with 'display' in the name. It should consist of the Display Text and Display Image components
- [ ] Click the **Display Text** component
  Flow Builder automatically places a copy of the component on the screen
- [ ] Enter the following:

Field	Value	Reason
API Name	lblResults	We need to give this component a name. The 'lbl' reminds us that this is a label (something that cannot be edited by our users).
Resource Picker	{!vScreenMessage}	Click the Resource Picker and select the vScreenMessage variable.

- [ ] Add any formatting you like.
  Have fun!
- [ ] Click the Done button
  Flow Builder closes the dialog and adds the screen to the flow
- [ ] Click the Save button on the top-right corner of the interface

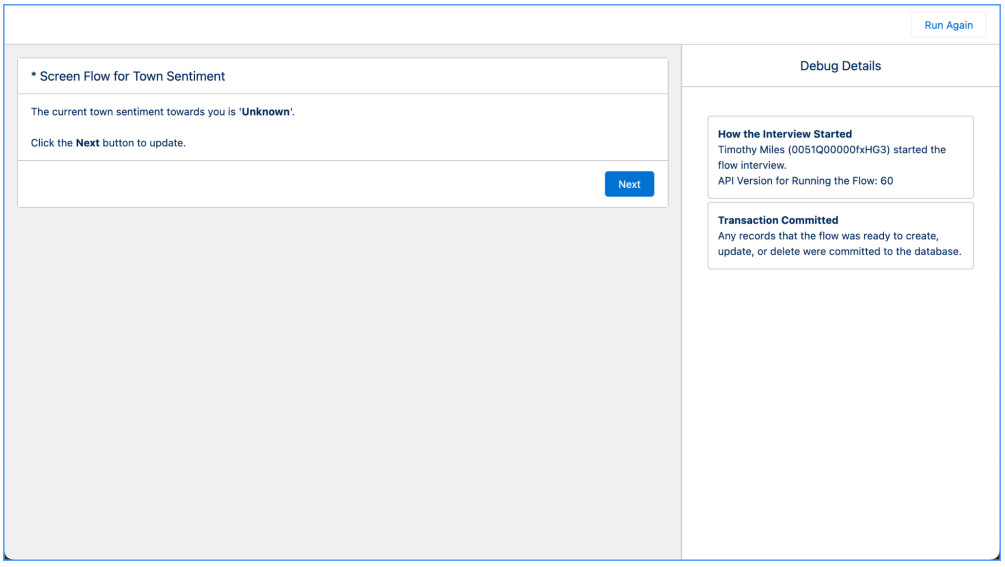
Test and Debug

We have our flow created and everything appears to be in order. We can use the Flow Debug tool to try it before it's seen by our users.

The Flow Debugger lets you step through and test flows built in Salesforce. You can use it to walk through each part of a flow, set breakpoints, and inspect the variables and logic. This helps you catch errors and issues before deploying flows. The Debugger gives visibility into your flows and makes it easy to identify and fix problems. It's an invaluable tool for flow builders.

Let's see it in action.

- ☐ Click the Debug button located at the top of the interface
  *Flow Builder opens a new tab along with the **Debug flow** dialog giving us options for how to proceed.*
  We will use the defaults at this time.
- ☐ Click the Run button to proceed
  The Flow Debugger displays the initial screen along with the Debug Details on the right side of the interface



- ☐ Click the Next button
  The flow executes as designed and displays the final screen with the result

Tip: Pay attention to the **Debug Details**. It tells you the results from every element added to your flow.

- [ ] Close the browser tab when done.
  This should return you to the Flow Builder and our Flow

Add to the Interface

We have completed building the automation and validated that it works as intended. The final step is to connect it to the user interface. We originally created this as a Screen Flow specifically so it could be integrated into the UI.

Fortunately, there is a Lightning Web Component ready-made to embed Screen Flows. To finish the integration, we simply need to add this component to the desired screen and configure it to run the flow we built. This will allow the automation to be launched through the user interface, which was the ultimate goal when we started this project.

Let's do it.

Important: Before we can use any flow in our org, it must be **Activated**. This is a special flag managed by Salesforce telling the platform that the flow is ready for Prime Time.

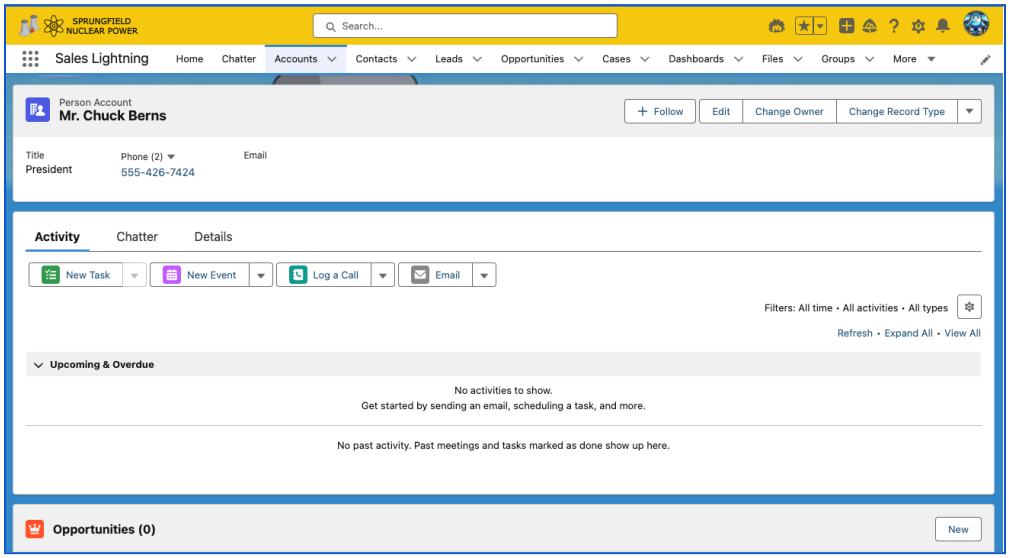
Assumptions:

- [x] We are currently in the Flow Builder interface working on the **Town Sentiment Screen Flow** Flow

Steps

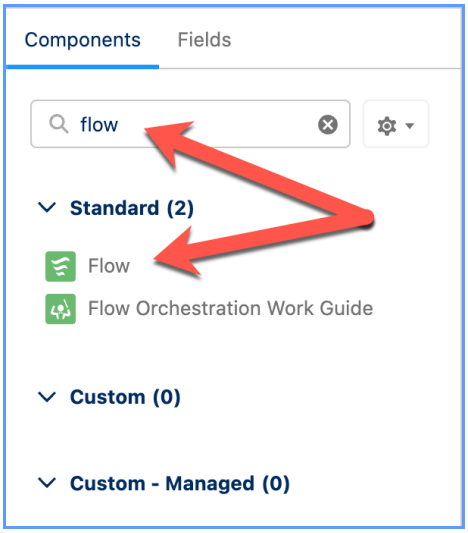
- [ ] Click the Activate button at the top of the interface
  *Salesforce runs in the background to activate the flow. Once completed, the **Save** button deactivates and the **Activate** button label is changed to **Deactivate**.*
- [ ] Return to any browser tab giving us access to the **App Launcher**
- [ ] Select **Sales Lightning** using the App Navigator 
- [ ] Search for **Berns** in the Search bar
- [ ] Select the **Chuck Berns Account**[1]
  This takes us to the Person account page for Berns

[1] Photos have been blurred to protect individual privacy and security

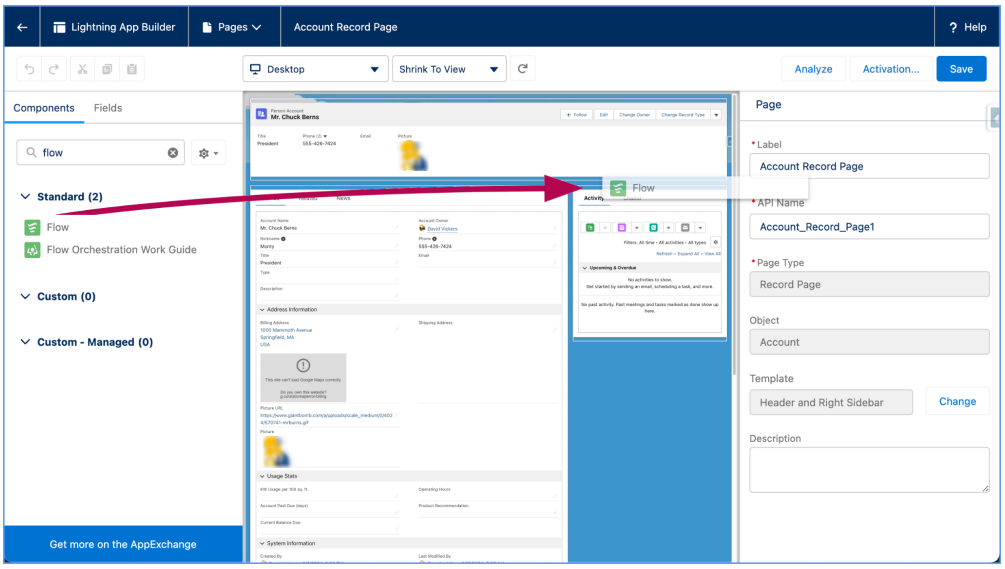


☐ Click the **Setup** ⚙ icon and select **Edit Page**
This will take you to the Lightning App Builder for the page

☐ Enter **Flow** in the **Search...** textbox in the **Components** pane
The list of available components is limited to those for Flows



☐ Click and drag the **Flow** component above the **Activity** and **Chatter** component on the right side of the interface.
The new component is added to the interface and simply states that it is a Flow Component. This is as designed.

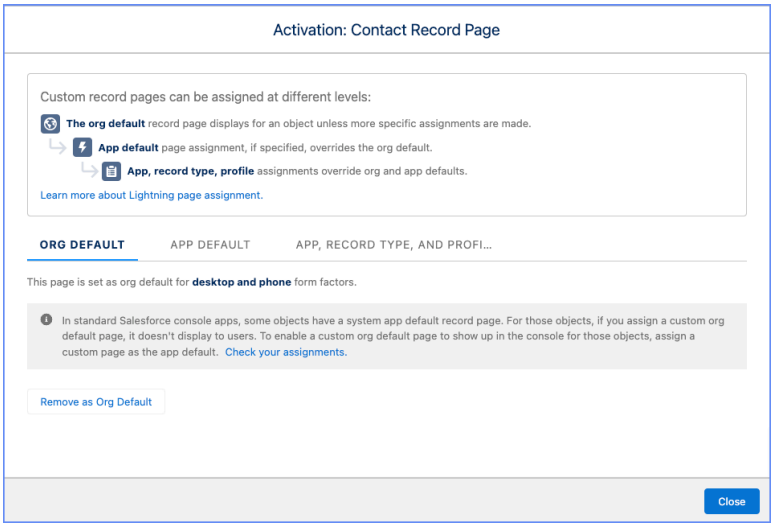


☐ Set the **Flow** field in the right pane to **Screen Flow for Town Sentiment Flow**
This tells the platform to render the flow we created in the interface.

☐ Click the Save button on the top-right corner of the interface

What if I Get the Activation Dialog?

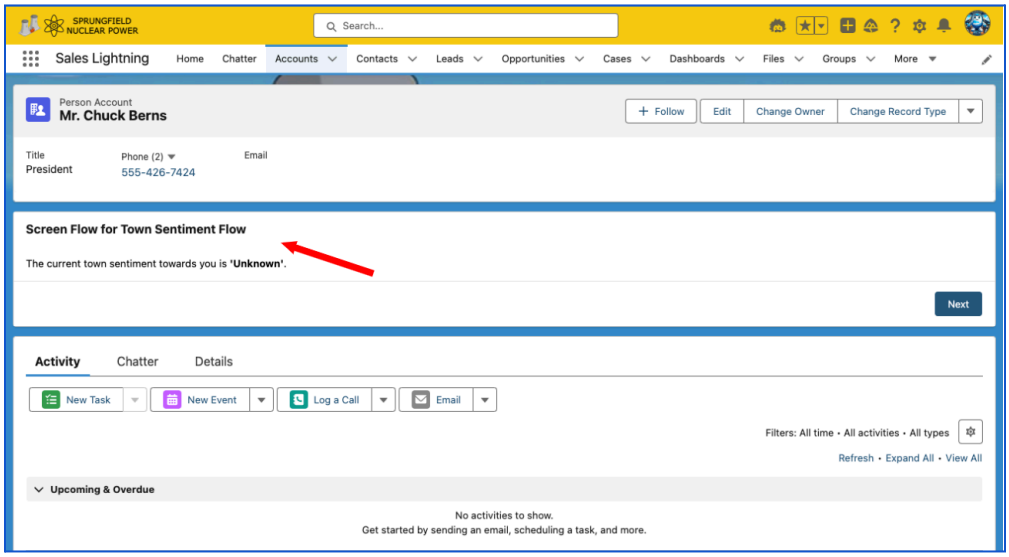
If this is the first time this page has been modified, you will receive the **Activation** dialog asking how this change should be viewed by others. For this workshop, we'll consider anything we've done should be used by **the entire organization**.



☐ Click the Close button to close the dialog.
You will be returned to the App Builder interface

☐ Click the Save button on the top-right corner of the interface again

☐ Click the Return ← button in the top-left corner of the interface
We will be returned to the interface with the new flow interface visible and ready for operation.



☐ Clicking the **Next** button will execute our flow, prompt, and external service.

Extra Credit: Expand the Flow

In the previous Step, we added the **Get Record** element to our flow, but we configured it to retrieve a single contact (Mr. Berns). This is obviously a very limited use case. It would be helpful to allow the flow to retrieve the current Person Account being viewed.

To do this, we must include additional logic.

- ☐ Create a new **Text Variable** vRecordID that is available for **Input**
- ☐ Add a new **Get Record** element *before* the **Get Town Sentiment** element
  - ☐ Name it: **Get Current Person**
  - ☐ Query the **Account** object where **Id = vRecordID**
- ☐ Edit the **Get Mr. Berns** Get Record element
  - ☐ Change the Label to: **Get Current Contact**
    The API Name can't be changed. That's OK.
  - ☐ Change the **Value** in the Filter logic to use the **Last Name** from the **Get Current Person** element
- ☐ Save the updated flow as a new Version
- ☐ Activate the new version
- ☐ Edit the Account Page
  - ☐ When you select the Flow component, there will be a new Input field property. Click the **Pass record ID into this variable** option
  - ☐ Save the changes
- ☐ Test with other accounts e.g. Homer Sampson

Want to learn more?

As the capabilities of Salesforce Data Cloud continue to expand, Salesforce is providing a growing set of enablement offerings to help customers maximize the value of the platform. A wide range of resources are now available to learn more about getting started with Data Cloud and utilizing its advanced features for analytics and data science.

Trailhead

There are many courses on **Trailhead** to help you get up to speed. Check out some Trailmixes here.

Documentation

Prompt Builder Information Page
See a demo and find additional information about Salesforce Prompt Builder.

Salesforce Ben
Interesting article about Salesforce Prompt Builder and a quick primer on operations

Salesforce Blog
Interesting blog post about Prompt Builder and how it is a game changer for administrators working with AI.

Where Do We Go From Here?

Data Cloud does the heavy lifting to unify data from different sources into complete customer profiles. With unified data established, we can then leverage automation tools like Data Action to easily trigger workflows and processes based on changes to that unified data. The power of Data Cloud is bringing disparate data together, so we can then activate it using the same declarative tools we're accustomed to in Salesforce.

There are many courses on **Trailhead** to help you get up to speed.

Ready to take your data unification to the next level? Reach out to your Account Executive today to schedule a customized session with our Technical Architects. Together, we'll explore how Data Cloud can solve your specific data challenges and demonstrate the possibilities firsthand. Imagine what you could accomplish with a complete 360-degree customer view. Let's make it happen!