



AppSheet Automation

Preview Release - Primer

Version 1.3

Updated: March 2, 2021

Disclaimer

This primer describes the basic capabilities of the AppSheet Automation Preview release. The Preview features described are not production ready and are intended for use in test environments only. **We do not recommend using these Preview features in production or with production data sources.** The Preview release is subject to change at any time and is provided to you for evaluation purposes only, without any SLAs. For more information, see the [product launch stages](#).

Version History

Version 1.0	Initial version. Introduction, Contact Sync use case
Version 1.1	Added a new Sheets eventing use case - Conference leads appointments generator.
Version 1.2	Added new concepts and use cases for Employee onboarding and Document processing.
Version 1.3	Added schema image for configuring Employee table, fixed some broken links

[Version History](#)

[Introducing AppSheet Automation](#)

[Key AppSheet concepts](#)

[Automation](#)

[Bot](#)

[Process](#)

[Step](#)

[Run a task](#)

[Branch on a condition](#)

[Wait for a condition](#)

[Call a process](#)

[Return values](#)

[Task](#)

[Event](#)

[Document Type](#)

[Fixed Schema Table](#)

[Folder Data Source](#)

[Extraction Confidence](#)

[Before you begin](#)

[Employee Onboarding](#)

[Creating the bot](#)

[Configuring the event](#)

[Configuring the process](#)

[Testing your automation](#)

[Automation Monitoring](#)

[Contact Sync](#)

[Creating the bot](#)

[Configuring the event](#)

[Configuring the process](#)

[Testing your automation](#)

[Conference leads appointment generator](#)

[Configuring the Appointment Confirmation Process](#)

[Creating the bot](#)

[Configuring the event](#)

[Configuring the main process](#)

[Testing your automation](#)

[Invoice Extraction Automation](#)

[Creating Invoice Table from Folder Data Source](#)

[File Processing](#)

[Adding Application Views](#)

[Low Confidence Extraction Handling](#)

[Failure Handling](#)

[Folder Contents as a Table](#)

[Creating an App](#)

[Creating a Table from Folder Data Source](#)

[Creating a useful view](#)

[Creating Slices to filter data](#)

[Linking Files to other Data](#)

[Appendix](#)

[Configuring Employees entity from Google Sheets datasource](#)

[Configuring the AppSheet Events add-on for Google Sheets](#)

[Configuring Company Contacts entity from Google Sheets datasource](#)

[Configuring Contact entity from Salesforce datasource](#)

[Configuring Leads entity from Google Sheets datasource](#)

[Configuring appointments entity from Google Sheets datasource](#)

[Configuring an email template for ops notification](#)

[Document processing states and status codes](#)

[Potential status codes and values](#)

[Why is there a separate currency code column?](#)

[Where can I store my files for Document processing?](#)

[What does 'collection of files' mean?](#)

Introducing AppSheet Automation

AppSheet is an intent-aware, no-code application development platform. AppSheet empowers businesses to create applications faster and with less money than a traditional, code-based approach. With AppSheet's true no-code platform, a business user does not need to know how to code in order to create a piece of software. The platform handles the technical heavy-lifting, allowing the business user to focus on the creation of effective apps.

This preview release introduces AppSheet Automation, a robust extension of AppSheet's platform offering new, integrated workflow automation capabilities.

Appsheet Automation:

- Unifies AppSheet's intent-aware, no code-platform with process/workflow automation as a first class feature.
- Offers Intelligent Process Authoring & Runtime with rich connectivity, allowing users to author and execute their business processes using entities.
- Introduces new concepts (including Bots and Process) to enrich the core AppSheet platform.
- Addresses the long tail of human centric processes, document based workflows and application integration use cases.

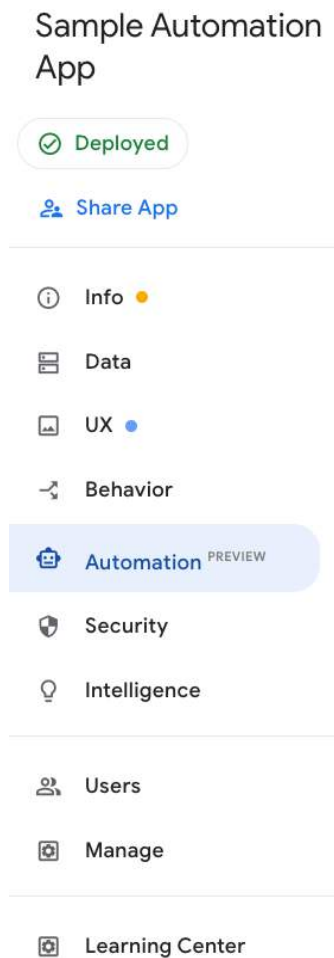
Key AppSheet concepts

This section describes some of the basic concepts introduced with the preview release of AppSheet Automation.

Automation

AppSheet *Automation* enables you to automate common business processes and document based workflows. You can access the new automation features from the new **Automation^{PREVIEW}** tab, as shown in the navigation image below.

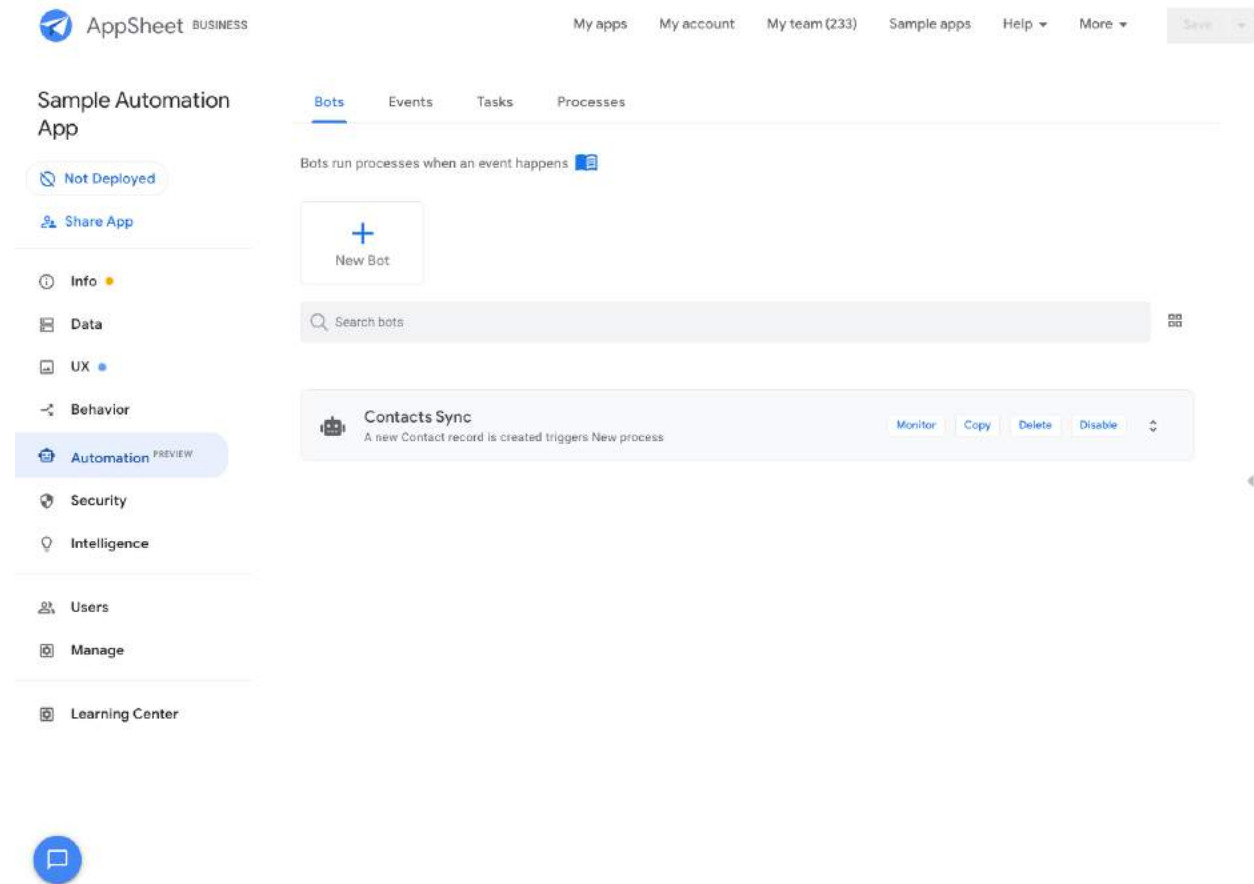
The platform encourages reusability, allowing you to build components once and use them in various automations, saving time and accelerating development. Events, processes and tasks are completely re-usable. That means you can reuse tasks within new processes, reuse processes within new bots, reuse events within new bots etc.



Bot

Bots are used to create automations by combining events with processes. Bots can be configured to trigger a process on detection of a specified event or according to a predetermined schedule. Once enabled, bots run in the background to listen for event triggers or trigger processes on a schedule. To terminate the process automation, you can disable the bot.

In the example shown below, the bot is configured to trigger the “Sync Contacts” process anytime a new “Contact” record is created in Salesforce. This bot continues to listen for the order creation event and trigger the associated process when it event occurs until it is disabled.



The screenshot displays the AppSheet Automation interface. On the left, a sidebar lists various app management options: Info, Data, UX, Behavior, Automation (highlighted with a 'PREVIEW' tag), Security, Intelligence, Users, Manage, and Learning Center. The main area shows a process named 'Contacts Sync' under the 'Bots' tab. The process description is 'A new Contact record is created triggers Sync contacts'. It is configured with the following steps:

- WHEN THIS EVENT OCCURS:** A new Contact record is created (Contact)
- RUN "SYNC CONTACTS":**
 - Update contact list
 - END

At the bottom of the process editor, there is a '+ Add a step' button and a 'Documentation' link.

Process

A *process* represents a typical business process. For example, an “order approval” is a business process. A process contains the following elements:

- A sequence of steps
- An input
- An output (this is optional)
- An event that triggers the process

For example, in the screenshot below, you can see a sample “Sync Contacts” process. In this example, the *event* that triggers the process is the addition of a new contact record.

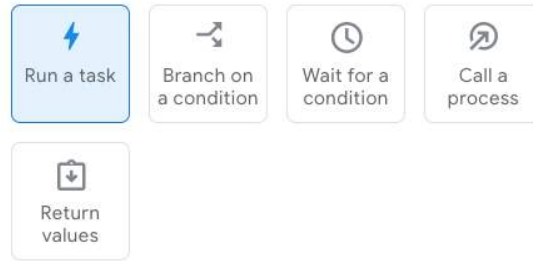
The screenshot displays the AppSheet Business interface. At the top, there are navigation links for 'My apps', 'My account', 'My team (233)', 'Sample apps', 'Help', and 'More'. A 'Save' button is visible in the top right corner. The main navigation menu on the left includes 'Info', 'Data', 'UX', 'Behavior', 'Automation' (highlighted with a 'PREVIEW' badge), 'Security', 'Intelligence', 'Users', 'Manage', and 'Learning Center'. The central area shows the 'Processes' tab selected, with a search bar for 'Search processes'. A process titled 'Sync contacts' is open, showing a single step 'Update contact list' with the entity type 'Contact'. Below the step, there is an 'END' label and an 'Add a step' button. A 'Documentation' link is visible at the bottom of the process editor.

Step

A *step* is a basic element in a process that defines a task to be carried. A process can have several steps.

Steps can be of several types:

- Run a task
- Branch on a condition
- Wait for a condition
- Call a process
- Return values



Run a task

This step type allows you to run one of the following task types - “Send an email”, “Send a notification”, “Send an SMS”, “Change data”, “Call a webhook” or “Create a new file”.

In our example, the *step* is “Update contact list.” The step runs a task (of type “Change data”) against the data provided to the process on the *input*. In this case, the input is the new contact record.

Step name

Update contact list

Do this

Run a task

Branch on a condition

Wait for a condition

Call a process

Return values

Task

What task do you want to run?

Update contact

Task category

Category of task

Send an email

Send a notification

Send an SMS

Change data

HTTP
Call a webhook

Create a new file

Branch on a condition

This step type allows you to implement conditional logic in your process; for example, `if <condition> - then <do some step> - else <do some other step>`. The condition value can be any valid expression.

Step name


Did contact reside outside the US?

Do this

Run a task Branch on a condition Wait for a condition Call a process

Return values

Condition to check

= [OtherCountry] != "USA" 

Wait for a condition

This step type allows the process to pause execution until the condition evaluates to true, at which point the process resumes execution.

Step name


Wait for supervisor approval

Do this

Run a task Branch on a condition Wait for a condition Call a process

Return values

Wait until this condition is true

= [Status] = "approved" 

Call a process

This step type allows you to invoke another process within the same application. You can pass data to the process being invoked using literal values or expressions.

Step name

Update contact list

Do this

Run a task

Branch on a condition

Wait for a condition

Call a process

Return values

Process name

Sync contact list

Process inputs

First Name

= [FirstName]

Last Name

= [LastName]

Email

= [Email]

Return values

This step type allows you to return specific values from within your process and access those values during later steps. For example, you may want to have a parent process invoke a sub-process, and use the *return value* of the sub-process to determine the next steps to be performed.

Step name

Return approval decision

Do this

Run a task Branch on a condition Wait for a condition Call a process

Return values

Return these values

ApprovalStatus = [Decision]

Add

In the example above, a parent process has a step called “Send for approval” that returns an approval decision using the *Return values* step type shown above. Once the return value is set, it is accessible within the parent process through the expression “[Send for approval].[ApprovalStatus].”

Task

As noted above, a step defines a *task* to be carried out in the process. There are six types of tasks:

- [Send an email](#)
- [Send a notification](#)
- [Send an SMS](#)
- [Change data](#)
- [Call a webhook](#)
- [Create a new file](#)

Our example shows configuration of a “Change data” task type along with the associated data action.

The screenshot displays the AppSheet Business interface. At the top, the AppSheet logo and 'BUSINESS' are on the left, and navigation links for 'My apps', 'My account', 'My team (233)', 'Sample apps', 'Help', and 'More' are on the right. A 'Save' button is also present. Below the navigation, the page title is 'Sample Automation App'. A sidebar on the left contains various app management options: 'Not Deployed', 'Share App', 'Info', 'Data', 'UX', 'Behavior', 'Automation' (highlighted with a 'PREVIEW' tag), 'Security', 'Intelligence', 'Users', 'Manage', and 'Learning Center'. The main content area is titled 'Tasks' and shows a 'New Task' button and a search bar. A task configuration card for 'Update contact' is visible, with the following details: 'Task category' is 'Change data'; 'Task name' is 'Update contact'; 'Table name' is 'Contact'; and 'Data Change Action Name' is 'Update company contact'. The 'Change data' task category is selected among other options like 'Send an email', 'Send a notification', 'Send an SMS', 'HTTP Call a webhook', and 'Create a new file'.

☰ Contact

Update company contact

effect: Data: add a new row to another table using values from this row

Copy Delete

For a record of this table
This action applies to rows of which table?

Contact

[View Definition](#)

Do this
The type of action to perform

Data: add a new row to another table using values from this row

Table to add to
Choose any table that is part of the app

CompanyContacts

[View Definition](#)

Set these columns
To the constant or expression values defined

⋮	First Name	=	[FirstName]	⚠	🗑
⋮	Last Name	=	[LastName]	⚠	🗑
⋮	Email	=	[Email]	⚠	🗑
⋮	Company	=	[AccountId]	⚠	🗑
⋮	Date Created	=	[CreatedDate]	⚠	🗑

Add

Appearance ▼

Event

An *event* represents a change to an entity (table). Each event has the following attributes:

- **Event type.** This attribute indicates the kind of change that triggers the event.
- **Condition (optional).** If specified, this condition is checked before firing the Action.
- **Target data.** Indicates which schema would trigger the event, if changed.
- **Update event.** Indicates the type of schema change that would trigger the event (i.e. additions, updates, deletions, etc.).

In our example, as shown in the image below, the New Event is a “Data Change” event only triggered by additions to the “Contact” data schema.

The screenshot displays the AppSheet Business interface for configuring an automation event. The top navigation bar includes 'My apps', 'My account', 'My team (215)', 'Sample apps', 'Support', and 'More'. The left sidebar contains various app management options, with 'Automation' (marked as ALPHA) selected. The main content area is titled 'Events' and shows a configuration for a new event: 'A new Contact record is created'. The event type is set to 'Data Change', the condition is '=', the target data is 'Contact', and the update event type is 'ADDS_ONLY'. A 'Documentation' link is visible at the bottom of the configuration panel.

Below is an example of a “Schedule” event type. In the case below, where **ForEachRowInTable** is not enabled, you can select any event except “Data change” as a valid step type in the process. You may want to use this option if your process doesn’t rely on data in any table.

Onboard new employees every Monday morning

Scheduled

Copy Delete

Event Type
What type of change triggers this event?

Data Change Schedule

ForEachRowInTable
Is this trigger for each row in the given table?

Schedule
What schedule will trigger this workflow rule?

Daily

Time 9:00 am

Time zone
What time zone should the schedule use?

UTC

Documentation

However, if the **ForEachRowInTable** option is enabled, the option to specify a filter condition appears. Use this option if you want to run a process for each row of data in the table.

Onboard new employees every Monday morning

Copy
Delete
⌵

Scheduled

Event Type

Data Change

Schedule

ForEachRowInTable

Is this trigger for each row in the given table?

Table

What table to query?

Contact
▾

[View Definition](#)

Filter Condition

Filter condition to specify which rows of the table?

=
[
Department
]
=
"HR"
⌵

Schedule

What schedule will trigger this workflow rule?

Daily
▾

Time

9:00 am

Time zone

What time zone should the schedule use?

UTC
▾

Documentation
▾

Document Type

Document types are available in document-based workflows. AppSheet automation classifies document types based on the document's content, regardless of mime type or extension type. For example, *document types* include: receipt, invoice, resume, work order, W9, etc. The *document type* does not refer to mime types (such as spreadsheet, presentation, or photo) or extensions (such as .pdf, .docx, .png). A document type can be represented in multiple formats depending on how it is received - exported as a PDF, scanned as a PDF, photographed as a .png, or otherwise. In any of those cases the *document type* remains "invoice" because of its content.

Fixed Schema Table

For document extraction to work properly, the table schemas for document extraction must match the content fields to be extracted. As a result, *fixed schema tables* created for the purpose of document extraction are immutable and can't be altered by application editors. A user may still edit the UX of the table views to suit their needs but must ensure that required fields are present for creation of entries into these tables.

Folder Data Source

For document-based workflows that utilize *document types*, a new data source type is also available to support folders as data sources. Google Drive is the first system supported, with the intention of adding more *folder data source* systems in the future.

Folders are fixed schema tables, where the attributes (table columns) are the properties of files (such as name, size, last modified at, etc.) and the entries (table rows) are the files themselves. In the AppSheet Automation Preview release, each folder should contain files of the same *document type*, although the files can vary in mime and extension type.

Note: *Supported extensions in the Preview Release: .tiff, .gif, and .pdf. Folders containing mixed document types can be used for document viewing and upload inside applications, but can not be automatically extracted.*

For the Preview Release, *folder data sources* can be either **implicitly** used as part of configuring document extraction, or **explicitly** used as data sources for building views to show documents in the application. For example, a "user manual view" in an application can explicitly use a Google Drive *folder data source* containing documents that can be viewed, updated, added, or deleted within an application.

Extraction Confidence

Some source documents may have defects that result in extraction accuracy that is less than 100%. For example, some documents may be based on scanned sources, utilize complex images, use unusual fonts, or have spelling errors or omissions. As a result, the extraction service includes an *extraction confidence* score that is then used by the automation process to determine if the extracted data should be manually reviewed. If the *extraction confidence* score falls below the configured threshold, an exception flow is triggered to involve manual review by a human.

Note: In the Preview Release, the confidence threshold cannot be adjusted by application editors.

Before you begin

Before testing the AppSheet Automation preview features, make sure you have completed the following steps:

- Created an account on appsheet.com.
- Ensure that you have access to the Automation preview release (Be able to see Automation ^{PREVIEW} logo in the left nav bar).
- Recommend that you create a new Sample Automation App that you specifically use for creating and testing your new automations.

Now you are ready to explore AppSheet Automation!

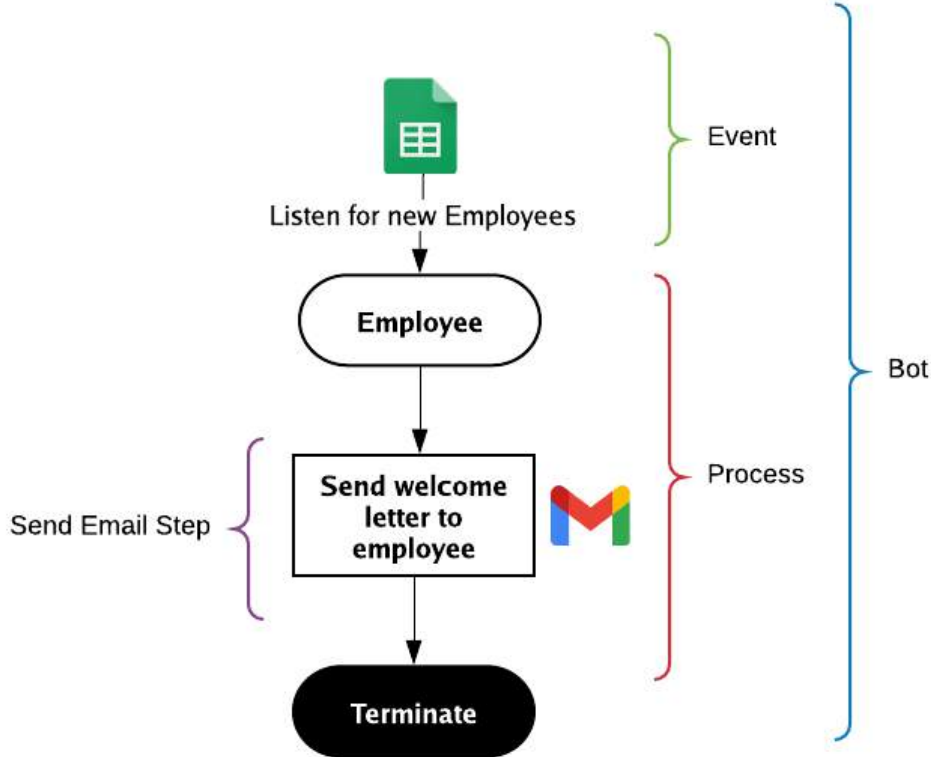
Employee Onboarding

This brief tutorial shows you how to set up an automated process to onboard new employees. When a new employee is added in a Google Sheet, AppSheet automatically emails a welcome letter to that employee.

Implementing Employee Onboarding with AppSheet Automation involves the following steps:

1. [Creating the Bot](#)
2. [Configuring the Event](#)
3. [Configuring the Process](#) (steps and actions)

The following figure describes the high level components:



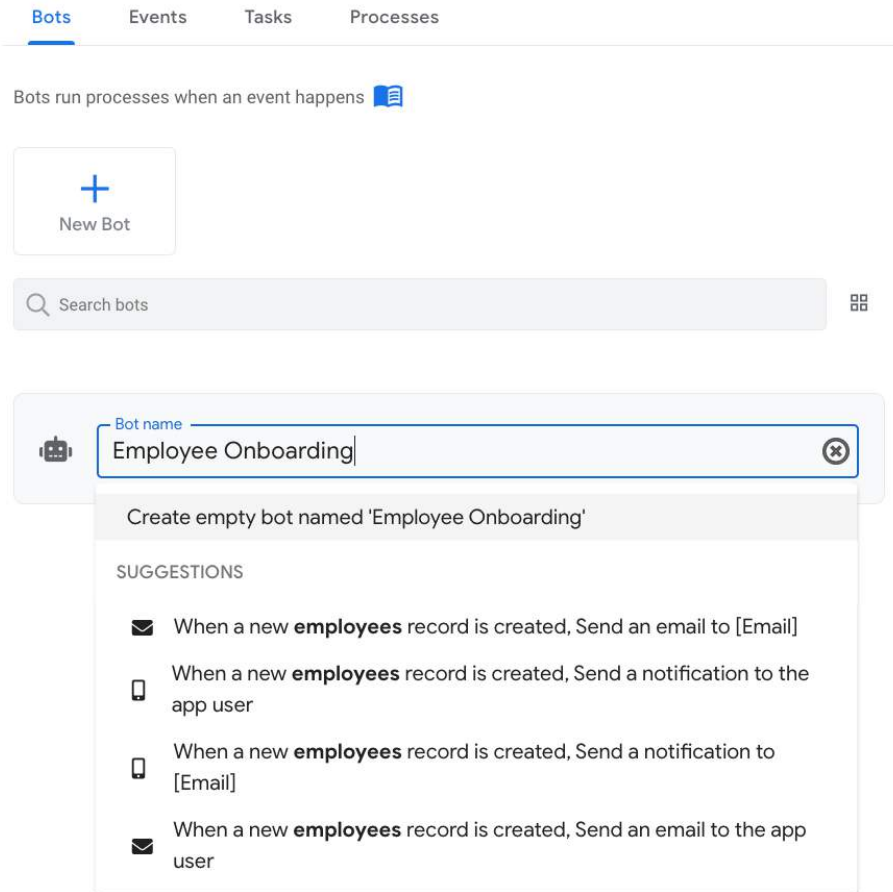
Before you begin this tutorial:

- Make sure you can log into the AppSheet Editor UI and access the Sample Automation App you created as a prerequisite in [Before you begin](#).
- Follow the steps in the Appendix to configure a [Employee Entity](#) from Google Sheets data source and install the AppSheet [Events Add-on](#).

Creating the bot

Follow the steps below to create a new bot:

1. From the AppSheet UI, navigate to the **Bots** tab and click **New Bot**.
2. Type `Employee Onboarding` into the dialog and suggested actions appear, as shown in the image below.
3. Select **Create empty bot named 'Employee Onboarding'** to create the bot.



Configuring the event

Once the bot is created, you can define the event for the bot. In this example, the event trigger is the addition of a new employee to Google Sheets.

To create an event:

1. Select the bot you just created in the previous section.
2. Click **Choose an event**.
3. Suggested events may appear. (Skip to step 5 if there aren't any suggested events)
4. Select **A new Employee record is created** to have the event automatically created for you. The platform does this for you and saves you valuable time.

The screenshot displays the AppSheet Automation interface. On the left is a navigation sidebar with categories like Info, Data, UX, Behavior, Automation (highlighted), Security, Intelligence, Users, Manage, and Learning Center. The main area shows the 'Bots' tab with a 'New Bot' button and a search bar. A bot named 'Employee Onboarding' is selected, showing its configuration. The bot is currently 'Incomplete Bot'. The configuration is divided into two sections: 'WHEN THIS EVENT OCCURS' with a 'Choose an event' button, and 'RUN THIS PROCESS' with an 'Add a step' button. Below these are sections for 'Appearance' and 'Documentation'.

Note: If you want to manually configure the event, you can navigate to the **Events** tab, select **Create new event**, and enter in the relevant details.

5. Enter **A new Employee record is created** as the event name and configure the event as follows:

AppSheet BUSINESS

My apps My account My team (234) Sample apps Help More Save

Sample Automation App

Not Deployed Share App

- Info
- Data
- UX
- Behavior
- Automation PREVIEW**
- Security
- Intelligence
- Users
- Manage
- Learning Center

Bots Events Tasks Processes

Bots run processes when an event happens

New Bot

Search bots

Employee Onboarding

Incomplete Bot

Monitor Copy Delete Disable

WHEN THIS EVENT OCCURS

Event name
A new Employee record is created

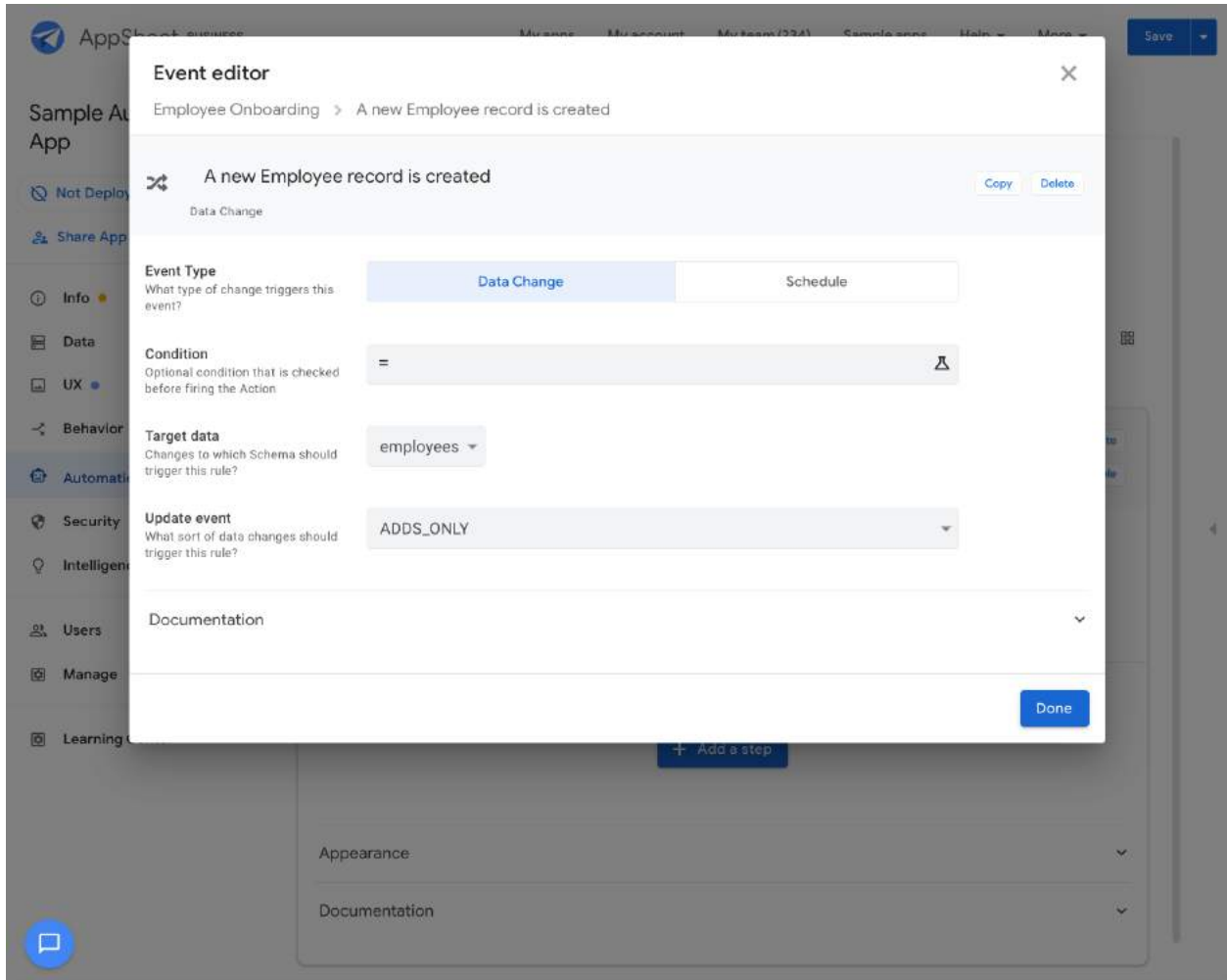
RUN THIS PROCESS

Create new event named 'A new Employee record is created'

+ Add a step

Appearance

Documentation



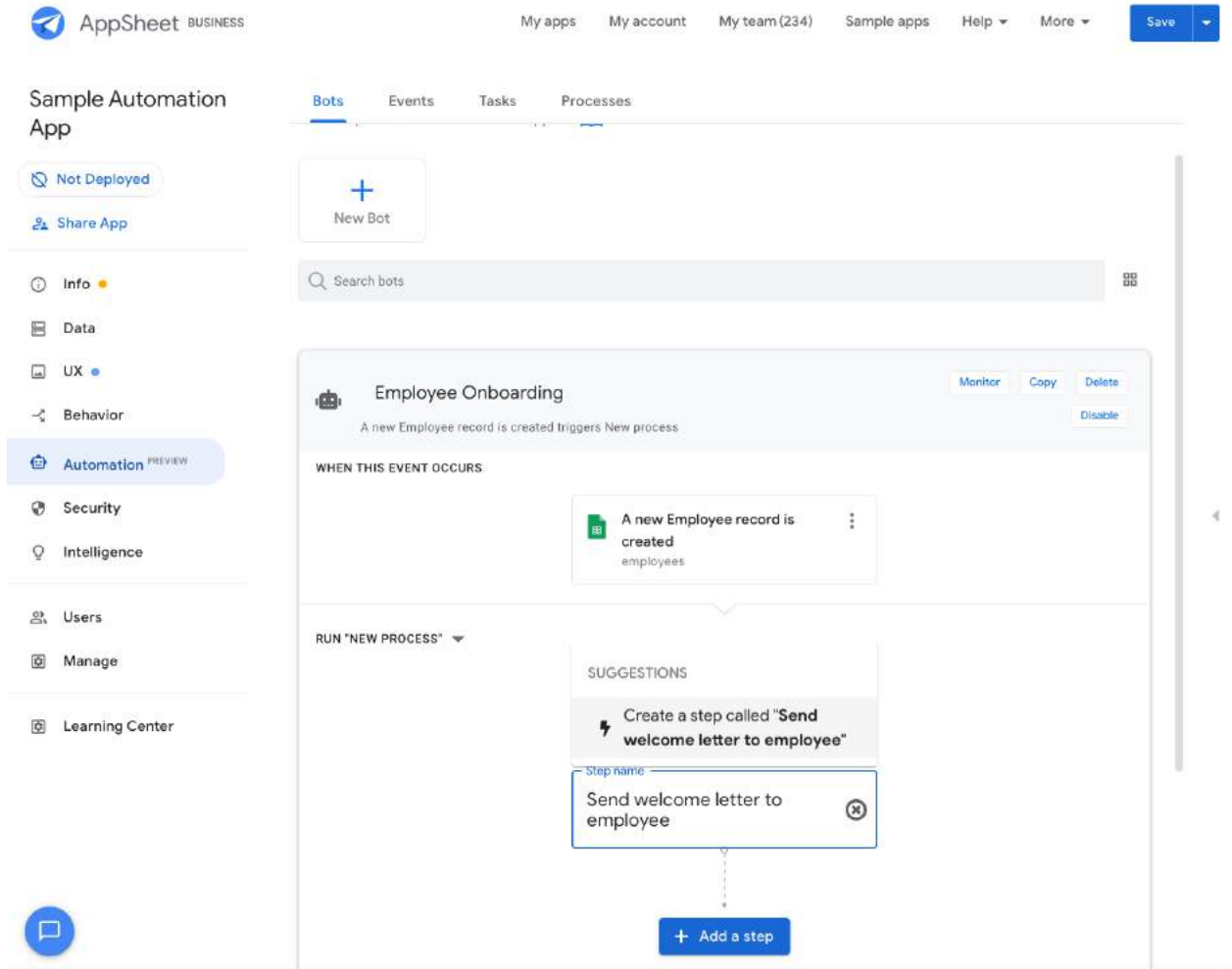
Once your event has been created, your bot editor should look like this:

The screenshot displays the AppSheet interface for configuring automation. On the left, a sidebar lists various app management options, with 'Automation' (marked as a preview) selected. The main workspace is titled 'Sample Automation App' and shows a 'Bots' tab. A 'New Bot' button is visible, along with a search bar for bots. The 'Employee Onboarding' bot is currently being edited, showing an 'Incomplete Bot' status. It is configured with the event 'A new Employee record is created' (triggered on the 'employees' table). Below the event trigger, the 'RUN THIS PROCESS' section contains an 'Add a step' button. At the bottom, there are sections for 'Appearance' and 'Documentation' settings.

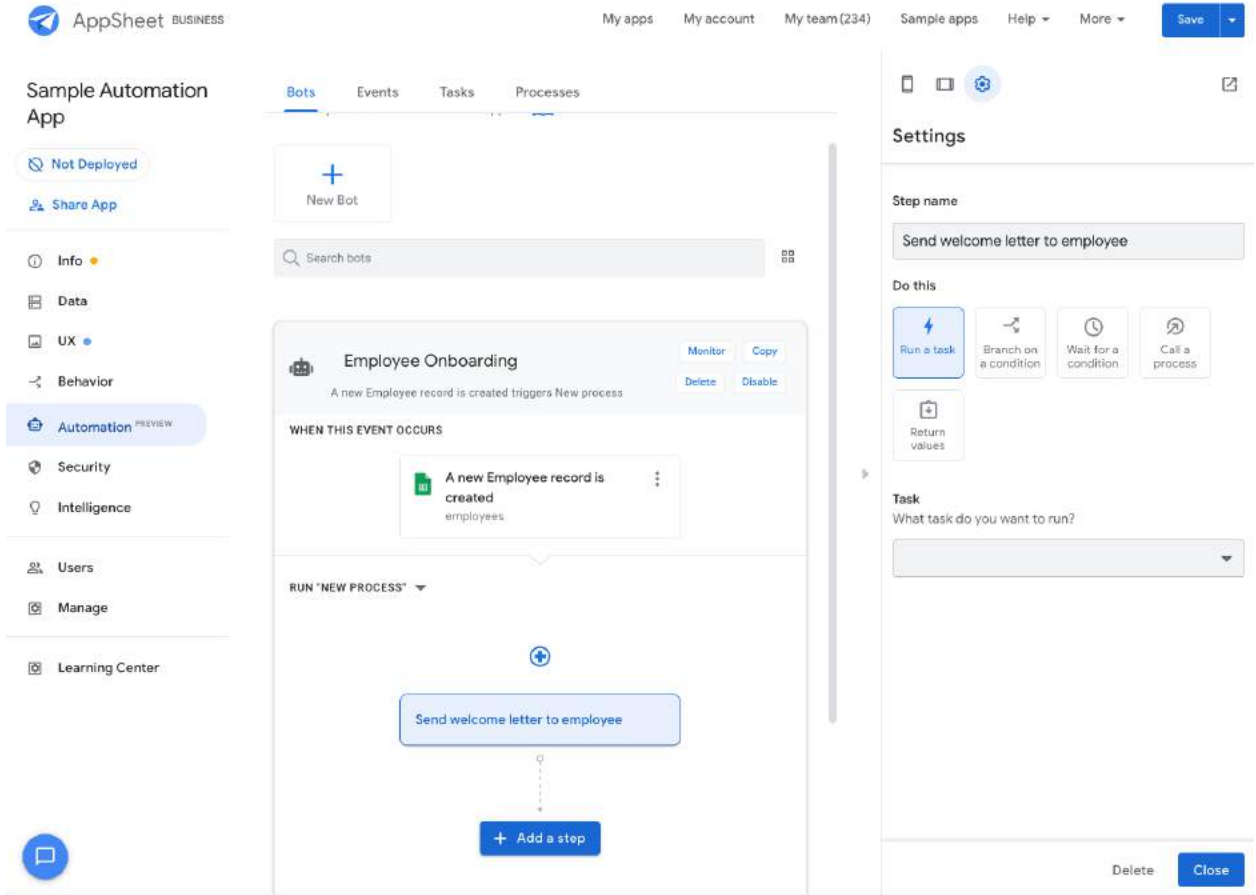
Configuring the process

Configure the process your event should trigger:

1. In the **Run this process** section of the editor, click **+Add a step**. Type `Send welcome letter to employee` into the text box and hit **enter** to automatically create an empty step. This implicitly creates a process.



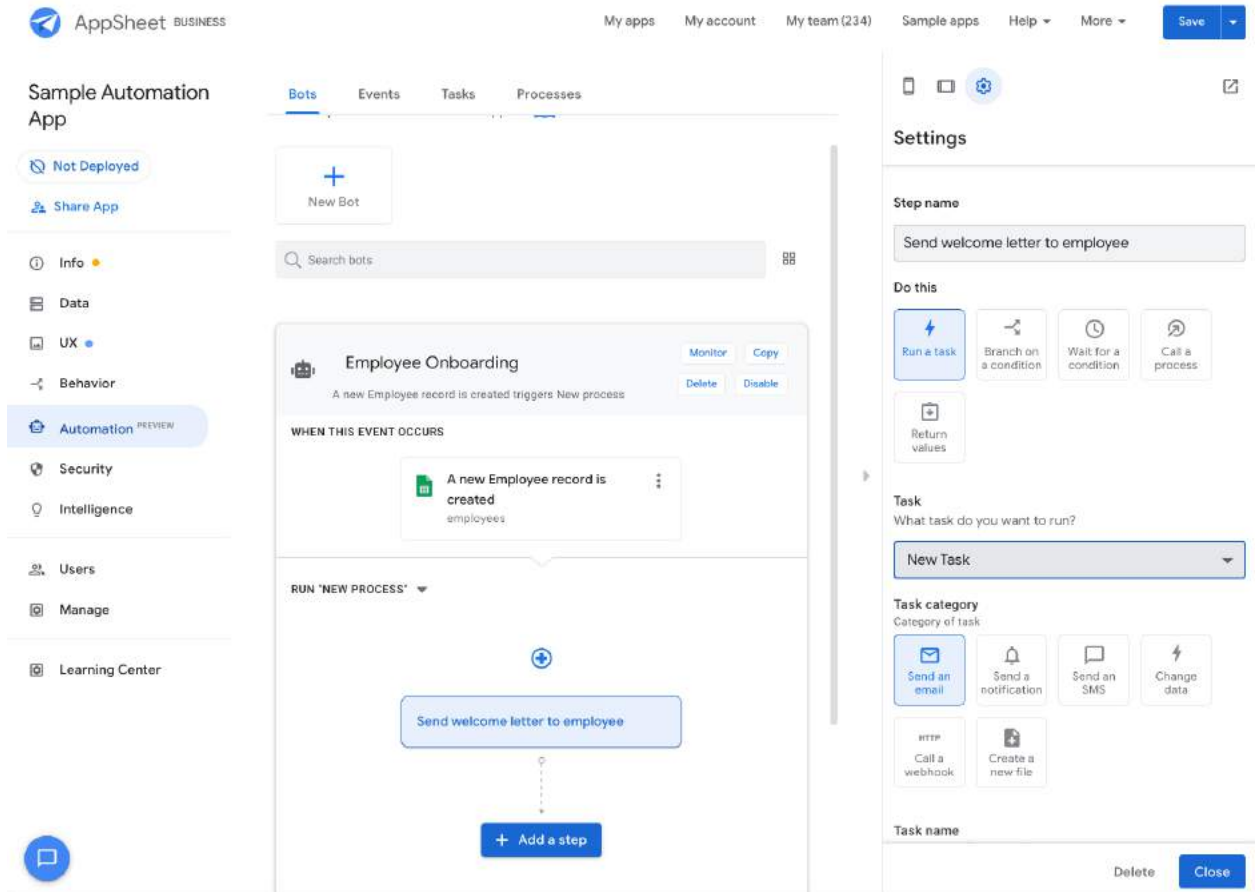
2. Click on the **Send welcome letter to employee** step to expand its definition, as shown below:



3. In the **Settings** pane, configure the step as follows:
 - a. In the **Do this** section, select **Run a task**.
 - b. In the **Task** section, click **Create new task**.
 - c. Refer to the table below for the rest of the step configuration:

Field Name	Value
Do this	Run a task
Task category	Send an email
Task name	Email welcome letter
Table Name	employees
To	[Email]

Use default content?	Disabled
Email Subject	Welcome onboard
Email body	<p><<TODAY()>></p> <p>Dear << CONCATENATE ([First Name], “ “, [Last Name])>>>,</p> <p>We are excited to have you on board. We are delighted to have you join us as a <<[Job Title]>> in our team. Your role is critical in fulfilling our mission.</p> <p>The Human Resources team is here to support you throughout your career.</p> <p>Sincerely, HR Manager</p>



The screenshot displays the AppSheet Automation interface. On the left, a navigation menu includes options like 'Info', 'Data', 'UX', 'Behavior', 'Automation', 'Security', 'Intelligence', 'Users', 'Manage', and 'Learning Center'. The main area is titled 'Sample Automation App' and shows a workflow for 'Employee Onboarding'. The workflow starts with the event 'A new Employee record is created' in the 'employees' table, which triggers the process 'New process'. A step in the process is 'Send welcome letter to employee'. A 'Settings' panel on the right is open, showing configuration for the 'Email welcome letter' task. The settings include: Task (Email welcome letter), Task category (Send an email), Task name (Email welcome letter), Table name (employees), and Via channel (System Default). The 'To' field is empty, indicating email addresses are to be determined by expressions.

AppSheet BUSINESS

My apps My account My team (234) Sample apps Help More Save

Sample Automation App

Not Deployed Share App

Info Data UX Behavior Automation Security Intelligence Users Manage Learning Center

Bots Events Tasks Processes

New Bot

Search bots

Employee Onboarding

Monitor Copy Delete Disable

A new Employee record is created triggers New process

WHEN THIS EVENT OCCURS

A new Employee record is created employees

RUN "NEW PROCESS"

Send welcome letter to employee

+ Add a step

Settings

Task

What task do you want to run?

Email welcome letter

Task category

Category of task

Send an email Send a notification Send an SMS Change data

HTTP Call a webhook Create a new file

Task name

Unique name for this task

Email welcome letter

Table name

What entity table does this action work against?

employees View Definition

Via channel

The messaging channel to use.

System Default

To

Send email to these email addresses. (Expressions that yield

Delete Close

The screenshot displays the AppSheet Automation interface. On the left, a sidebar lists various app management options: Info, Data, UX, Behavior, Automation (highlighted), Security, Intelligence, Users, Manage, and Learning Center. The main workspace is titled 'Sample Automation App' and shows a workflow for 'Employee Onboarding'. The workflow starts with the event 'A new Employee record is created' in a Google Sheet, which triggers the process 'New process'. The process step is 'Send welcome letter to employee'. A 'Settings' panel on the right is open, showing configuration options for the email action, including recipient addresses, subject line ('Welcome onboard'), and email body.

AppSheet BUSINESS My apps My account My team (234) Sample apps Help More Save

Sample Automation App

Not Deployed Share App

Info Data UX Behavior Automation Security Intelligence Users Manage Learning Center

Bots Events Tasks Processes

New Bot

Search bots

Employee Onboarding Monitor Copy Delete Disable

A new Employee record is created triggers New process

WHEN THIS EVENT OCCURS

A new Employee record is created employees

RUN 'NEW PROCESS'

Send welcome letter to employee

+ Add a step

Settings

To Send email to these email addresses. (Expressions that yield email addresses are supported.)

[Email] T A

Add

Use default content? If enabled, the action constructs a sensible default message. If disabled, you should define the content explicitly.

Email Content

Email Subject Subject line for the email. (Template that yields text is supported. Leave empty for default email subject.)

Welcome onboard

Email Body The email body. (Template that yields text or HTML is supported. Leave empty for default email body.)

Delete Close

The screenshot displays the AppSheet Business interface. On the left, a navigation menu includes options like 'Info', 'Data', 'UX', 'Behavior', 'Automation' (highlighted), 'Security', 'Intelligence', 'Users', 'Manage', and 'Learning Center'. The main workspace is titled 'Sample Automation App' and shows a 'New Bot' button and a search bar. The central area is titled 'Employee Onboarding' and contains a workflow diagram. The trigger is 'A new Employee record is created employees'. The process is 'RUN 'NEW PROCESS'' and includes a step 'Send welcome letter to employee'. On the right, a 'Settings' panel is open, showing options for 'Use default content?' (disabled), 'Email Content' (with subject 'Welcome onboard' and body containing 'Dear << CONCATENATE ([First Name], " ", <<TODAY()>>' and an 'Add' button), and 'CC' (with an 'Add' button). A 'Save' button is visible in the top right corner.

4. Click **Save** to save all your changes, click **Close** to hide the right pane.

Your bot should now look like this:

The screenshot displays the AppSheet Business interface. At the top, there are navigation links for 'My apps', 'My account', 'My team (234)', 'Sample apps', 'Help', and 'More'. A 'Save' button is visible in the top right corner. On the left, a sidebar menu includes 'Sample Automation App', 'Not Deployed', 'Share App', 'Info', 'Data', 'UX', 'Behavior', 'Automation (PREVIEW)', 'Security', 'Intelligence', 'Users', 'Manage', and 'Learning Center'. The main content area is titled 'Bots' and contains a 'New Bot' button and a search bar. Below this, a detailed view of a bot named 'Employee Onboarding' is shown. The bot's trigger is 'A new Employee record is created triggers New process'. The configuration is divided into two sections: 'WHEN THIS EVENT OCCURS' and 'RUN "NEW PROCESS"'. The event section shows a trigger card for 'A new Employee record is created employees'. The process section shows a task card 'Send welcome letter to employee' followed by an 'Add a step' button and an 'END' label.

You have created and enabled your bot. All the required components, including the event, process, and tasks, have been created. Each component can be accessed and edited from individual tabs if required.

Note: Make sure your app is **deployed**. If not, deploy it for the bot to run. The bot listens for events and trigger processes only if the app is deployed and the specific bot is enabled.

If you explore the Events, Tasks and Processes tabs, you can see the reusable artifacts that have been generated when you were creating your bot. Each of these components can be re-used inside processes and inside other bots.

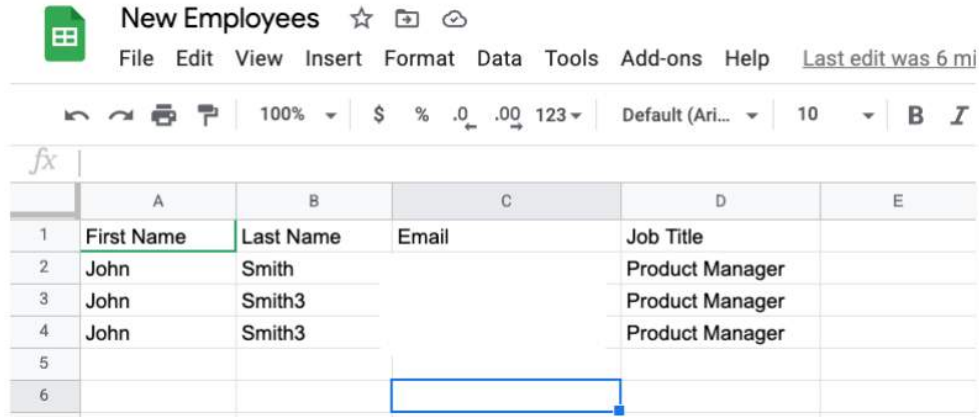
Now you are ready to test your bot.

Testing your automation

Follow these steps to test your automation:

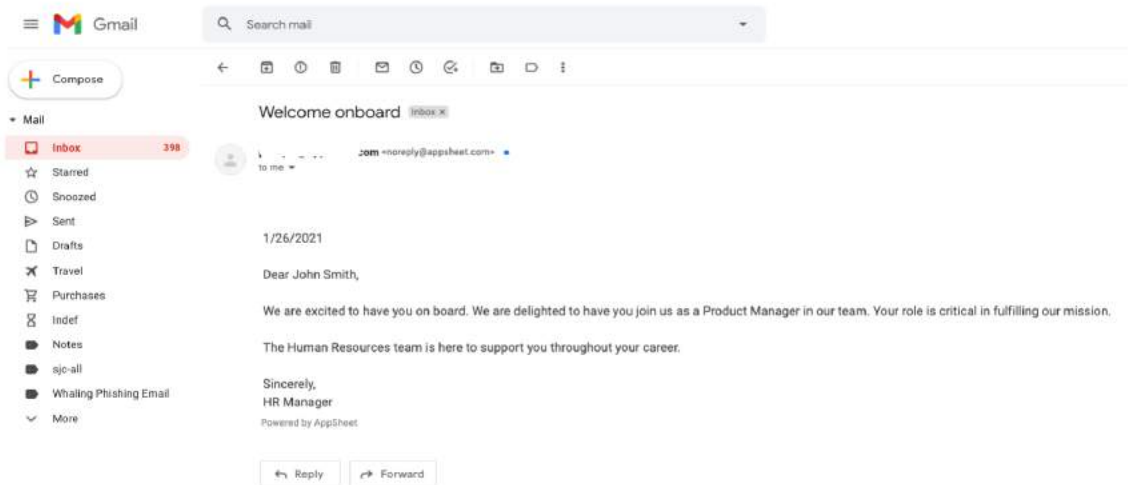
1. Make sure your bot is **Enabled**.

2. Add a new employee row to your sheet. You can copy a whole row (from a separate sheet) and paste it into your employees sheet or you can manually enter values in each cell



The ADD event is fired as soon as you are done entering values in the row (currently the add-on has a 20 second wait as a default wait time to allow for data entry manually)

3. As soon as the event is detected, the bot should send an email at the “To” address configured in the task:



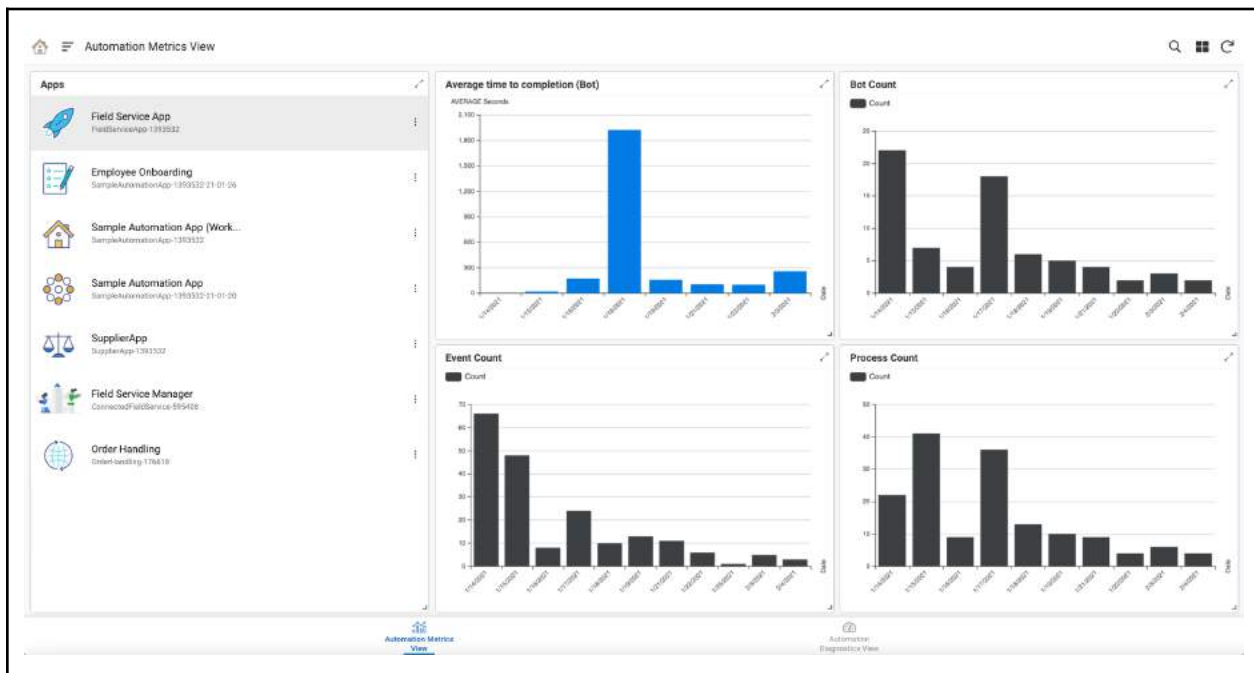
Automation Monitoring

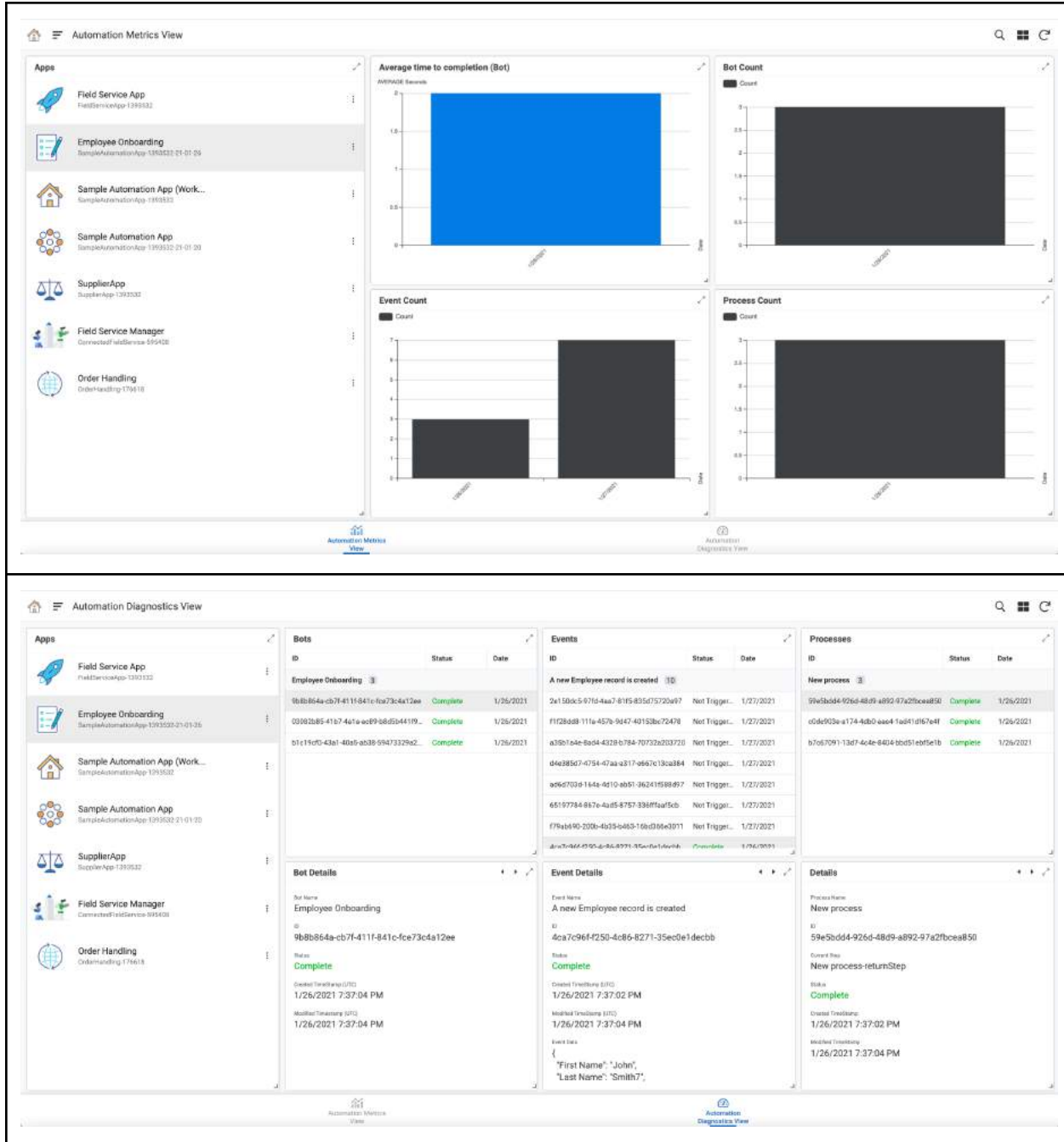
As various bots are configured, events are detected and processes start executing, it becomes important to keep a track of which executions were successful, which ones are pending and

which ones encountered errors. Automation monitoring is delivered as an AppSheet app that can be easily accessed by clicking **Monitor** on your bot definition.

In the current version of the monitoring app, you have two views 1) Automation Metrics View (High level metrics for bots/events/processes) and 2) Automation Diagnostics View (Bots/Events/Processes execution logs). We are constantly adding new views/features to this monitoring app to make it even more useful.

Note: Your view may look different from the screen shots presented below. Time is in UTC.





Contact Sync

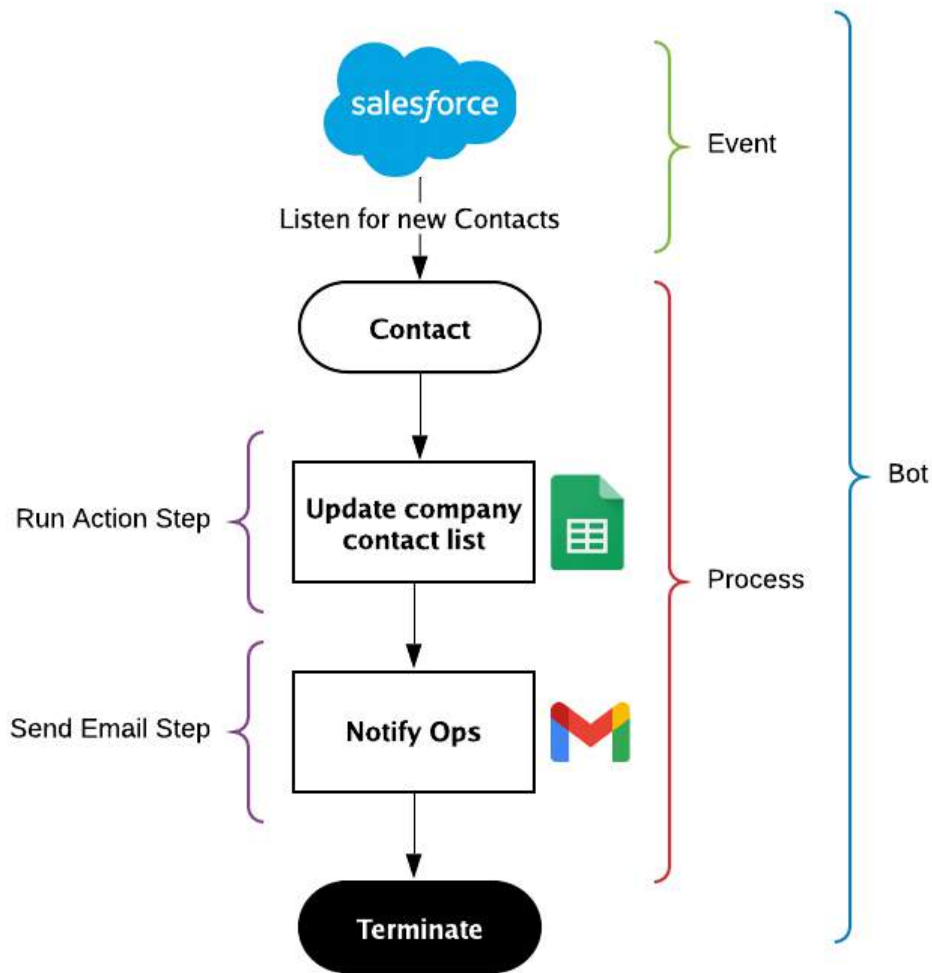
This brief tutorial shows you how to set up an automated process to sync contacts between two systems. This example implements a solution to a common business process scenario. When a new contact is created in Salesforce, AppSheet automatically updates a Google Sheet

with the new contact.

Implementing contact sync with AppSheet Automation involves the following steps:

1. [Creating the Bot](#)
2. [Configuring the Event](#)
3. [Configuring the Process \(steps and actions\)](#)

The following figure describes the high level components:



Before you begin this tutorial:

- Make sure you can log into the AppSheet Editor UI and access the Sample Automation App you created as a prerequisite in [Before you begin](#).
- Follow the steps in the Appendix to configure a [Contact entity](#) from a Salesforce data

source and [Company Contacts](#) entity from a Google Sheets data source.

Automations begin with configuring the bot. A bot binds an event (ex: a new contact created in Salesforce) to a process (ex. a sequence of steps to update a contact Sheet). The intuitive AppSheet editor enables you to create all the components of the automation/bot in-situ. That is, you can configure the event and process (including steps/tasks) without needing to switch between tabs to configure individual components.

In addition, AppSheet Automation is an intent-aware platform. The platform understands user intent and recommends configuration options that align with what you are trying to achieve.

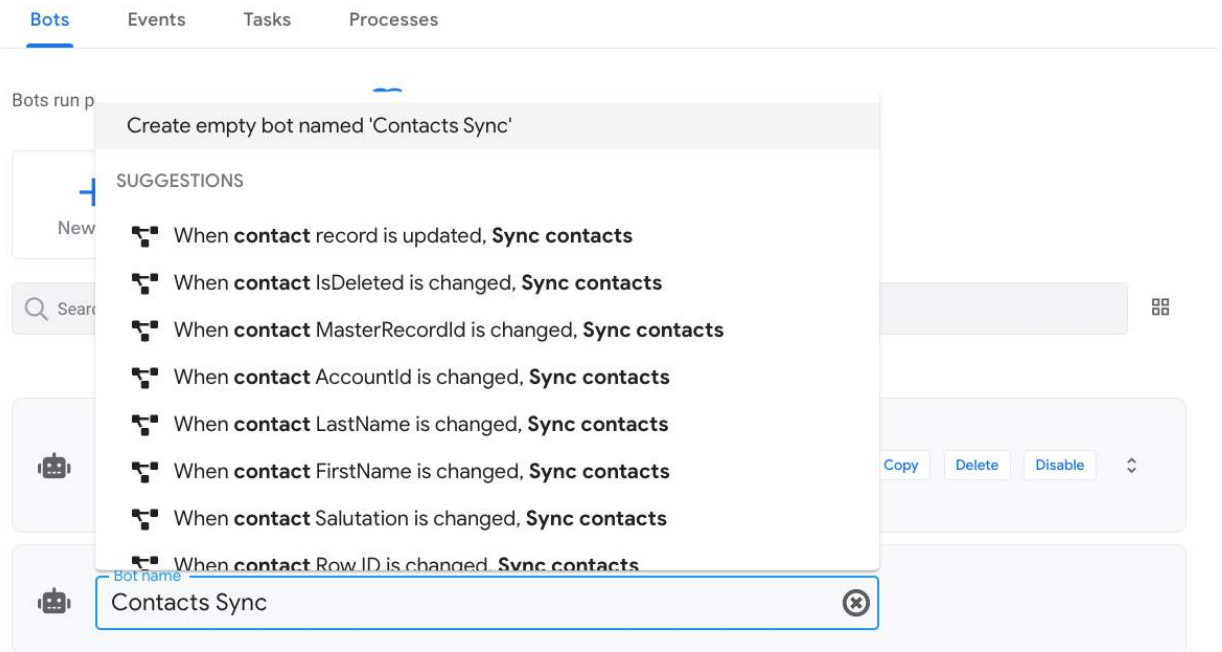
Note: *The suggestions you see depend on your data configuration, existing event configurations and may appear different from what you see in some of the images below.*

The following sections guide you through configuring your end to end automation:

Creating the bot

Follow the steps below to create a new bot:

1. From the AppSheet UI, navigate to the **Bots** tab and click **New Bot**.
2. Type `Contacts Sync` into the dialog and suggested actions appear, as shown in the image below.
3. Select **Create empty bot named 'Contacts Sync'** to create the bot.



Configuring the event

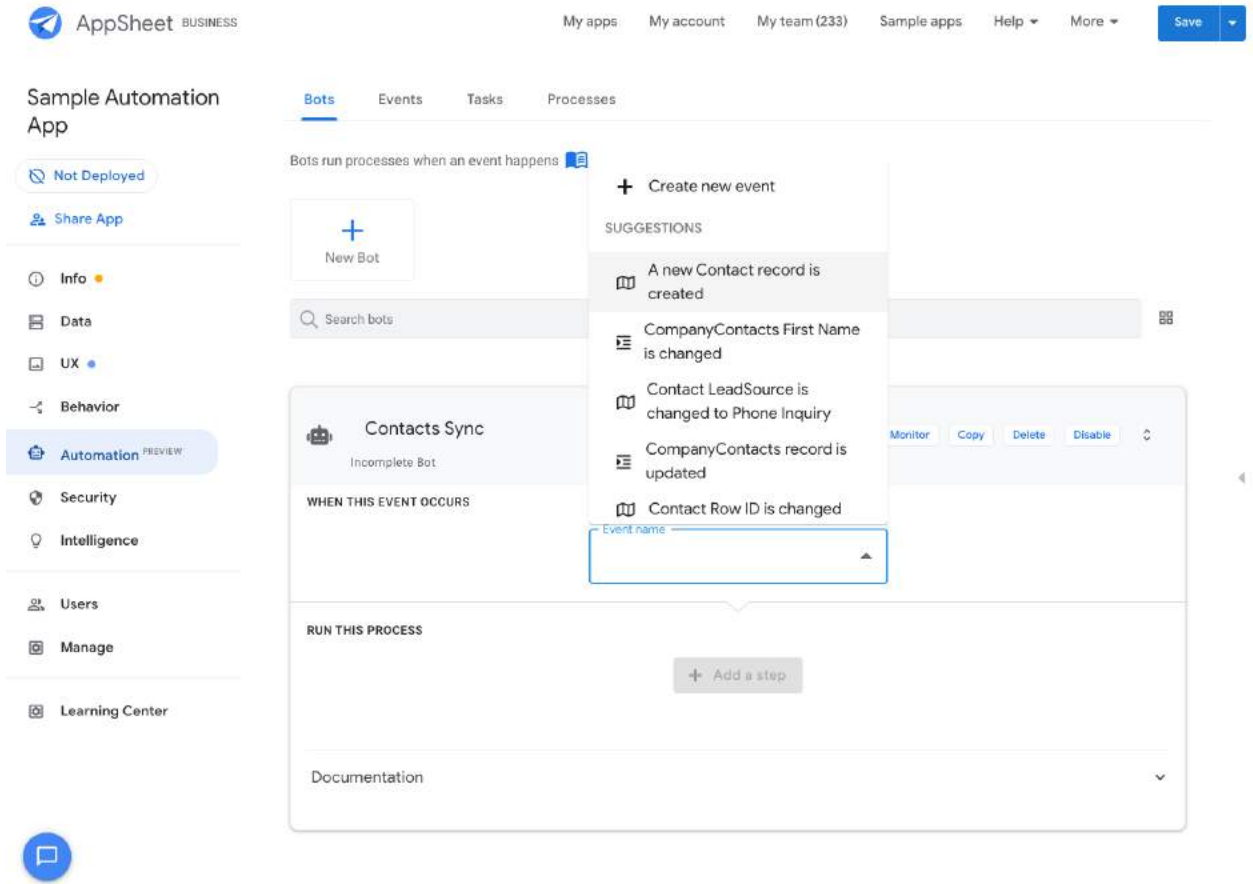
Once the bot is created, you can define the event for the bot. In this example, the event trigger is the addition of new contacts to Salesforce.

To create an event:

1. Select the bot you just created in the previous section.
2. Click **Choose an event**.
3. Suggested events appear.
4. Select **A new Contact record is created** to have the event automatically created for you. The platform does this for you and saves you valuable time.

The screenshot displays the AppSheet interface for configuring a bot. The top navigation bar includes 'AppSheet BUSINESS', 'My apps', 'My account', 'My team (233)', 'Sample apps', 'Help', 'More', and a 'Save' button. The main content area is titled 'Sample Automation App' and shows a 'Bots' tab selected. A 'New Bot' button is visible. Below it, a search bar for bots is present. The main configuration area is for a bot named 'Contacts Sync' (Incomplete Bot). It features a 'Choose an event' button under the 'WHEN THIS EVENT OCCURS' section and an 'Add a step' button under the 'RUN THIS PROCESS' section. A 'Documentation' link is at the bottom. The left sidebar contains various settings like 'Info', 'Data', 'UX', 'Behavior', 'Automation', 'Security', 'Intelligence', 'Users', 'Manage', and 'Learning Center'.

Note: If you want to manually configure the event, you can navigate to the **Events** tab, select **Create new event**, and enter in the relevant details.

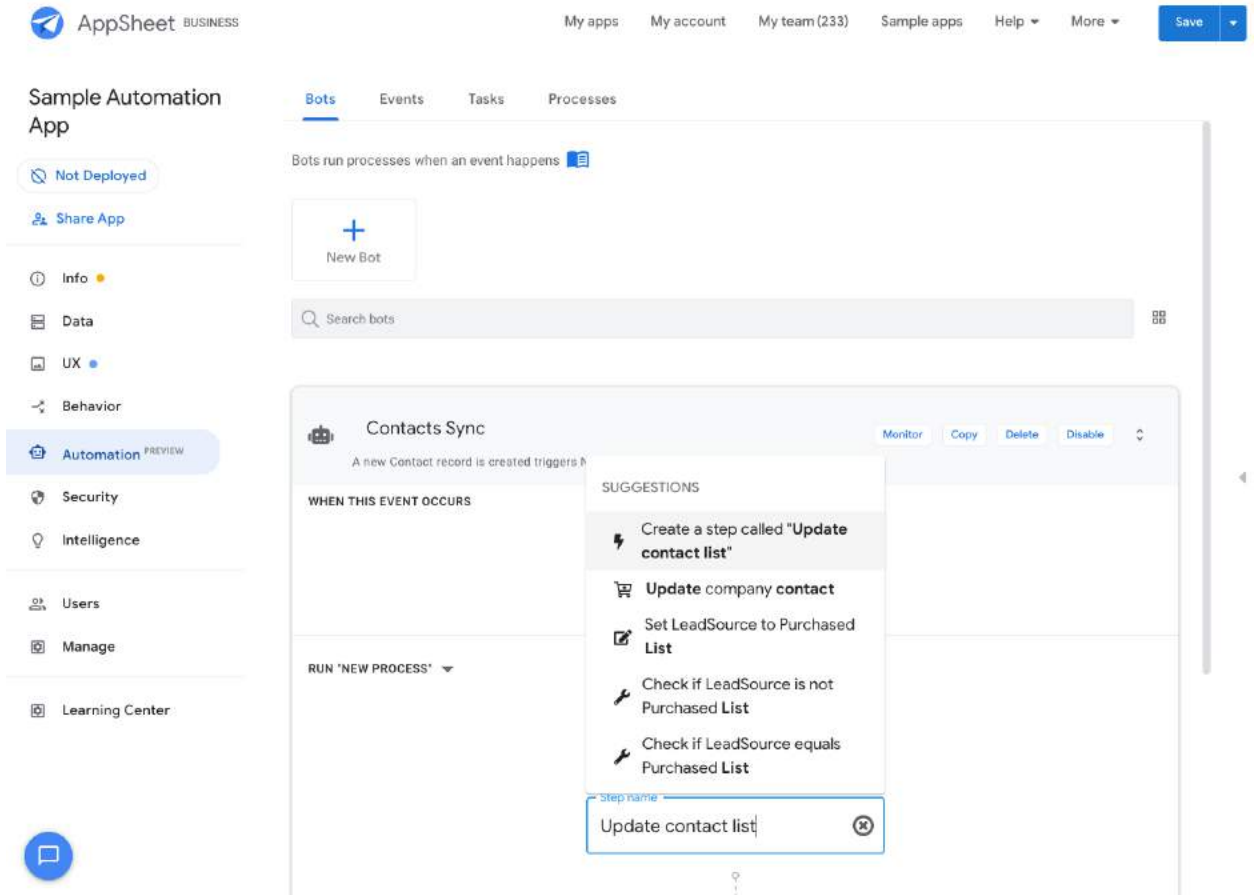


Once your event has been created, your bot editor should look like this:

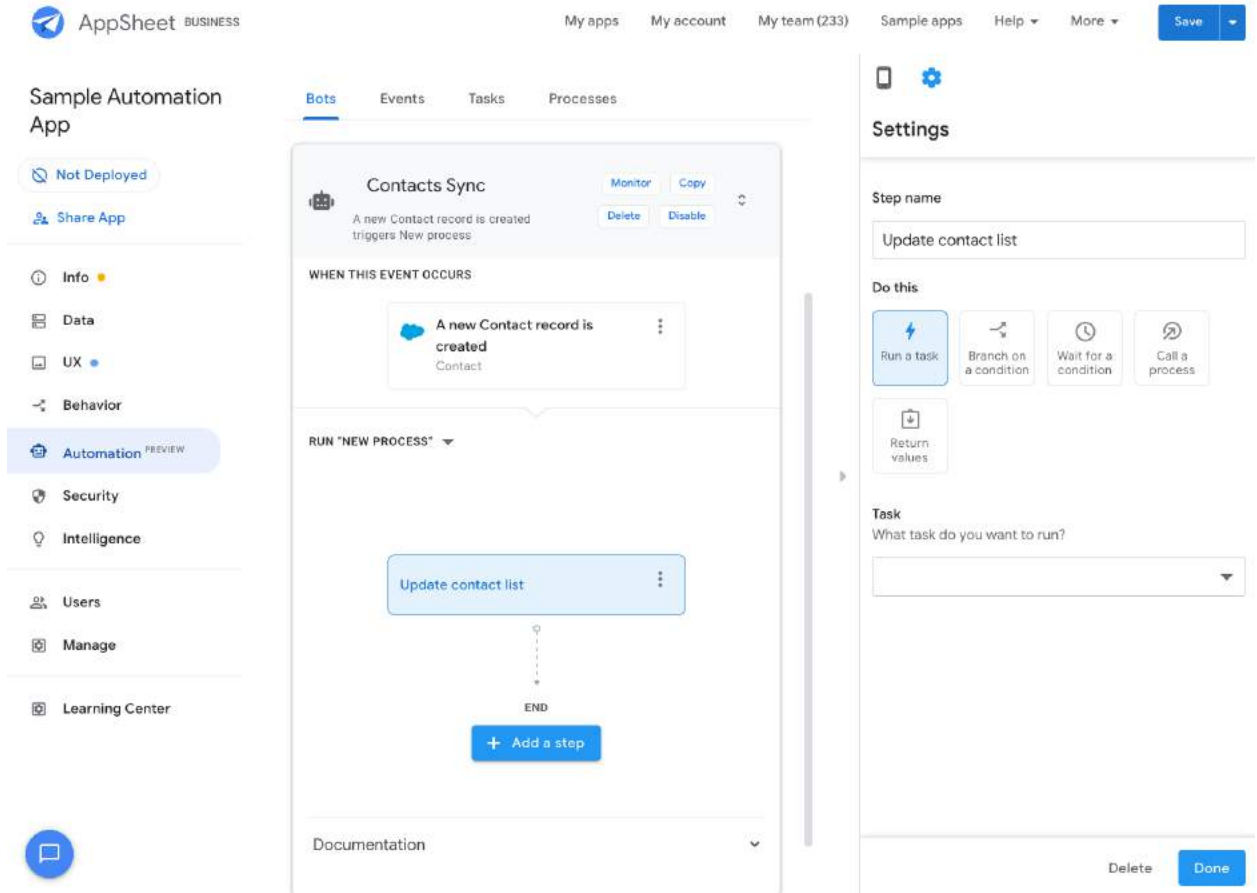
Configuring the process

Configure the process your event should trigger:

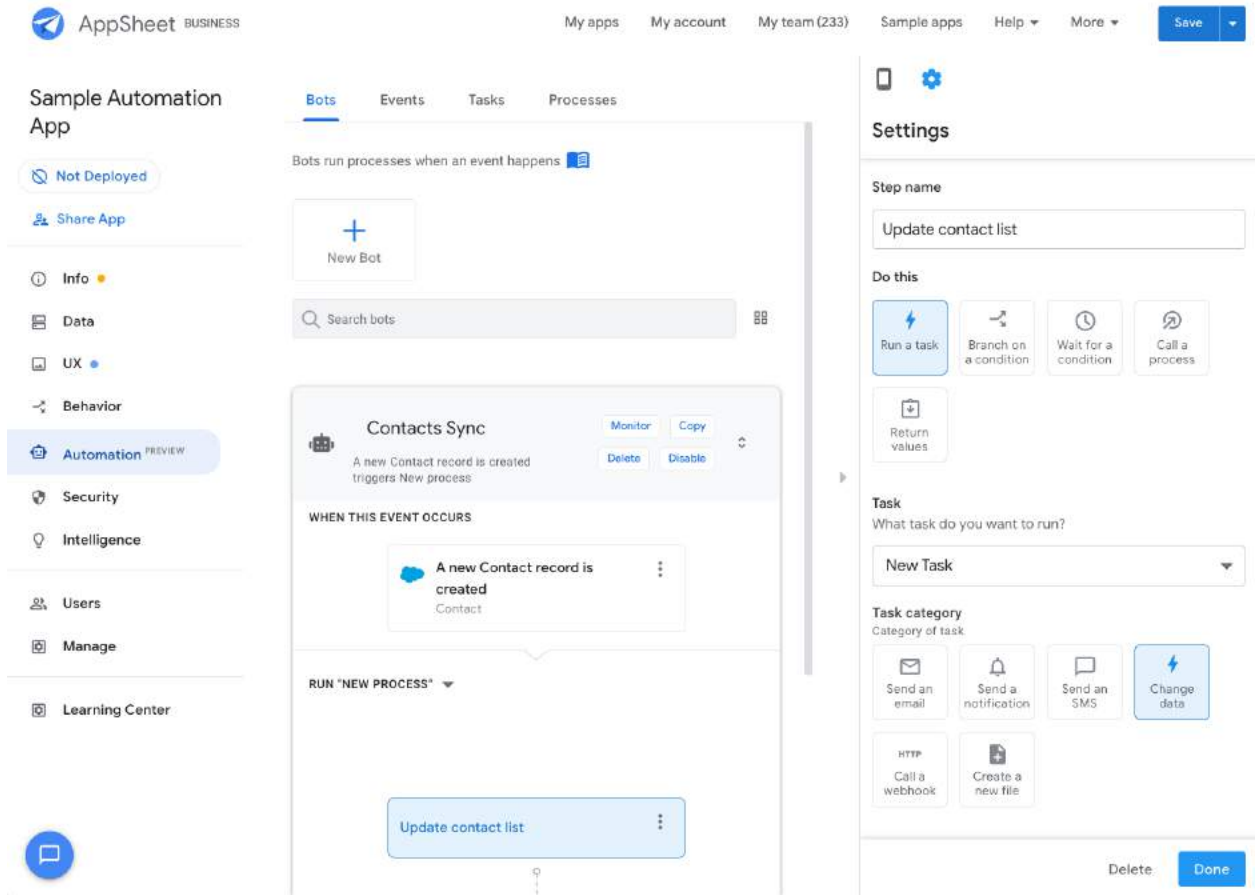
1. In the **Run this process** section of the editor, click **+Add a step**. Type `Update contact list` into the text box and hit **enter** to automatically create an empty step. This implicitly creates a process.



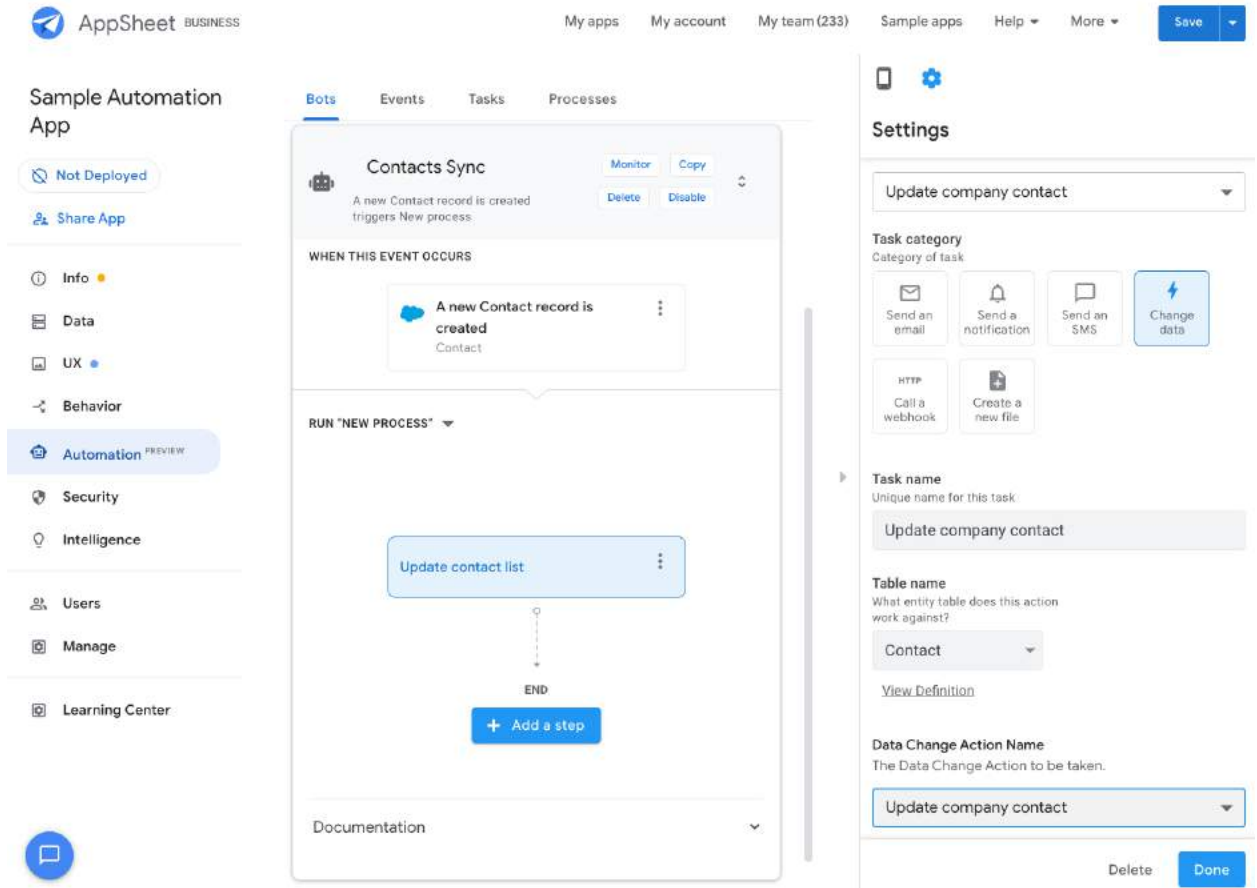
2. Click on the **Update contact list** step to expand its definition, as shown below:



3. In the **Settings** pane, configure the step as follows:
 - a. In the **Do this** section, select **Run a task**.
 - b. In the **Task** section, click **Create new task**.
 - c. And under **Task Category**, select **Change data**.



- d. Type Update company contact as the **Task name**.
- e. Select **Contact** as the **Table name**.
- f. Select **Update company contact** as the **Data Change Action Name**.



g. Click **Save** to save all your changes.

4. Add another step to notify the Ops team of the new contact:
 - a. Click **+ Add a step**.
 - b. Type `Notify ops` as the step name and select **Create a step called Notify Ops** from the suggested options.

The screenshot displays the AppSheet interface for configuring an automation. On the left, a sidebar shows navigation options like 'Info', 'Data', 'UX', 'Behavior', 'Automation', 'Security', 'Intelligence', 'Users', 'Manage', and 'Learning Center'. The main workspace shows a process named 'Contacts Sync' with a trigger 'A new Contact record is created' and a task 'Update contact list'. Below the task, a step named 'Notify ops' is being configured. On the right, the 'Settings' pane is open, showing options for 'Task category' (Send an email, Send a notification, Send an SMS, Change data), 'Task name' (Update company contact), 'Table name' (Contact), and 'Data Change Action Name' (Update company contact).

- c. In the **Settings** pane, under **Do this** select **Run a task** .
- d. From the **Task** dropdown, select **New Task**.
- e. Select **Send an email** as the **Task category**.

The screenshot displays the AppSheet interface for configuring an automation. On the left, a sidebar lists various app management options, with 'Automation' highlighted. The main workspace shows a process named 'Contacts Sync' with the trigger 'A new Contact record is created'. The process flow includes 'Update contact list' and 'Notify ops' steps. On the right, the 'Settings' panel is open, showing options for 'Do this' (Run a task, Branch on a condition, Wait for a condition, Call a process) and 'Task category' (Send an email, Send a notification, Send an SMS, Change data, Call a webhook, Create a new file). The 'Task name' field is currently empty, and the 'Task category' is set to 'New Task'.

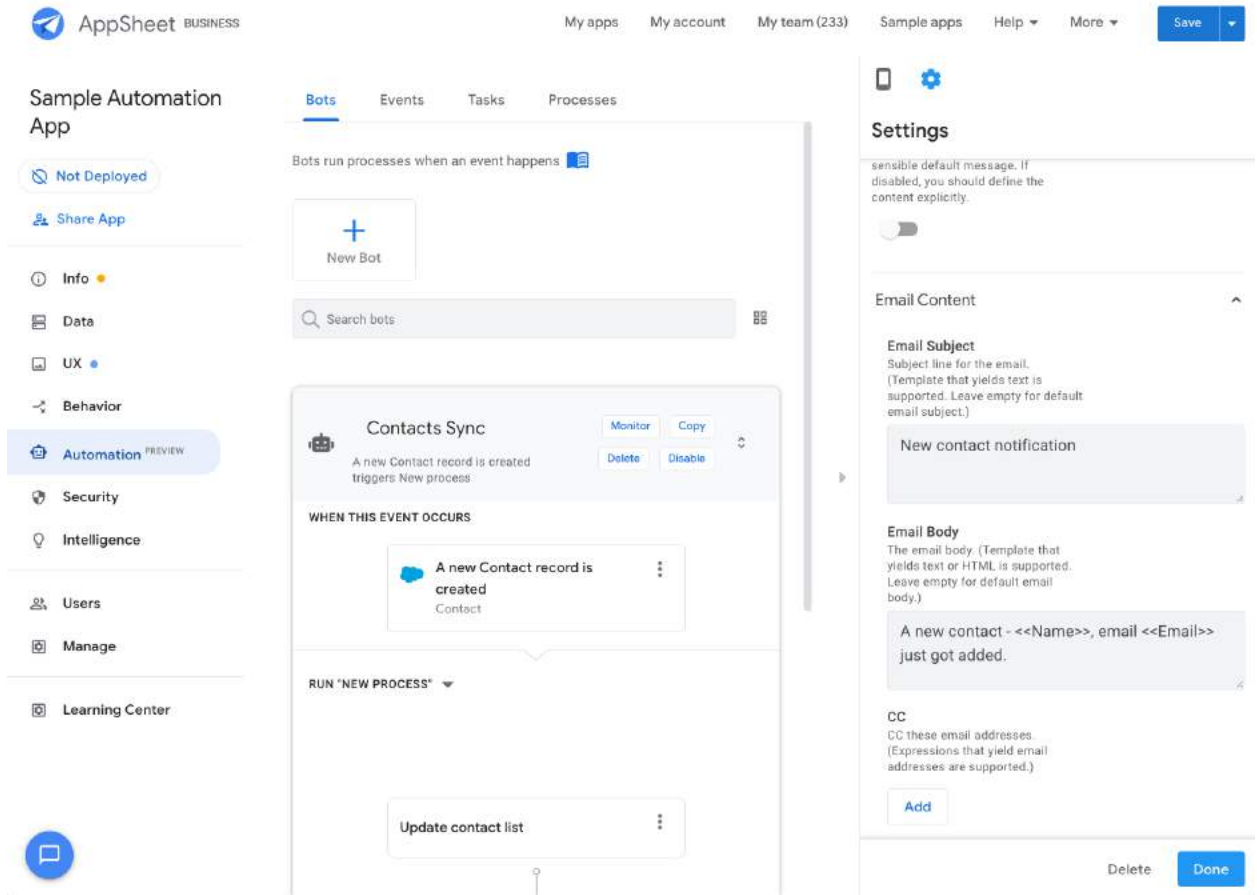
- f. Type Send an email to ops for the **Task name**.
- g. Select **Contact** as the **Table name**.

The screenshot displays the AppSheet interface for configuring an automation. On the left, a sidebar lists various app management options, with 'Automation' highlighted. The main workspace shows a process named 'Contacts Sync' triggered by the event 'A new Contact record is created'. The process flow includes two steps: 'Update contact list' followed by 'Notify ops', which then leads to an 'END' terminal. A 'Settings' panel on the right is open, showing configuration options for the 'Notify ops' step, including 'Task name', 'Table name' (set to 'Contact'), and 'Via channel' (set to 'System Default').

h. Enter <your email address> in the **To** field.

The screenshot displays the AppSheet interface for configuring an automation. On the left, a sidebar lists various app management options, with 'Automation' highlighted. The main workspace shows a workflow for 'Contacts Sync' under the 'Bots' tab. The workflow starts with the event 'A new Contact record is created' and proceeds to the task 'Update contact list', followed by 'Notify ops', and ends at 'END'. A 'Settings' panel on the right is open, showing configuration options for the 'Notify ops' task. The settings include a task name, table name (set to 'Contact'), a channel (set to 'System Default'), and a list of email addresses (currently containing 'no-reply@appsheet.com').

- i. For **Email Subject**, type New contact notification.
- j. For **Email Body**, type A new contact - <<Name>>, email <<Email>> just got added. Leave all the other fields at their default values.



5. Click **Save** and then click **Done**.

Your bot should now look like this:

The screenshot displays the AppSheet Business interface. On the left, a sidebar menu includes options like 'Info', 'Data', 'UX', 'Behavior', 'Automation' (highlighted with a 'PREVIEW' badge), 'Security', 'Intelligence', 'Users', 'Manage', and 'Learning Center'. The main area shows a bot configuration for 'Contacts Sync'. The bot is currently 'Not Deployed'. The configuration is divided into two sections: 'WHEN THIS EVENT OCCURS' and 'RUN "/>

You have created and enabled your bot. All the required components, including the event, process, and tasks, have been created. Each component can be accessed and edited from individual tabs if required.

Note: Make sure your app is **deployed**. If not, deploy it for the bot to run. The bot listens for events and trigger processes only if the app is deployed and the specific bot is enabled.

If you explore the Events, Tasks and Processes tabs, you can see the reusable artifacts that have been generated when you were creating your bot. Each of these components can be re-used inside processes and inside other bots.

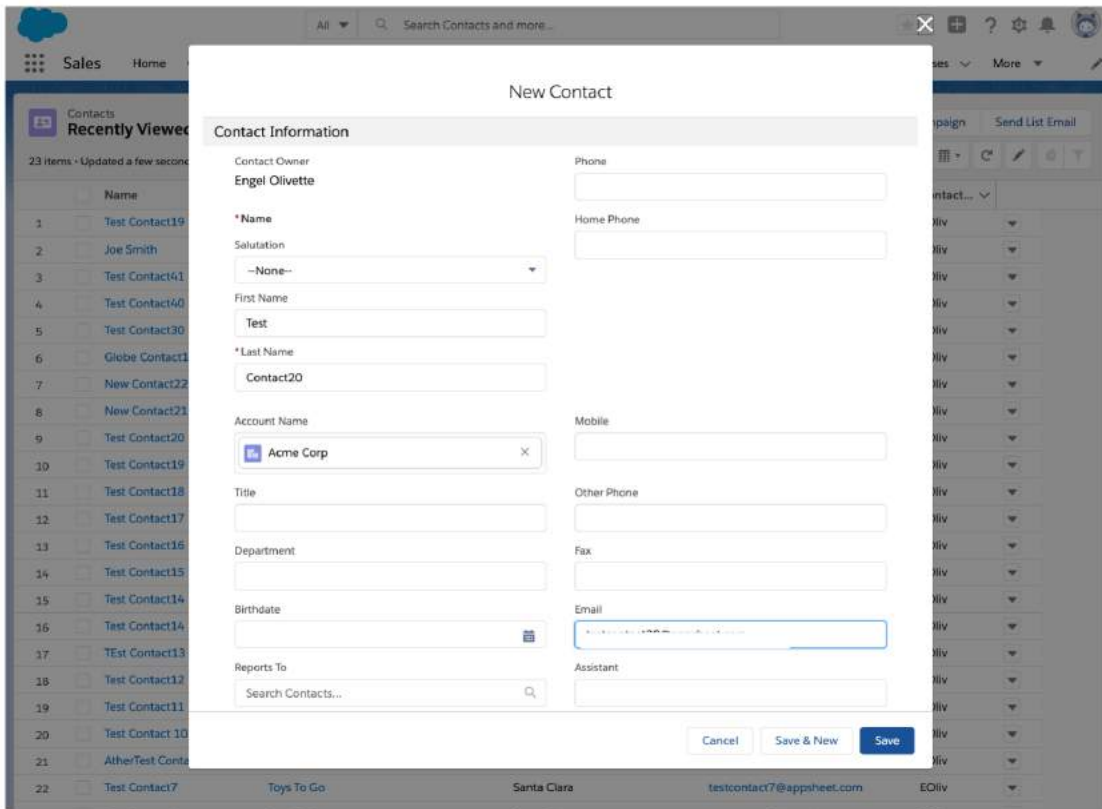
Now you are ready to test your bot.

Testing your automation

Follow these steps to test your automation:

1. Make sure your bot is **Enabled**.

2. Login to your Salesforce Instance and create a new contact.

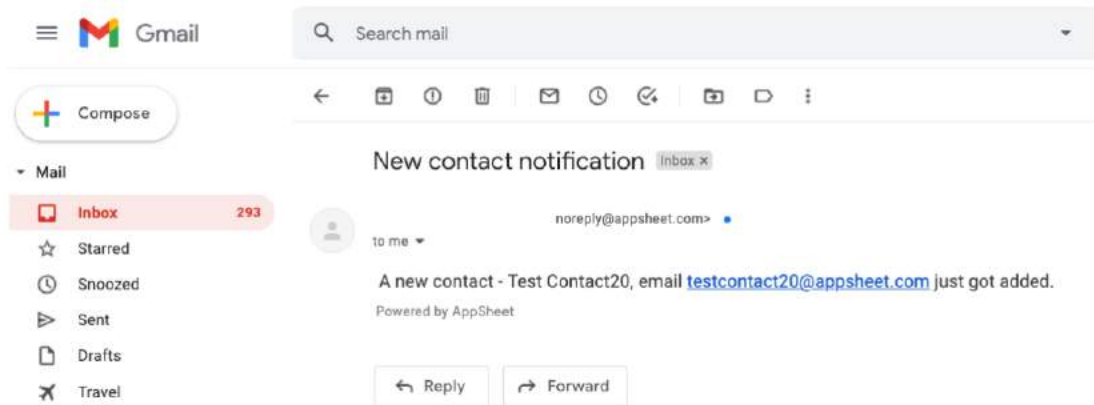


The Bot listens for new Contact events every 60 seconds.

3. In about 60 seconds, check to confirm that a new contact is added to your CompanyList Sheet.

	A	B	C	D	E
1	First Name	Last Name	Email	Company	Date Created
2	Test	Contact11	.com	0016g00000Nno8PAAR	10/31/2020
3	Test	Contact12	.com	0016g00000Nno8PAAR	10/31/2020
4	Test	Contact13	.com	0016g00000Nno8PAAR	11/2/2020
5	Test	Contact14	.com	0016g00000Nno8PAAR	11/4/2020
6	Test	Contact18	.com	0016g00000NnUrgAAF	1/19/2021
7	Test	Contact19	.com	0016g00000NnUrgAAF	11/5/2020
8	Test	Contact20	@sheet.com	0016g00000NnUrgAAF	1/20/2021
9					
10					

4. You should also receive an email similar to this:

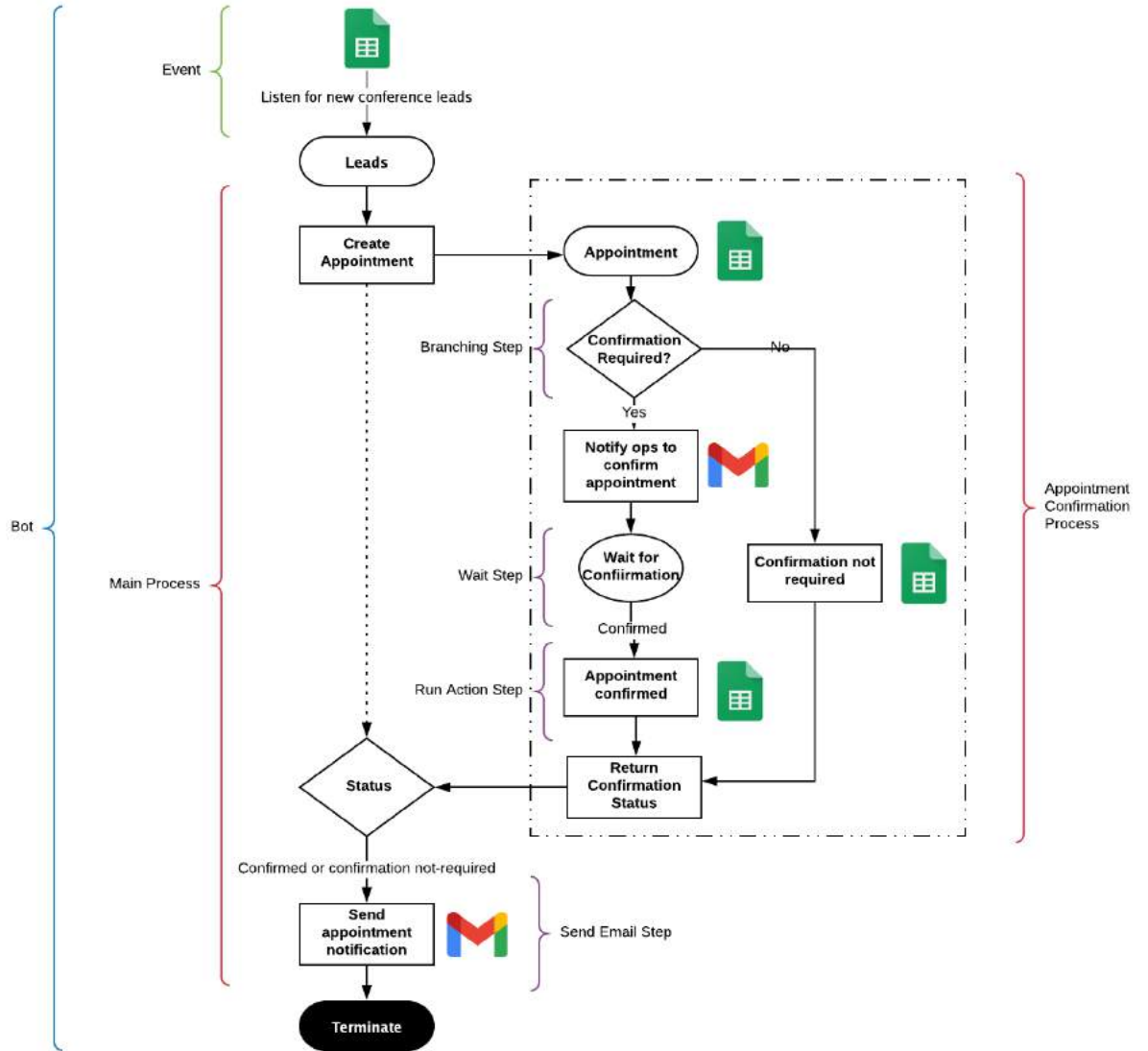


Conference leads appointment generator

This brief tutorial shows you how to set up an automated process to create appointments for leads attending a conference who have requested a follow up. When a new lead is created in Google Sheets, AppSheet determines if a follow up has been requested by the individual. If so, AppSheet automatically creates an appointment entry in another Google Sheet and notify the individual via email.

A specific lead can have a profiled value of “High”, “Medium” or “Low”. If the lead is a “High” profile lead, then a phone based confirmation is required for the lead. Once the appointment is confirmed by phone, a second notification (SMS) is sent to the individual.

The following figure depicts the high level process flow:



Implementing this use case with AppSheet Automation involves the following steps:

1. [Configuring the Appointment Confirmation Process](#) (A sub process invoked from the main process)
2. [Creating the Bot](#)
3. [Configuring the Event](#)
4. [Configuring the Main Process](#) (steps and tasks)

Before you begin this tutorial:

- Configure the [leads](#) and [appointments](#) entities (tables) in your Sample Automation App.
- Install the [AppSheet Events add-on](#) for Google Sheets.

- Configure an [email template](#) for ops notification

The following sections guide you through configuring your end to end automation *in-situ* using a bot.

Configuring the Appointment Confirmation Process

Navigate to the Process Tab from the top.

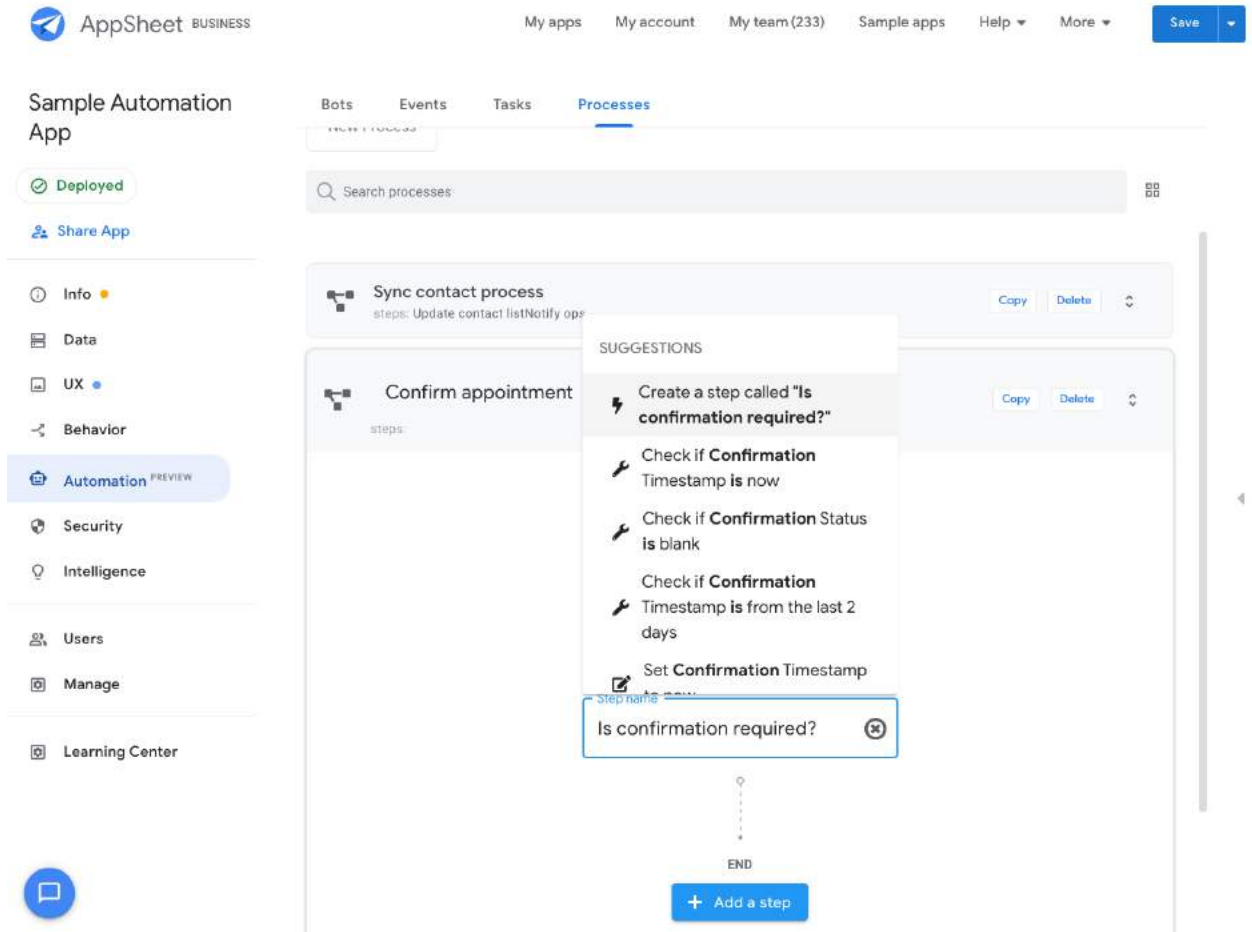
1. Click **+ New Process** to create a new empty process.

The screenshot shows the AppSheet Business interface. The top navigation bar includes 'AppSheet BUSINESS', 'My apps', 'My account', 'My team (233)', 'Sample apps', 'Help', 'More', and a 'Save' button. The left sidebar contains various app management options like 'Info', 'Data', 'UX', 'Behavior', 'Automation (PREVIEW)', 'Security', 'Intelligence', 'Users', 'Manage', and 'Learning Center'. The main content area is titled 'Sample Automation App' and shows the 'Processes' tab selected. A 'New Process' button is visible, and a search bar for processes is present. Below the search bar, there are two process cards: 'Sync contact process' and 'New Process'. The 'New Process' card is expanded, showing an 'Entity' dropdown menu set to 'None' and a '+ Add a step' button. The question 'What type of entity does this process apply to?' is displayed below the dropdown. A 'Documentation' section is also visible at the bottom of the card.

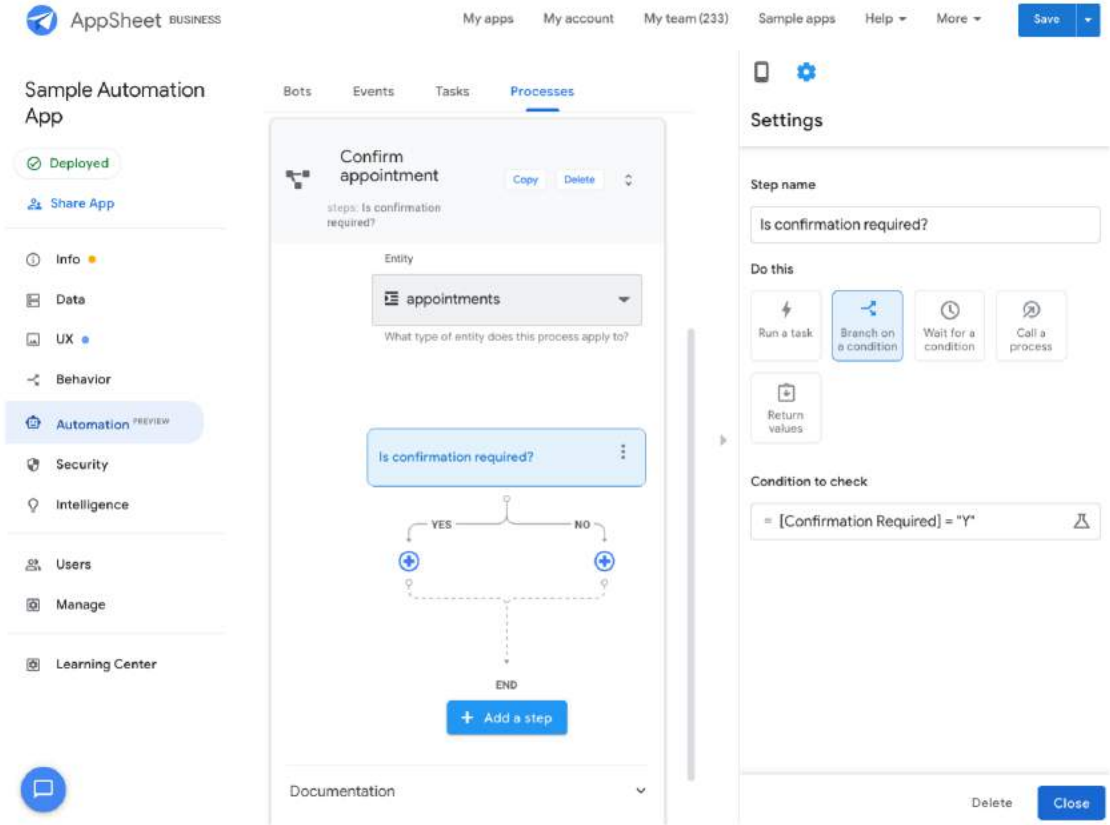
2. Type `Confirm appointment` as the **Process name**.
3. Select **appointments** as the entity name from the **Entity** drop down. At this point your process should look like this:

The screenshot shows the AppSheet interface for a 'Sample Automation App'. The top navigation bar includes 'My apps', 'My account', 'My team (233)', 'Sample apps', 'Help', 'More', and a 'Save' button. The left sidebar contains various app management options, with 'Automation' highlighted. The main content area is titled 'Processes' and shows a list of processes. The 'Confirm appointment' process is selected, and its configuration is displayed. The 'Entity' dropdown is set to 'appointments', and the 'Add a step' button is highlighted with a blue box. The 'Sync contact process' is also visible above it.

4. To add steps to complete the process, click **+ Add a step**.
5. In **Step name**, type `Is confirmation required?` and select **Create a step called 'Is confirmation required?'** from the dropdown.



6. In the **Settings** pane, select **Branch on a condition** under **Do this**.
7. Enter `[Confirmation Required] = "Y"` under **Condition to check**. Your step should look like this:



8. Click the + icon on the **Yes** branch.
9. In the **Settings** pane, enter `Notify ops to confirm appointment` as the **Step Name** and configure this step as follows:

Field Name	Value
Do this	Run a task
Table Name	Appointments
To	<your email address>
Email Subject	Appointment confirmation request
Email body template	Ops Notification Template (selected from Google Drive) Note: For more information, see Configure an email template for ops notification.
Use default content?	Disabled

Leave default settings for other fields. The following screenshots show the step config in detail:

The screenshot displays the AppSheet interface for configuring a process. On the left, a sidebar shows navigation options: Info, Data, UX, Behavior, Automation (highlighted with a 'PREVIEW' badge), Security, Intelligence, Users, Manage, and Learning Center. The main area is titled 'Confirm appointment' and shows a flowchart. The flow starts with an entity 'appointments', followed by a decision step 'Is confirmation required?'. A 'YES' path leads to a task 'Notify ops to confirm appointment', which then leads to 'END'. On the right, a 'Settings' panel is open for the 'Notify ops to confirm appointment' step. It includes fields for 'Step name' (Notify ops to confirm appointment), 'Do this' (Run a task), 'Task' (New Task 2), and 'Task category' (Send an email, Send a notification, Send an SMS, Change data, HTTP Call a webhook, Create a new file). Buttons for 'Delete' and 'Close' are at the bottom right.

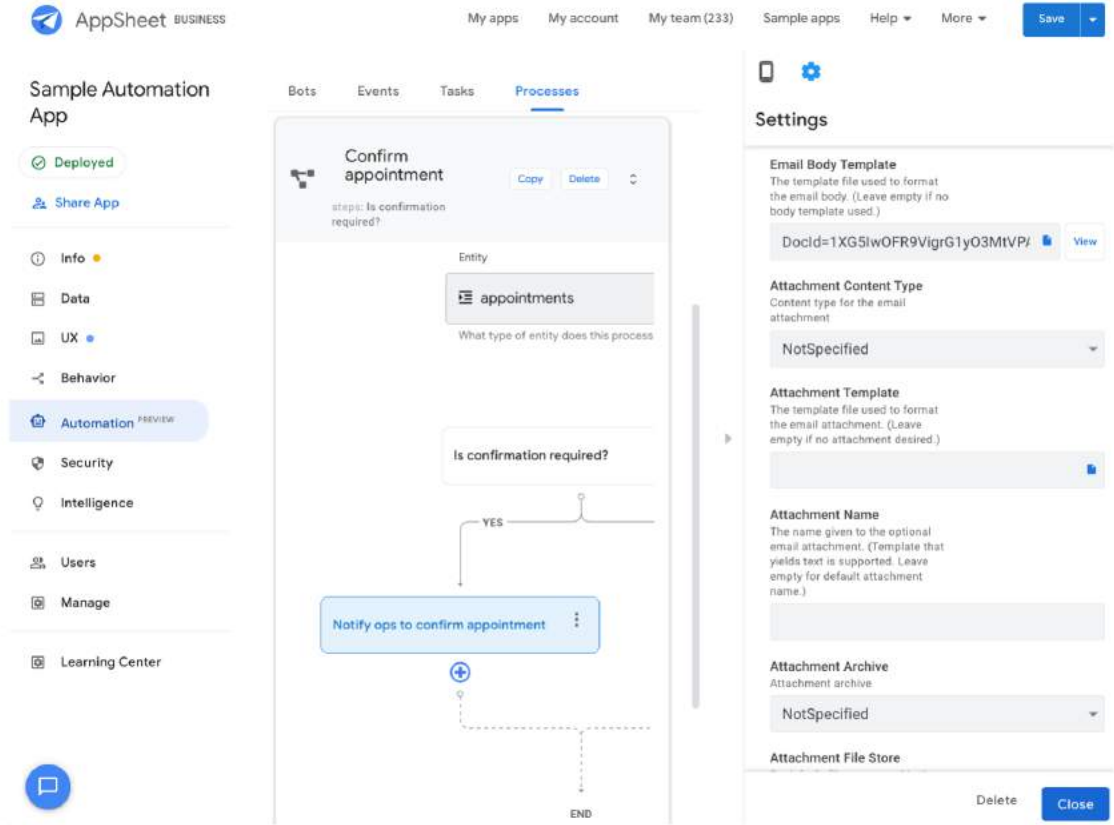
The screenshot displays the AppSheet interface for configuring an automation process. On the left, a sidebar lists various app management options, with 'Automation' highlighted. The main workspace shows a process flow for 'Confirm appointment' with the following steps:

- Entity:** appointments
- Decision:** Is confirmation required? (YES path leads to the next step)
- Action:** Notify ops to confirm appointment
- End:** END

On the right, the 'Settings' panel for the 'Notify ops to confirm appointment' action is visible, including:

- Task name:** Unique name for this task (Notify ops)
- Table name:** What entity table does this action work against? (appointments)
- Via channel:** The messaging channel to use. (System Default)
- To:** Send email to these email addresses. (no-reply@appsheet.com)
- Use default content?** If enabled, the action constructs a sensible default message.

The screenshot displays the AppSheet interface for configuring an automation process. The top navigation bar includes the AppSheet logo, user information (My apps, My account, My team (233)), and utility buttons (Sample apps, Help, More, Save). The left sidebar lists various app management options, with 'Automation' highlighted. The main workspace is titled 'Processes' and shows a process named 'Confirm appointment' with the step 'Is confirmation required?'. The process flow starts with an 'Entity' of 'appointments', followed by a decision diamond 'Is confirmation required?'. A 'YES' path leads to a task 'Notify ops to confirm appointment', which then leads to an 'END' node. The right sidebar shows the 'Settings' for the email content, including fields for 'Email Subject' (with a default value 'Appointment confirmation request'), 'Email Body', 'CC', and 'BCC', each with an 'Add' button.



10. Click **Save** to save your changes.
11. Add a wait step to pause execution in the process until the condition (Ops have appointment confirmation) is satisfied.
 - a. Click the **+** icon below **Notify ops to confirm appointment** to create a new step.
 - b. Type `Wait for appointment confirmation` into **Step Name** and select **Create a step called "Wait for appointment confirmation."**
 - c. In the **Settings** pane, select **Wait for a condition** under **Do this**.
 - d. Type `UPPER([Confirmation Status]) = "CONFIRMED"` in **Wait until this condition is true** as follows:

AppSheet BUSINESS

My apps My account My team (233) Sample apps Help More Save

Sample Automation App

Deployed Share App

Info Data UX Behavior Automation PREVIEW Security Intelligence Users Manage Learning Center

Bots Events Tasks **Processes**

Entity: appointments

What type of entity does this process apply to?

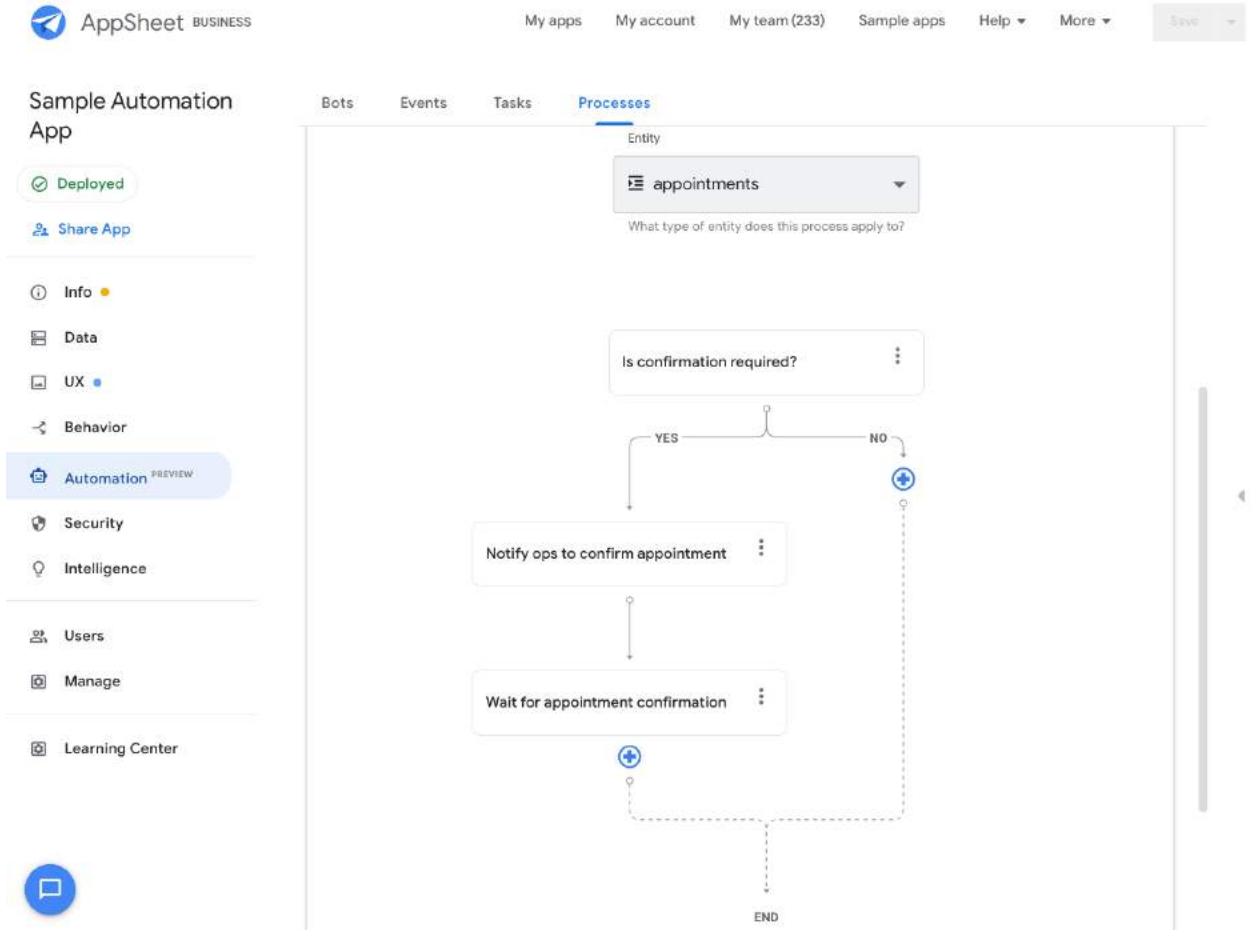
SUGGESTIONS

- Create a step called "Wait for appointment confirmation"
- Set Appointment Date to today
- Check if Confirmation Timestamp equals now
- Check if Appointment Date is from the last 2 days
- Check if Confirmation Status is not blank

Step name: Wait for appointment confirmation

The screenshot displays the AppSheet Business interface for configuring a process. The top navigation bar includes 'My apps', 'My account', 'My team (233)', 'Sample apps', 'Help', 'More', and a 'Save' button. The left sidebar lists various app components: 'Sample Automation App', 'Deployed', 'Share App', 'Info', 'Data', 'UX', 'Behavior', 'Automation (PREVIEW)', 'Security', 'Intelligence', 'Users', 'Manage', and 'Learning Center'. The main workspace is titled 'Processes' and shows a flowchart for the 'appointments' entity. The process starts with a decision step 'Is confirmation required?'. If 'YES', it proceeds to 'Notify ops to confirm appointment', followed by 'Wait for appointment confirmation', and finally 'END'. A 'Settings' panel on the right is open, showing the step name 'Wait for appointment confirmation', the 'Do this' action 'Wait for a condition', and the condition '= UPPER([Confirmation Status]) = "CONFIRM"'. The 'Settings' panel also includes a 'Return values' section and 'Delete' and 'Close' buttons at the bottom.

12. Click **Save** and then **Close**. Your process configuration should look like this:



13. Finish the “No” branch of the process.
 - a. Click + to create a new step.
 - b. Type `Update confirmation status to "not-required"` as the **Step Name**.
 - c. In the **Settings** pane, configure the fields as follows:

Field name	Value
Do this	Run a task
Task	Create new task
Task category	Change data
Task name	Update confirmation status
Table name	appointments
Data change action name	Update confirmation status to

	"not-required"
--	----------------

The screenshot displays the AppSheet Business interface for configuring an automation process. The main workspace shows a process flow for the 'appointments' entity. The flow starts with a decision step 'Is confirmation required?'. If 'YES', it proceeds to 'Notify ops to confirm appointment', followed by 'Wait for appointment confirmation', and then 'Return confirmation status'. If 'NO', it proceeds to 'Update confirmation status to "not-required"'. Both paths then merge and lead to 'Return confirmation status'. The right-hand 'Settings' panel is open, showing the configuration for the 'Update confirmation status to "not-required"' task. The 'Task' dropdown is set to 'Update confirmation status', and the 'Task category' is 'Change data'. The 'Task name' field is empty, and there are 'Delete' and 'Close' buttons at the bottom of the settings panel.

AppSheet BUSINESS

My apps My account My team (234) Sample apps Help More Save

Sample Automation App

Deployed Share App

Info Data UX Behavior Automation (PREVIEW) Security Intelligence Users Manage Learning Center

Bots Events Tasks Processes

appointments

What type of entity does this process apply to?

Is confirmation required?

YES NO

Notify ops to confirm appointment

Update confirmation status to "not-required"

Wait for appointment confirmation

Return confirmation status

Settings

Step name

Update confirmation status to "not-required"

Do this

Run a task Branch on a condition Wait for a condition Call a process

Return values

Task

What task do you want to run?

Update confirmation status

Task category

Category of task

Send an email Send a notification Send an SMS Change data

HTTP

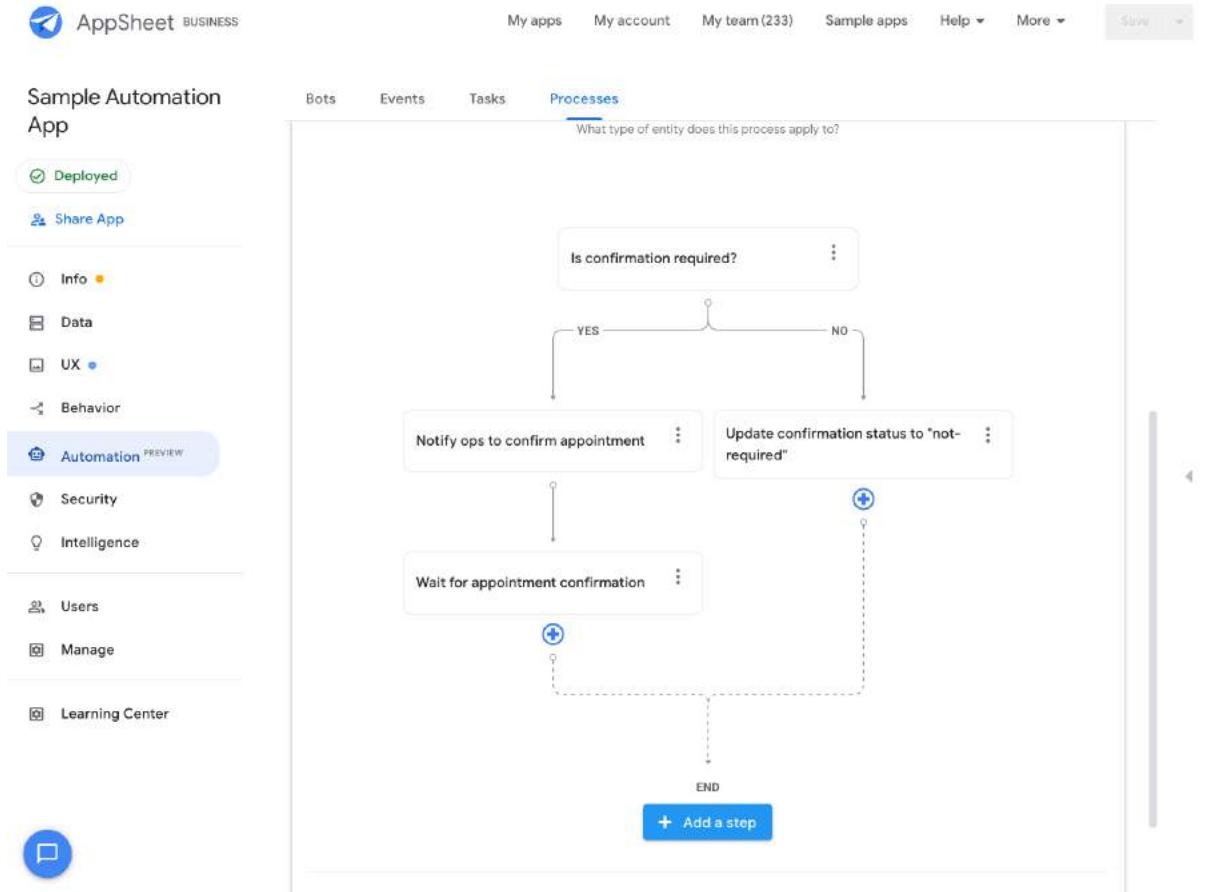
Call a webhook Create a new file

Task name

Delete Close

The screenshot displays the AppSheet interface for configuring an automation process. On the left, a sidebar shows navigation options like 'Info', 'Data', 'UX', 'Behavior', 'Automation', 'Security', 'Intelligence', 'Users', 'Manage', and 'Learning Center'. The main workspace is titled 'Processes' and shows a flowchart for the 'appointments' table. The flow starts with a decision box 'Is confirmation required?'. If 'YES', it goes to 'Notify ops to confirm appointment', then 'Wait for appointment confirmation', and finally 'Return confirmation status'. If 'NO', it goes to 'Update confirmation status to "not-required"'. The 'Update confirmation status to "not-required"' box is highlighted in blue. On the right, the 'Settings' panel is open, showing configuration for this task: 'Task name' is 'Update confirmation status', 'Table name' is 'appointments', 'Data Change Action Name' is 'Update confirmation status to "not-required"', and 'Action type' is 'Set row values'. The 'Action name' is also 'Update confirmation status to "not-required"', and the column 'Confirmation Status' is set to the value 'not-'.

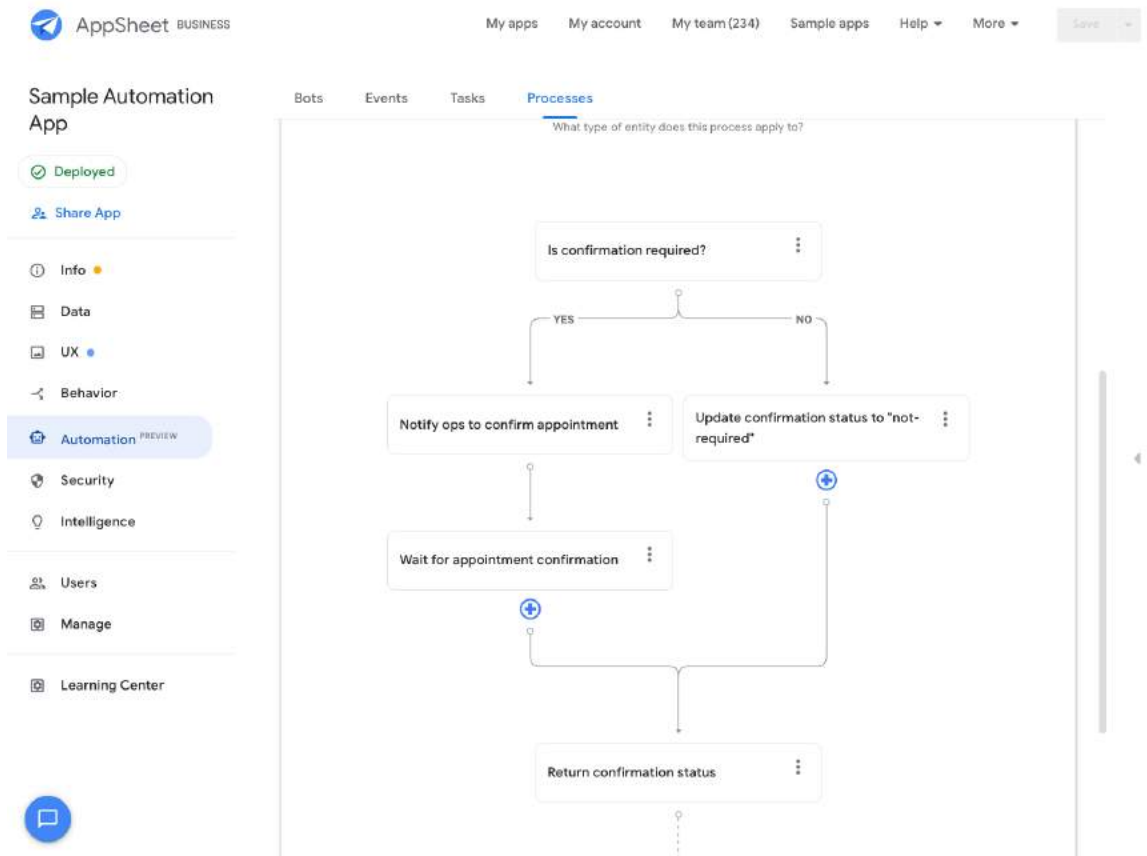
- d. Click **Save** and **Close** to save your changes so far. At this point your process should look like this:



14. Add a final step to return the **confirmation status** value used by the main process to make further decisions.
 - a. Click **+ Add a step** to create a new step.
 - b. Set **Step name** to **Return confirmation status**.
 - c. In the **Settings pane**, under **Do this**, select **Return values**.
 - d. Type `ConfirmationStatus = [Confirmation Status]` in **Return these values**.

The screenshot displays the AppSheet Business interface for configuring an automation process. On the left, a sidebar lists various app management options: Deployed, Share App, Info, Data, UX, Behavior, Automation (highlighted), Security, Intelligence, Users, Manage, and Learning Center. The main workspace is titled 'Processes' and shows a flowchart for the 'appointments' entity. The process starts with a decision 'Is confirmation required?'. If 'YES', it proceeds to 'Notify ops to confirm appointment', then 'Wait for appointment confirmation', and finally 'Return confirmation status'. If 'NO', it goes to 'Update confirmation status to "not-required"', then 'Return values', and finally 'Return confirmation status'. The 'Return values' step is configured in the 'Settings' panel on the right, where the 'Return these values' section shows 'ConfirmationStatu' with the value '[Confirma'.

e. Click **Save** and then **Close**. Your process should now look like this:



Creation of the Appointment Confirmation Process is complete!

Creating the bot

Follow the steps below to create a new bot:

1. From the AppSheet UI, navigate to the **Bots** tab and click **New Bot**.
2. Type `Conference leads appointment generator` into the text box and suggested actions appear, as shown in the image below.
3. Select **Create empty bot named 'Conference leads appointment generator'** to create the bot.

The screenshot shows the AppSheet Business interface. On the left is a navigation sidebar with the 'Automation' menu item highlighted in blue. The main content area is titled 'Sample Automation App' and has tabs for 'Bots', 'Events', 'Tasks', and 'Processes'. Below the tabs, it says 'Bots run processes when an event happens'. There is a 'New Bot' button with a plus sign. Below that is a search bar labeled 'Search bots'. The list of bots includes 'Contacts Sync' (with 'Monitor', 'Copy', 'Delete', and 'Disable' buttons) and 'Conference leads appointment generator' (with a 'Bot name' field and a 'Create empty bot named 'Conference leads appointment generator'' description).

Configuring the event

Once the bot is created, you can define the event for the bot. In this example, the event trigger is the addition of new leads to the Leads Google Sheet.

To create an event:

1. From the AppSheet UI, select the bot you created in the previous section.
2. Click **Choose an event**.
3. Type `leads` and suggested events appear.
4. Select **A new leads record is created** from the suggestions.

The screenshot displays the AppSheet interface for configuring an automation bot. On the left, a sidebar lists various app categories: Info, Data, UX, Behavior, Automation (highlighted with a 'PREVIEW' badge), Security, Intelligence, Users, Manage, and Learning Center. The main area is titled 'Sample Automation App' and shows a 'New Bot' button. Below this is a search bar for bots and a list of existing bots, including 'Contacts Sync' and 'Conference leads app'. The 'Conference leads app' is selected, and its configuration is shown. Under the 'WHEN THIS EVENT OCCURS' section, a dropdown menu is open, displaying a list of event suggestions: 'Create leads event', 'Contact LeadSource is changed', 'leads record is updated', 'leads Email is changed', 'A new leads record is created', 'leads Date is changed to today', 'leads First Name is changed', and 'leads Phone is changed'. The 'leads' event is currently selected in the dropdown. Below the event selection, there is a section for 'RUN THIS PROCESS' with an 'Add a step' button. At the bottom, there is a 'Documentation' link.

The screenshot shows the AppSheet interface for a 'Sample Automation App'. The top navigation bar includes 'My apps', 'My account', 'My team (233)', 'Sample apps', 'Help', 'More', and a 'Save' button. The left sidebar contains navigation options: 'Info', 'Data', 'UX', 'Behavior', 'Automation (PREVIEW)', 'Security', 'Intelligence', 'Users', 'Manage', and 'Learning Center'. The main content area is divided into 'Bots', 'Events', 'Tasks', and 'Processes'. Under 'Bots', there is a 'New Bot' button and a search bar. Two bots are listed: 'Contacts Sync' (A new Contact record is created triggers New process) and 'Conference leads appointment generator' (Incomplete Bot). The 'Conference leads appointment generator' bot is expanded, showing its configuration. Under 'WHEN THIS EVENT OCCURS', there is one event: 'A new leads record is created' (leads). Under 'RUN THIS PROCESS', there is a '+ Add a step' button. A 'Documentation' section is visible at the bottom of the bot configuration panel.

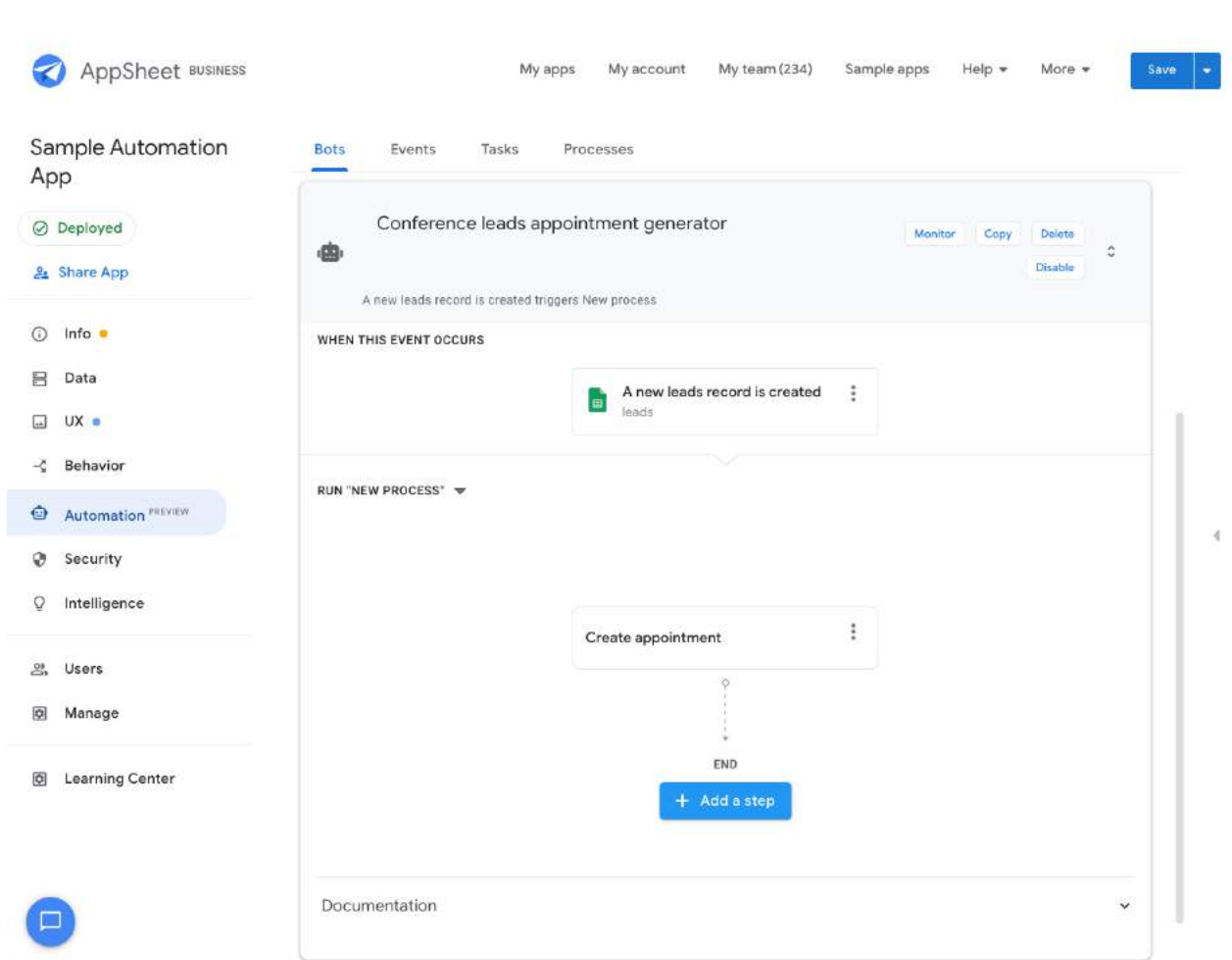
5. Click on the event and edit the configuration from the right hand panel as follows:

6. Click **Close** and then **Save**.

Configuring the main process

Configure the process your event should trigger:

1. In the **Run 'New Process'** section of the editor, click **+Add a step**.
2. Type `Create appointment` into the text box and hit **Enter** to automatically create an empty step. This implicitly creates a process.



3. In the **Settings** pane, configure the fields as follows:

Field	Value
Do this	Call a process
Process name	Confirm appointment
Process inputs	<ul style="list-style-type: none"> • Full name: CONCATENATE([First Name], " ", [Last Name]) • Email: [Email] • Appointment date: [DATE]+[FollowupAfterDays] • Confirmation required: "SWITCH(UPPER([Profile]), "HIGH", "Y", "N")"

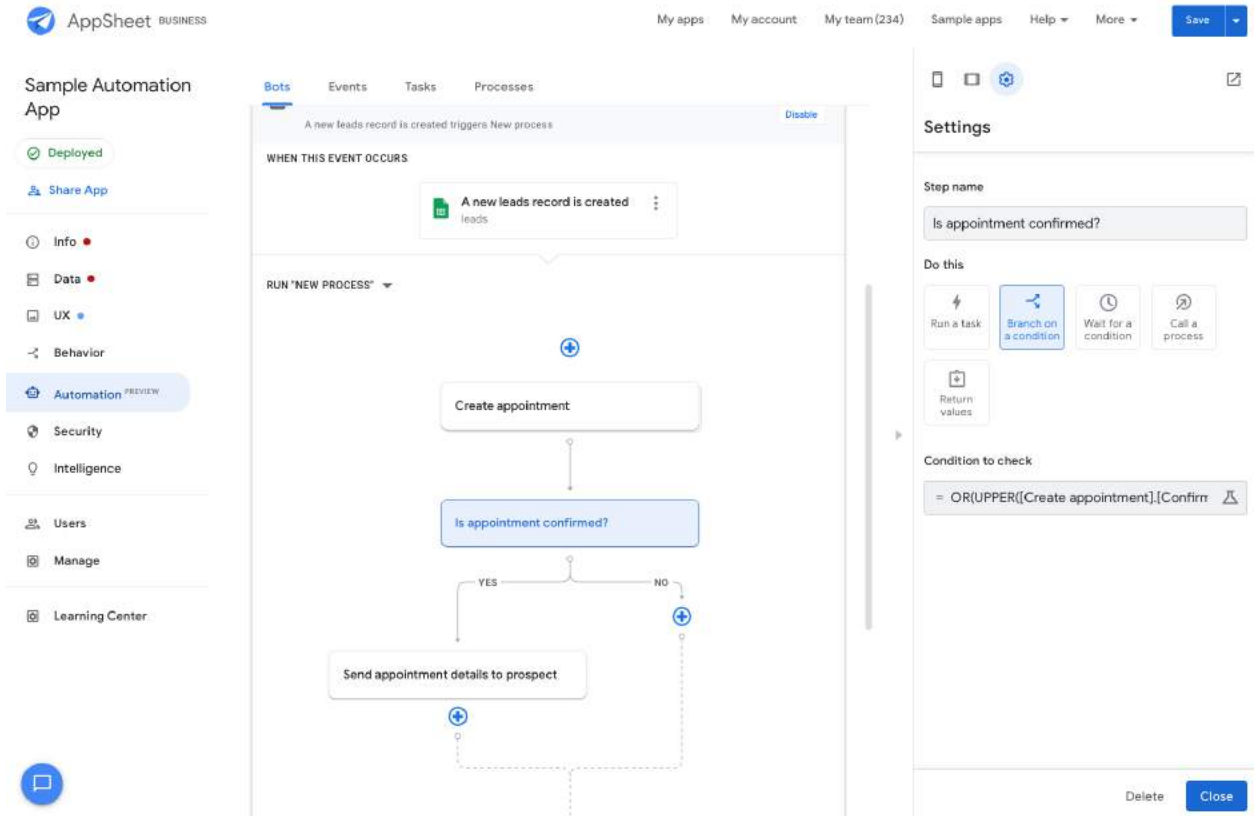
Your step config should look like this:

The screenshot displays the AppSheet interface for configuring an automation process. On the left, a sidebar shows navigation options: Info, Data, UX, Behavior, Automation (PREVIEW), Security, Intelligence, Users, Manage, and Learning Center. The main workspace is titled 'Conference leads appointment generator' and shows a flow starting with the event 'A new leads record is created' (leads) under the heading 'WHEN THIS EVENT OCCURS'. This triggers the 'RUN 'NEW PROCESS'' section, which contains a 'Create appointment' task. Below the task is an 'END' label and a '+ Add a step' button. On the right, the 'Settings' panel is open, showing options to 'Do this' (Run a task, Branch on a condition, Wait for a condition, Call a process, Return values). The 'Process name' is set to 'Confirm appointment'. Under 'Process inputs', the 'Full Name' is configured as `= CONCATENATE([First Name], " ", [Last Name])`, 'Email' is `= [Email]`, and 'Appointment Date' is `= [Date] + [FollowupAfterDays]`. A 'Confirmation Required' checkbox is visible at the bottom of the settings panel.

4. Click **Save** and **Done** to save your changes so far. At this point your process should look like this:

The screenshot displays the AppSheet interface for a 'Sample Automation App'. The main content area shows a process titled 'Conference leads appointment generator'. The process is triggered by the event 'A new leads record is created' (leads). The process flow includes a step 'Create appointment' followed by an 'END' step. A '+ Add a step' button is visible below the 'END' step. The interface includes a sidebar with navigation options: Info, Data, UX, Behavior, Automation (PREVIEW), Security, Intelligence, Users, Manage, and Learning Center. The top navigation bar includes 'My apps', 'My account', 'My team (234)', 'Sample apps', 'Help', and 'More'. The 'Automation' section is currently selected in the sidebar.

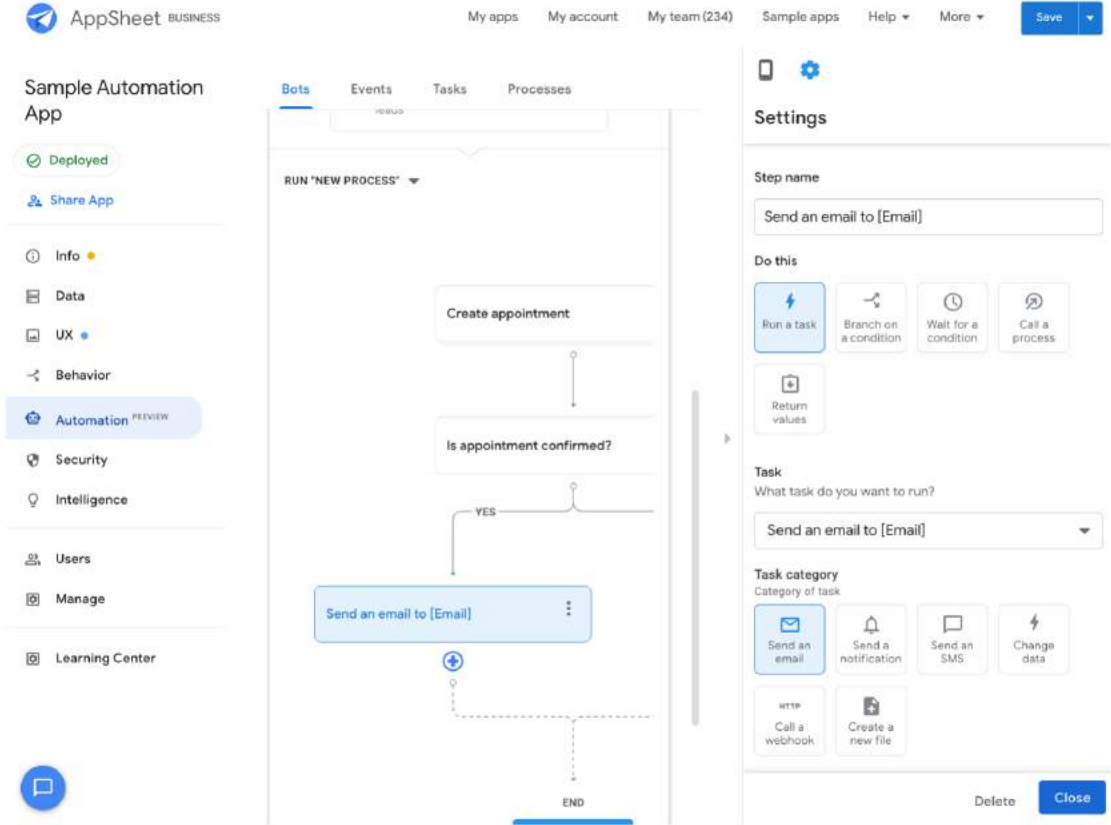
5. Add a branch step to determine if an appointment notification needs to be sent out.
 - a. Click **+ Add a step**.
 - b. In **Step Name**, type `Is appointment confirmed?` and create a step with that name.
 - c. In the **Settings** pane, under **Do this**, select **Branch on a condition**.
 - d. Under **Condition to check**, type `OR(UPPER([Create appointment].[ConfirmationStatus]) = "CONFIRMED", UPPER([Create appointment].[ConfirmationStatus]) = "NOT-REQUIRED")`. The step configuration should look like this:



6. Add one final step to the process.
 - a. Click the + icon in the **Yes** branch.
 - b. In **Step Name**, type `email` and select the suggested step **Send an email to [Email]**.

Note: The platform detects your intent based on the text you enter, correlates that the Leads table has an email address column type and pre-configures the step/task for you.

At this point your step should look like this.



c. Configure the remaining fields as follows:

Field	Value
Table name	appointments
To	[Email]
Email subject	Appointment confirmation request
Email body	Dear <<[Full Name]>> Your appointment has been scheduled for <<[Appointment Date]>>.

Leave all other default values.

d. Click **Save** and **Done** to save your changes.

7. Your completed bot should look like this:

The screenshot displays the AppSheet interface for a bot named "Conference leads appointment generator". The bot is in a "Preview" state. The trigger is "A new leads record is created" from the "leads" sheet. The process flow is as follows:

- WHEN THIS EVENT OCCURS:** A new leads record is created (leads).
- RUN 'NEW PROCESS':**
 - Create appointment
 - Is appointment confirmed? (Decision point)
 - YES:** Send an email to [Email]
 - NO:** (Loop back to the start of the process)

The interface includes a left sidebar with navigation options like Info, Data, UX, Behavior, Automation (Preview), Security, Intelligence, Users, Manage, and Learning Center. The top navigation bar shows "My apps", "My account", "My team (234)", "Sample apps", "Help", and "More".

Testing your automation

Follow these steps to test your automation:

1. Make sure your bot is **Enabled**.
2. Go to your "Leads" Sheet and enter an entire leads record into a new row.

Tip: Create a temporary Sheet and add your row of data there, copy it and then paste it into the "Leads" Sheet)

Note: Automation only supports adding a full record at this point. Do not add a partial

lead record.

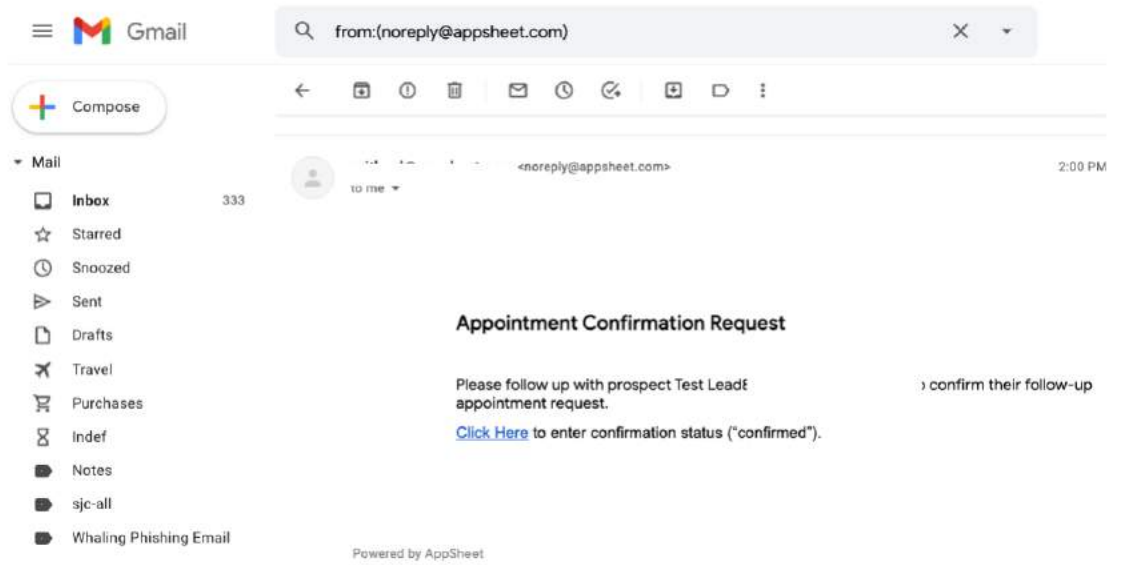
	A	B	C	D	E	F	G	H	I
1	First Name	Last Name	Email	Phone	Company	Date	FollowupRequested	FollowupAfterDays	Profile
2	Test	Lead1		3148889292	Acme Inc	11/1/2020	Y		5 High
3	Test	Lead2		4082224455	Acme Inc	11/1/2020	Y		6 Medium
4	Test	Lead3		9256667788	Acme Inc	11/10/2020	Y		7 Low
5	Test	Lead6		6691112222	Acme Inc	1/6/2021	Y		5 Low
6	Test	Lead7		9256667788	Acme Inc	1/20/2021	Y		5 High
7	Test	Lead8		9256667788	Acme Inc	1/21/2021	Y		7 High
8									

3. Within a few seconds, an appointment record is added to your Appointments Sheet.

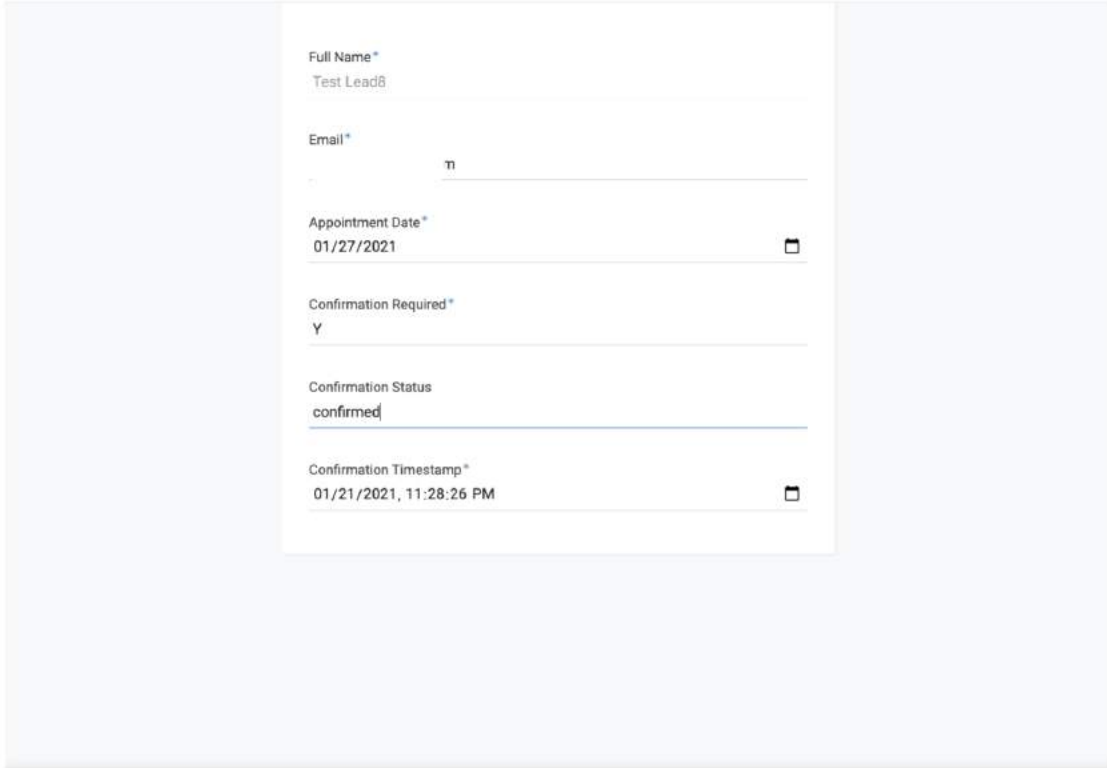
	A	B	C	D	E	F
1	Full Name	Email	Appointment Date	Confirmation Required	Confirmation Status	Confirmation Timestamp
2	Test Lead1	te	11/6/2020	Y	confirmed	1/20/2021 12:34:56
3	Test Lead2	te	11/7/2020	N	not-required	1/19/2021 14:28:43
4	Test Lead3	te	11/17/2020	N	not-required	1/20/2021 18:35:27
5	Test Lead8	te	1/28/2021	Y		1/22/2021 0:07:23
6						

Based upon the process created in previous steps, a new appointment record is only created when the value of the “FollowupRequested” column is “Y” in the Leads Sheet. The “Full Name” field in the Appointment Sheet is a concatenation of the “First Name” and “Last Name” values from the Leads Sheet. The “Appointment Date” in the Appointments Sheet should reflect the date value calculated from the Expression Assistant formula.

4. Because this is a “High” Profile lead the ops team should received an email at the email address in the Sheet



5. Click **Click Here**.
6. When the form opens, on the form, click the **pencil** icon.
7. In **Confirmation Status**, type `confirmed`.
8. Click **Save**.



A screenshot of a mobile application form. At the top left, there is a back arrow icon. The form contains several fields: 'Full Name*' with the value 'Test Lead8'; 'Email*' with a partially visible value ending in '@m'; 'Appointment Date*' with the value '01/27/2021' and a calendar icon; 'Confirmation Required*' with the value 'Y'; 'Confirmation Status' with the value 'confirmed'; and 'Confirmation Timestamp*' with the value '01/21/2021, 11:28:26 PM' and a calendar icon. At the bottom of the form, there are two buttons: 'Cancel' and 'Save'.

9. Once the record updates, a second (appointment confirmation) email is sent to the email address.

Invoice Extraction Automation

This brief tutorial shows you how to configure an Invoice Extraction Automation workflow. In the Preview release of AppSheet Automation, there are three supported document type extraction models for document automation: invoices, receipts, and W9 documents. Additional supported document type extraction models are planned for subsequent releases.

The tutorial below walks through the steps required to automate extraction of invoice documents. However, the same steps can be used to configure extraction automation for all supported document types.

Creating Invoice Table from Folder Data Source

In the following steps, you can:

- Create a table for storing invoice data extracted from *invoice* document types.
- Implicitly configure a table (not visible to users) representing a Google Drive folder containing the *invoice* documents to be automatically extracted.

Once an invoice document is processed, users can move those documents to another folder. In a future release, users can configure a process to automatically move the processed documents to another folder.

To create a table for document extraction:

1. Prepare a Drive folder with invoices. You can obtain some sample invoices [here](#) and upload them to a Google Drive folder of your choice.
2. In the Data section of the application using the extraction, under the **Table** tab, select **New Table**.
3. If you do not have **Google** as a configured data source, click **+** and connect **Google** as a source.
4. Select **Documents on Google Drive**.
5. Under **Documents**, select **Invoices**.
6. Select the Drive folder where you placed the sample invoices and click **Select**.

The screenshot shows the 'Document Sources' configuration dialog in AppSheet. It is titled 'Document Sources' and includes a brief description: 'AppSheet can extract structured data from unstructured documents (folders that contain invoices, receipts, etc), automating document processing and saving you time.'

The dialog is divided into two main sections: 'Documents' and 'Folder'.

- Documents:** This section prompts the user to 'Choose one of these supported document types and AppSheet will automatically read and process the information for you.' Three buttons are visible: 'Invoices' (selected), 'Receipts', and 'W-9 forms'.
- Folder:** This section prompts the user to 'Connect to files within a Google folder. AppSheet will provide a table with the folder contents and file metadata to use within your application.' A button labeled 'Collection of files' is visible.

Below these sections, there are three main configuration fields:

- Source Invoice Folder:** A text input field containing '/Invoice Files' and a 'Change' button. Below it, the text reads 'Connect and extract invoices from a Google Drive folder'.
- Automatic Invoice Detection:** A checkbox that is checked. Below it, the text reads 'New invoices added to this folder will automatically be read and processed.'
- Table name:** A text input field containing 'Invoice Files'. Below it, the text reads 'Name for this table (user-visible)'.

At the bottom left, there is a 'Learn more' link. At the bottom right, there is a prominent blue 'Create Table' button.

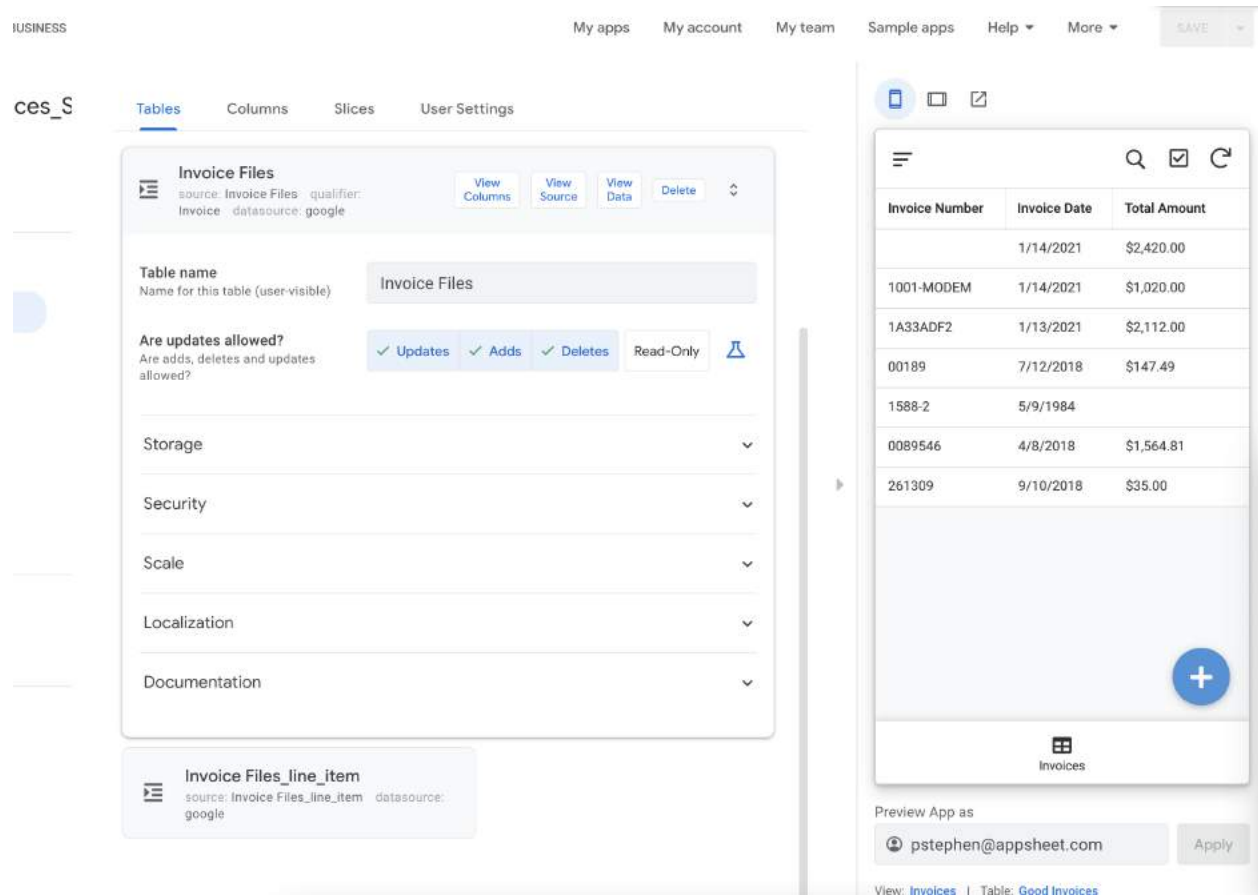
7. Select **Create Table**. This creates two tables:
 - Invoice Files
 - Invoice Files_line_Item

File Processing

Once the table is created, invoice processing begins automatically. The invoices in the folder when the table is created are processed first. If new documents are added to the folder at any point while the table is included as part of the application, the new documents are automatically processed.

Note: Processing may take some time, depending on the number of invoices in the folder. In the Preview release, the UI shows a loading indicator until the processing is completed.

Upon successful completion, the screen looks like this:



There are some limitations to keep in mind while using this capability. These limitations are

likely to change over time as the service matures.

Note: *Limitations for Preview release:*

- *Processing only works for 1,000 documents for a given table. New files are not processed after 1,000 files have been processed.*
- *Documents need to be under 20 MB and 5 pages to be processed. Otherwise the file processing fails.*
- *The only extension types supported in preview are .tiff, .gif, .pdf*

Adding Application Views


As with any table, views can be added to the application using document extraction tables. However, in document-based workflows, App Editors should be careful not to change the underlying table schema.

A few views are automatically created:

- Invoice_Files_Details
- Invoice_Files_Form
- Invoice Files_line_item_Detail
- Invoice Files_line_item_Form
- Invoice Files_line_item_Inline

Ref Views

Invoice Files

 **Invoice Files_Detail**
data: Invoice Files type: detail

 **Invoice Files_Form**
data: Invoice Files type: form

Invoice Files_line_item

 **Invoice Files_line_item_Detail**
data: Invoice Files_line_item type: detail

 **Invoice Files_line_item_Form**
data: Invoice Files_line_item type: form

 **Invoice Files_line_item_Inline**
data: Invoice Files_line_item type: table

Low Confidence Extraction Handling

In the event that an extraction confidence score is below the threshold considered successful, a user should review the results and adjust the values if needed.

In the Preview release, the user performing the manual review can create a new slice identifying the documents that need attention using this expression:

```
AND([NeedsAttention], [StatusCode] != "EXTRACTION_ERROR")
```

With the slice created, the user can build a view to see the processed invoices that need attention. Users can also leverage the capabilities of process automation to detect "adds" inside a filtered table and trigger additional flows.

Failure Handling

In some cases, a complete failure may occur during document extraction. Common causes

include attempts to extract documents that are totally illegible (i.e written in cursive handwriting), written in languages not supported by our AI services, in an incorrect file format, or that are of the incorrect document type.

In the event that the extraction fails, the extraction results are captured by a 'Failed Invoices' slice with the following expression:

```
AND([NeedsAttention], [StatusCode] == "EXTRACTION_ERROR")
```

A view can be created to see which documents failed. Users can also leverage the capabilities of process automation to detect "adds" inside the slice and trigger additional flows.

For more details about status codes and how to use them, see [Document processing states and status codes](#).

Folder Contents as a Table

This brief tutorial shows you how to configure an application that can represent files in a Google Drive folder as a table in AppSheet. This "folder-backed" table can be further utilized by AppSheet with *views* and *slices*. The ability to use folder contents as a table provides a simple mechanism for including files in your application.

The following steps demonstrate how to:

- [Creating an app using the contents of a Drive folder](#)
- [Creating a table from a folder data source](#)
- [Creating a useful view for your app](#)
- [Creating slices to filter data](#)
- [Adding virtual columns](#)
- [Linking files to other data](#)











Creating an App


This [ZIP file](#) contains a folder with a Sheet and images detailing information about inventory in an office setting. To begin, download the ZIP file and extract the contents. Each inventory item is represented by the following pieces of data:

- Name
- Location
- Type
- Model identification number

When you create a new app with this Sheet, you may see an error message about the location base image. To resolve the error, edit the location column and set the base image to "<https://storage.googleapis.com/geodata-appsheet-demos/office%20floorplan.jpg>", as shown below.


Tables Columns Slices User Settings

NAME	TYPE	KEY?	LABEL?
1  _RowNumber	Number	<input checked="" type="checkbox"/>	<input type="checkbox"/>
2  item id	Decimal	<input type="checkbox"/>	<input type="checkbox"/>
3  name	Name	<input type="checkbox"/>	<input checked="" type="checkbox"/>
4  description	Text	<input type="checkbox"/>	<input type="checkbox"/>
5  image	Image	<input type="checkbox"/>	<input checked="" type="checkbox"/>
6  category	Enum	<input type="checkbox"/>	<input type="checkbox"/>
7  location	XY	<input type="checkbox"/>	<input type="checkbox"/>
8  status	Text	<input type="checkbox"/>	<input type="checkbox"/>
9  last date inspected	DateTime	<input type="checkbox"/>	<input type="checkbox"/>
10  last inspected by	Email	<input type="checkbox"/>	<input type="checkbox"/>


assets : location Done 

type: XY

Column name
Column name


Show?
Is this column visible in the app?
You can also provide a 'Show_If'
expression to decide. 


Type
Column data type


Type Details 

Optional Url for KML File

Background image for the XY coordinates

Data Validity 

Auto Compute 

Update Behavior 

Once you have edited the location, the app should appear as follows:



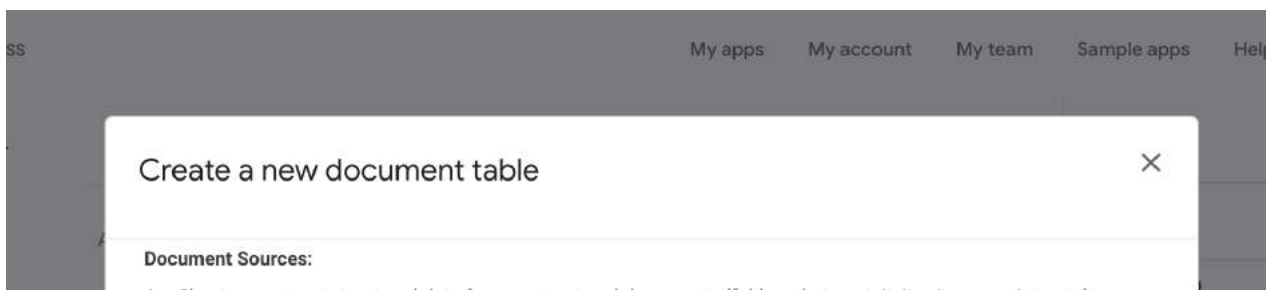
The app should have created the following views:

- *Map* and *assets* UX view
- *assets_Detail* view
- *assets_Form* view

Creating a Table from Folder Data Source

To create a table for document extraction:

1. Prepare a Drive folder with files. For this example, you can obtain some sample files here (content is not important, only the names) and upload them to a Google Drive folder of your choice.
2. In the Data section of the app, select **Table > New Table**.
3. If you do not have **Google** as a configured data source, click **+** and connect **Google** as a source.
4. Select **Documents on Google Drive**.
5. Select **Documents > Collection of files**.
6. Located the Drive folder where you placed the sample invoices and click **Select**. Drive



7. Select **Create Table**. This will create a table named *User manuals*. The table name is, based on the folder name selected.

Once the table is created, a set of metadata available for files in the folder is provided, as shown below:

	NAME	TYPE	KEY?	LABEL?
1	<input type="text" value="_ID"/>	<input type="text" value="Text"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
2	<input type="text" value="Path"/>	<input type="text" value="Text"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
3	<input type="text" value="File"/>	<input type="text" value="File"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	<input type="text" value="CreateTime"/>	<input type="text" value="DateTime"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	<input type="text" value="LastModifiedBy"/>	<input type="text" value="Email"/>	<input type="checkbox"/>	<input type="checkbox"/>

Note: In the release, the table is read-only. You can upload and delete files directly from the app in future releases.

Additionally, eventing support is not available. This means it is not possible to couple events on the folder with automation action. Eventing support is planned for a future release.

Creating a useful view

As with any table, views can be added to apps created with folder-backed tables. However, App Editors should be careful not to change the underlying table schema. Two views are

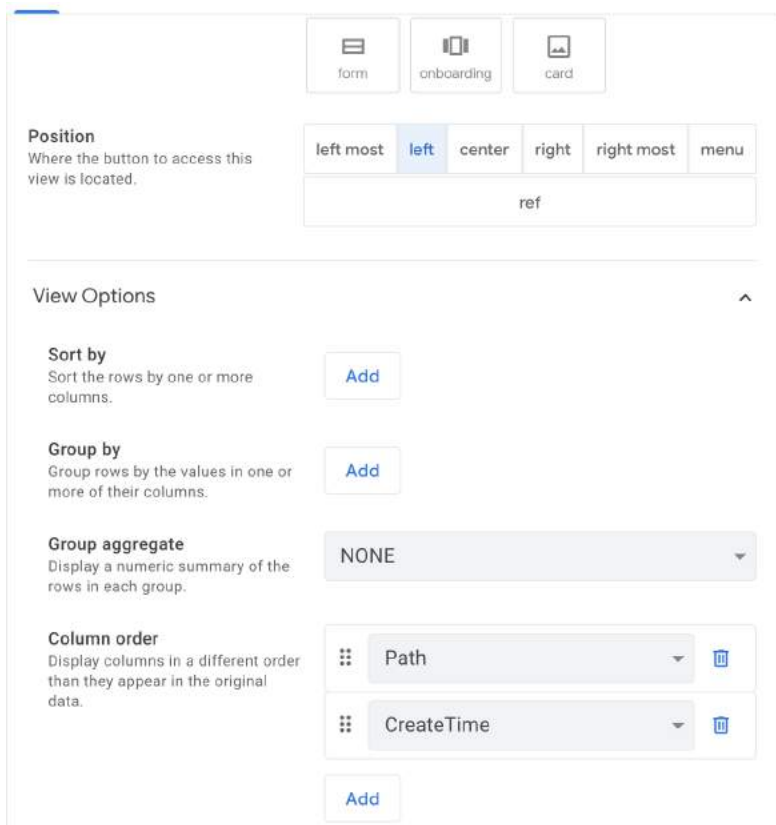
automatically created:

- User_Manuals_Details
- User_Manuals_Form

The two most common views of the table enable users to:

- List files
- Click on the list to open a file

You can create a file viewer by creating a new UX view. For simplicity, you can create the UX view as a table view, using **Path** and **Creation Date** as the two columns, as shown below:



This view displays to app users like this:

Path	CreateTime
MANUAL_Chair36.pdf	2/2/2021 5:34:21 PM
MANUAL_Chair854.pdf	2/2/2021 5:34:22 PM
MANUAL_Couch123.pdf	2/2/2021 5:34:11 PM
MANUAL_Table123.pdf	2/2/2021 5:34:12 PM
MANUAL_Laptop321.pdf	2/2/2021 5:29:02 PM
MANUAL_Laptop123.pdf	2/2/2021 5:07:57 PM
bad_tag.png	2/2/2021 5:34:32 PM

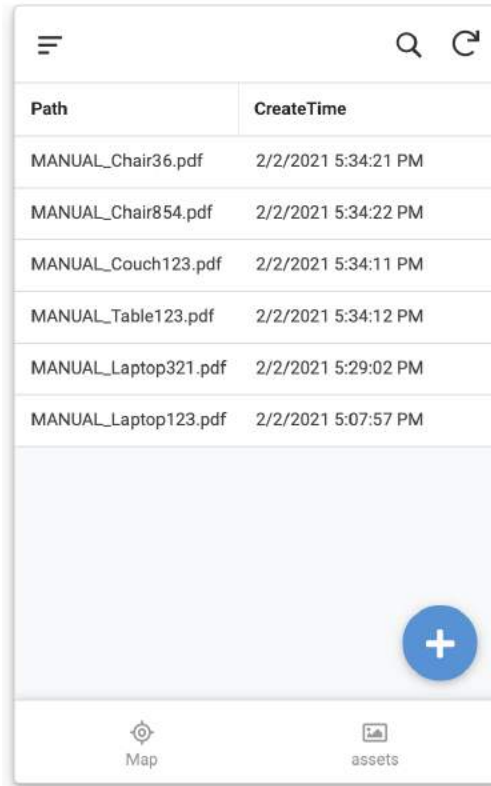
Creating Slices to filter data

You may wish to create data slices for your app. For example, you can create a slice that only shows the `.pdf` files with the prefix `MACHINE`.

To do this, filter for files where the **Path** field contains the prefix `MANUAL_` and the suffix `.pdf`. Use the following query:

```
AND(FIND(".pdf", LOWER([Path])) > 0, FIND("MANUAL",[Path]) > 0)
```

The resulting file list displays the slice containing only the machine files. For example, if the view is changed to target this slice, the `bad_tag.png` file is no longer listed.



Path	CreateTime
MANUAL_Chair36.pdf	2/2/2021 5:34:21 PM
MANUAL_Chair854.pdf	2/2/2021 5:34:22 PM
MANUAL_Couch123.pdf	2/2/2021 5:34:11 PM
MANUAL_Table123.pdf	2/2/2021 5:34:12 PM
MANUAL_Laptop321.pdf	2/2/2021 5:29:02 PM
MANUAL_Laptop123.pdf	2/2/2021 5:07:57 PM

Adding Virtual Columns

In addition to slices, you can add virtual columns to the table. For example, you can create a virtual column that only displays files with the structure `MANUAL_[ID].[extension]`. To create a virtual column to extract the ID content, use the following expression in the **App formula** field:

```
IF(AND(FIND(".pdf", LOWER([Path])) > 0, FIND("MANUAL",[Path]) > 0),
MID([Path], 8, LEN([PATH])-11), "")
```

User Manuals : New Virtual Column (virtual)

type: Text formula: =IF(AND(FIND(".pdf",...))

Delete Done ⌵

Column name

Column name

Manual ID|

App formula

Compute the value for this column instead of allowing user input.

= IF(AND(FIND(".pdf", LOWER([Path])) > 0, FIND("MANUAL",[Path]) > 0)

Show?

Is this column visible in the app? You can also provide a 'Show_If' expression to decide.

⚙️

Type

Column data type

Text ⌵

Type Details

⌵

Maximum length

Minimum length

Data Validity

⌵

Linking Files to other Data

With the slices and virtual column created, you can link the User Manual for each machine by Part ID to the data in the spreadsheet. For example, add a virtual column to the assets table using the configuration below:

assets : New Virtual Column (virtual) Delete Done ↕

type: File formula: =LOOKUP([name], "Use...

Column name
Column name:

App formula
Compute the value for this column instead of allowing user input. 🔗

Show?
Is this column visible in the app? You can also provide a 'Show_If' expression to decide. 🔗

Type
Column data type: ▼

Type Details ^

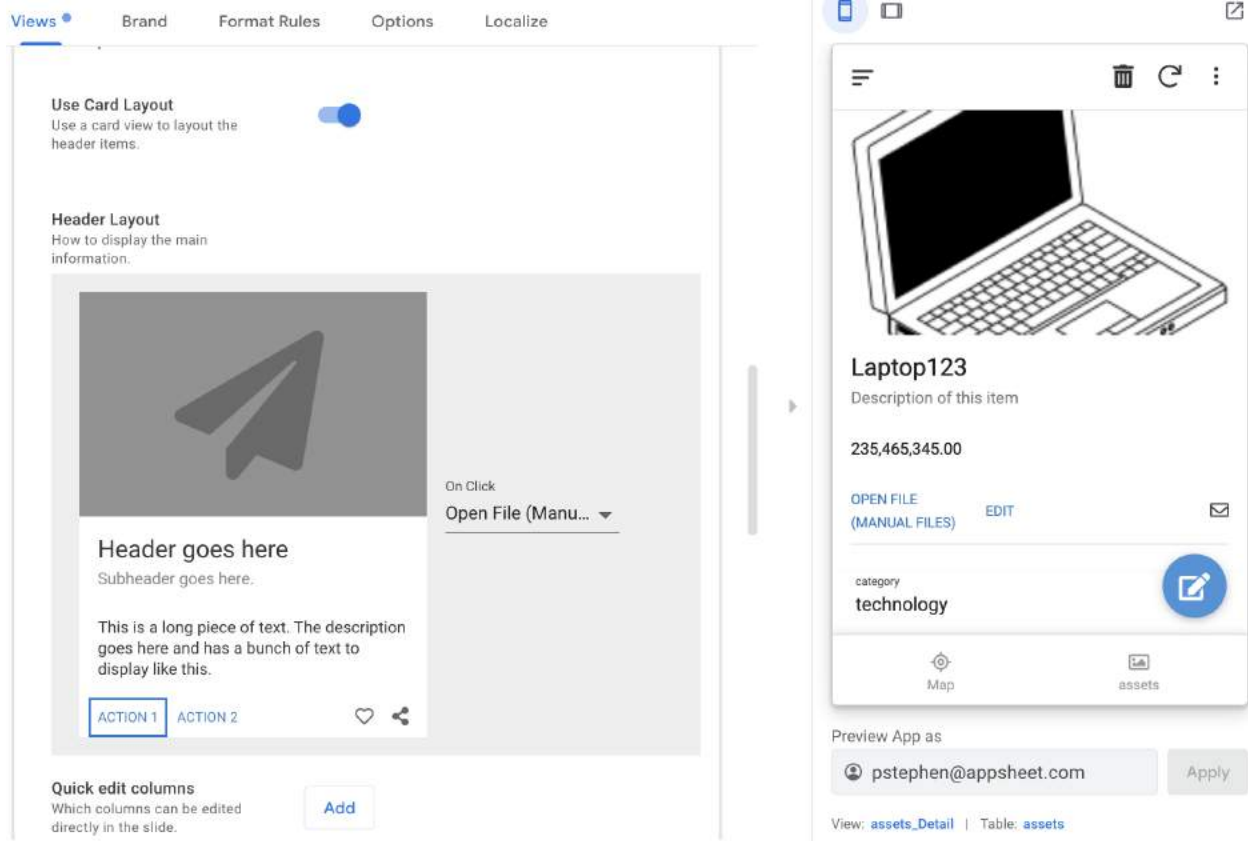
Image/File folder path
Folder path where images or files are saved (only respected by some data sources). Leave blank for default behavior. 🔗

Data Validity ▼

Display ▼

To make this file accessible in the detailed view for each inventory item:

1. Select **Use Card Layout**.
2. Change **ACTION 1** from **Delete** to **Open File (User Manuals)**.

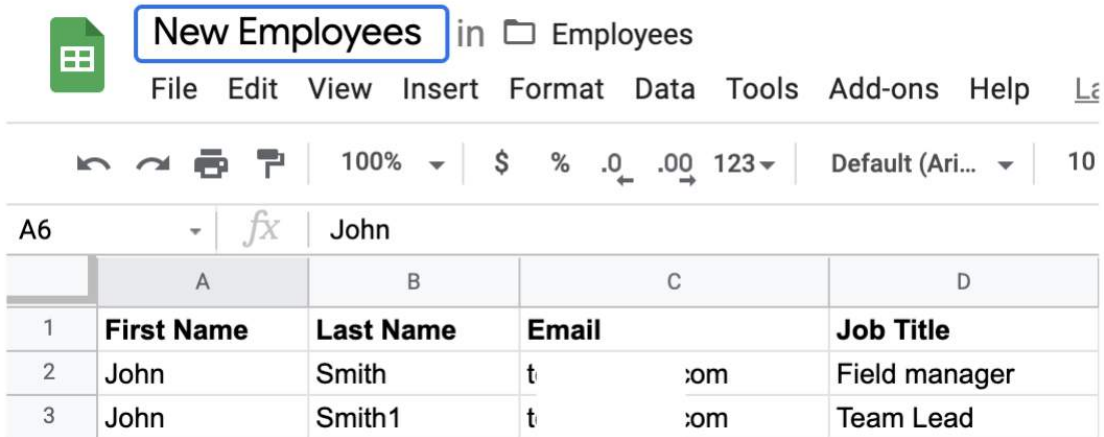


Appendix

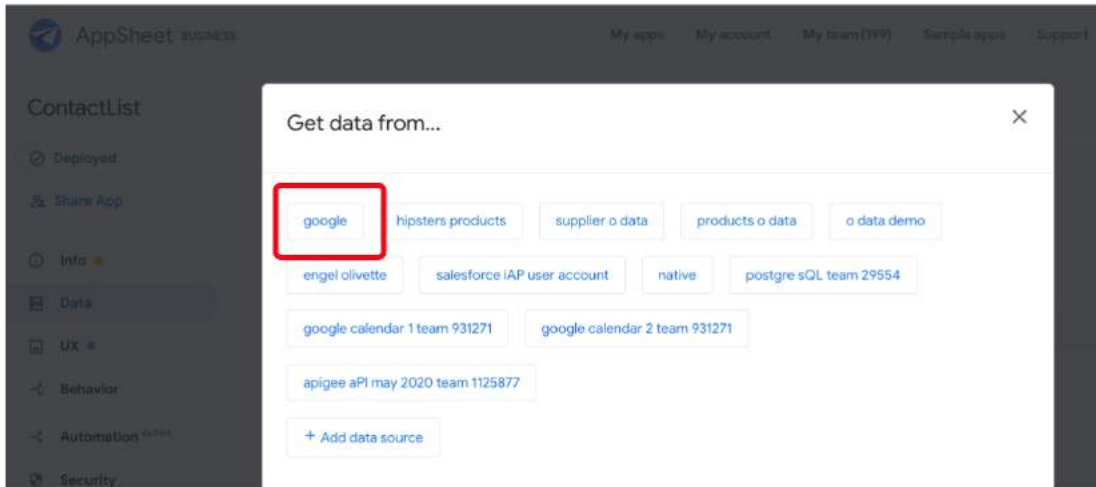
Configuring Employees entity from Google Sheets datasource

To configure your “employees” entity from Google Sheets:

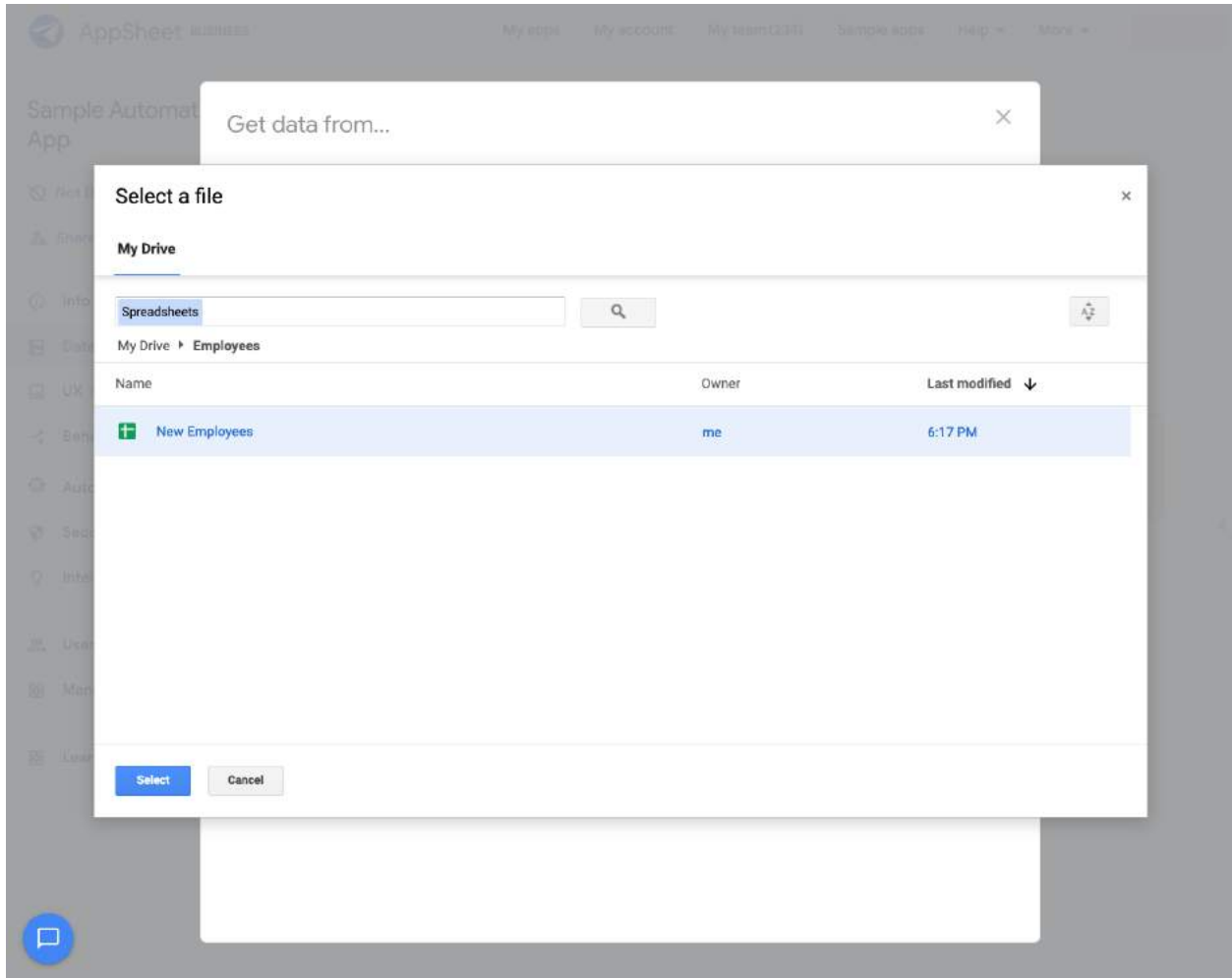
1. Create a new sheet called “New Employees” like this

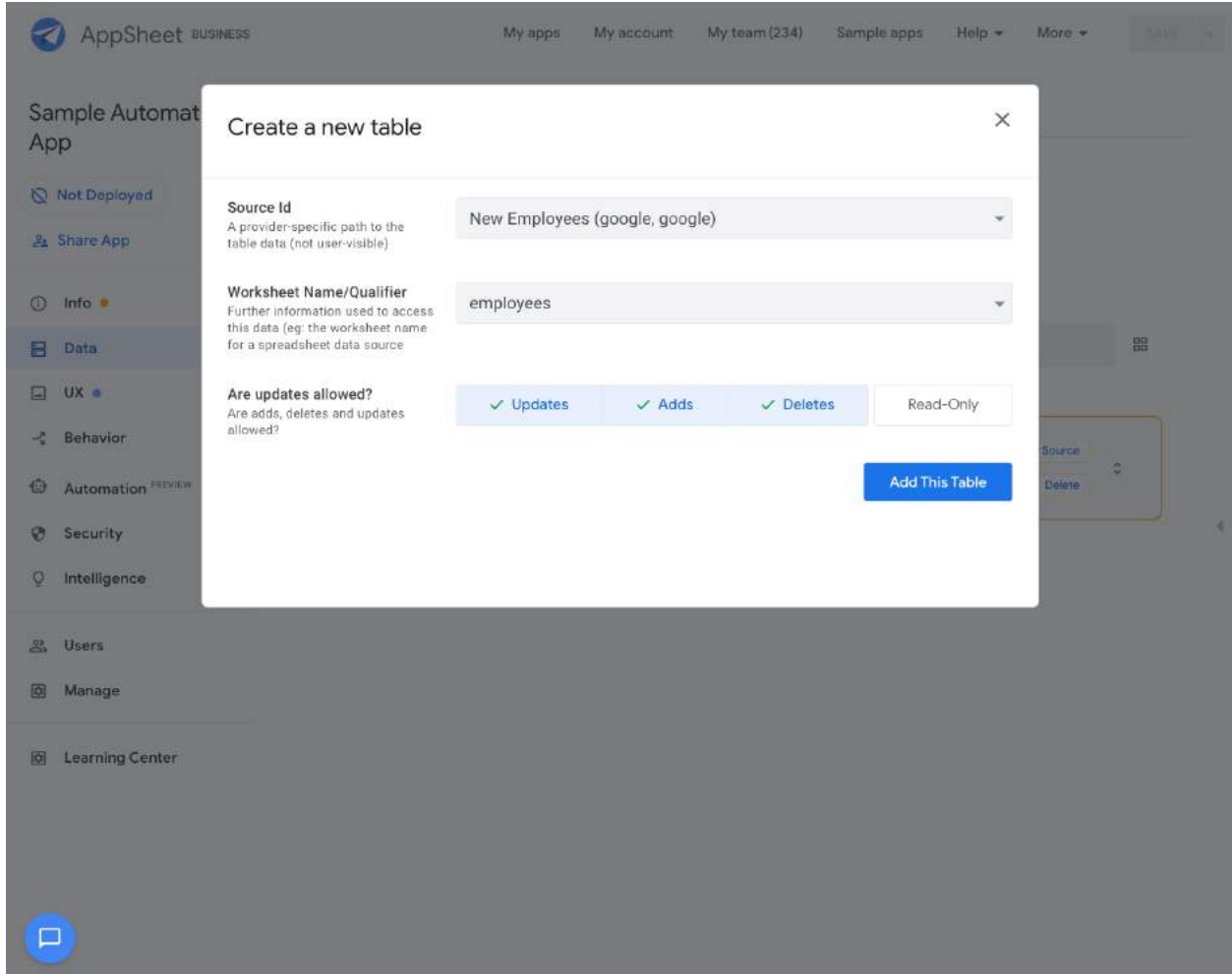


2. From the AppSheet UI, click **Data** in the left navigation bar.
3. Click **New Table**.
4. Select your configured Google Data Source.



5. Use the file browser to select the appropriate **New Employees** Sheet as the data source.





6. Select **employees** as the table for your entity.

Configuring the AppSheet Events add-on for Google Sheets

Follow the steps below to configure the AppSheet Events add-on for Google Sheets:

1. Navigate to:
https://gsuite.google.com/marketplace/app/appsheet_events/592572205846
2. Click **Install** to install the add-on. Follow the authorization steps and authorize.

Note: Make sure you are only logged into one Google account in your browser or that you are using an Incognito tab for this step. This is only required for the API keys authorization process. Once this step is complete, you can use the sheet in a different window with multiple accounts logged in.

Follow the steps below to configure the permissions and authorization required for your Sheet to communicate with AppSheet:

1. Open the Google Sheet you want to use for automation.
2. Click **Add-ons > Appsheet Events** in the main menu.
3. Click **Update App Keys** and authorize if necessary

Note: *If the current Sheet is not automatically selected, and there are no Sheets in the drop down, please ensure you are only logged into Google account or you are using an Incognito tab.*
4. If the selected Sheet is not the one you intend to link, change the Sheet in the drop-down.
5. Click **Add**.
6. Obtain an API Key:
 - a. Log in and open your app in AppSheet UI.
 - b. Click **Manage** in the side navigation.
 - c. Select the **Integrations** tab.
 - d. Click on the **IN: from cloud services to your app** to enable your Sheet to communicate with your app.
 - i. Click the **Enable** toggle to enable the integration, if it is not enabled.
 - ii. Under **Application Access Keys**, check to ensure the access key is not expired.

Note: *If the key is valid, you should see a message like this “This access key will stop working on <YYYY-MM-DD>” containing a future date.*
 - iii. If the key is expired, click on **Create Application Access Key** to generate a new key.
 - iv. Click **Show Access Key** to display the valid key string. For example, “V2-itrnz-sF0Z5-MQxNn-TPrB2-RCaIr-x7qdV-JYct6-QoXiy”
 - v. Copy the access key into a notepad. The key string is used in a later step.
7. Go back to the Sheet and paste the API key in the text box.
8. Click **Add**. The API key should now appear in the list. The key is masked.
9. Your Sheet is now linked to your app for events.

Notes:

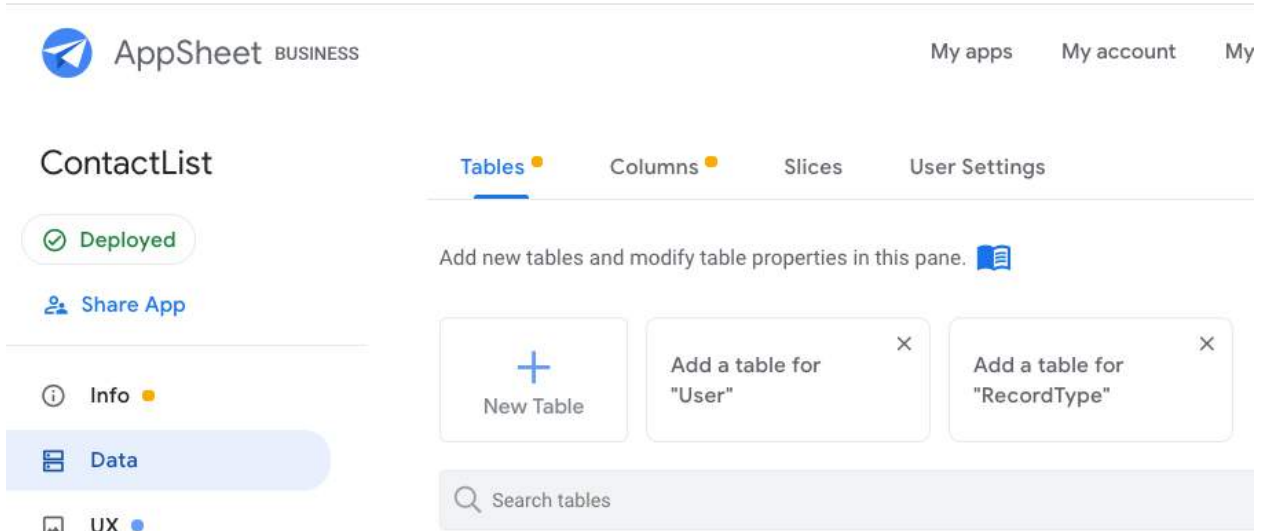
- Adds and Updates are supported. We do not support deletes.
- For your appsheet table, your primary key should be the first column of your google sheet, or it should include the first column of your google sheet (in the case of a composite key).
- There is at least a 20 second delay between events.
- Events do not trigger if the file is open in read-only (view or comment) mode.

The screenshot displays the Google Sheets interface for a spreadsheet named "New Employees". The spreadsheet has columns labeled "First Name", "Last Name", "Email", and "Job Title". The "Update Appsheet Keys" sidebar is open on the right, showing options to connect the AppSheet Automation with the sheet. The sidebar includes a "Select a Sheet:" dropdown menu with "employees" selected, an "Add" button, and a section for "Connected Tables" with a table ID "V2-1MgKY-XXXXX-XXXXX-XXXXX-XXXXX".

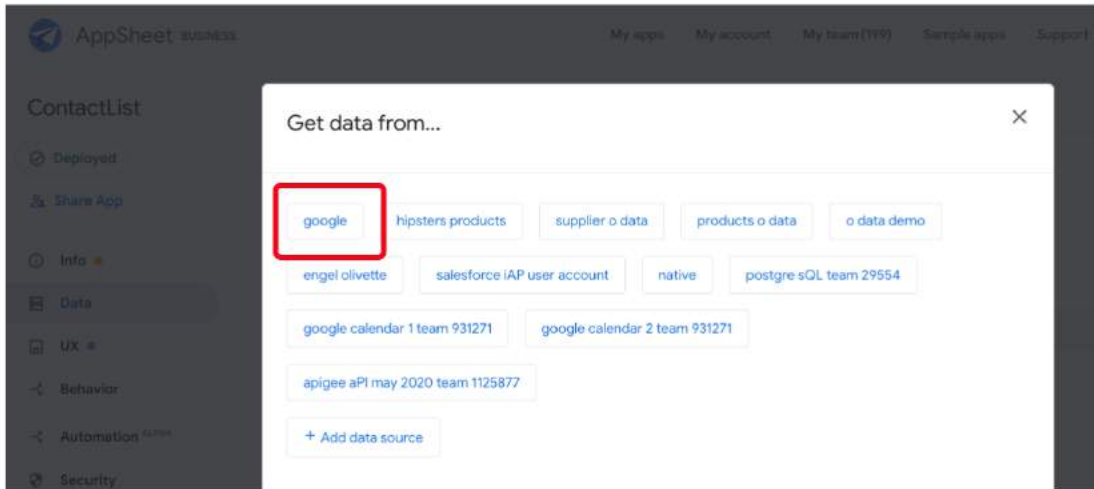
Configuring Company Contacts entity from Google Sheets datasource

To configure your “Company Contacts” entity from Google Sheets:

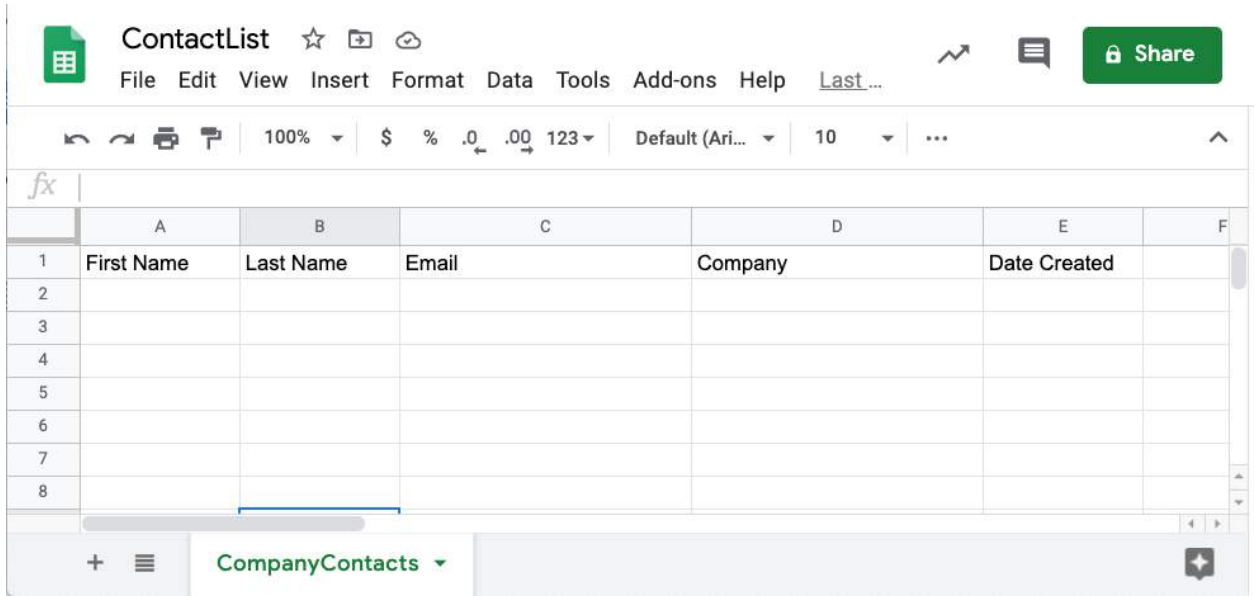
1. From the AppSheet UI, click **Data** in the left navigation bar.
2. Click **New Table**.



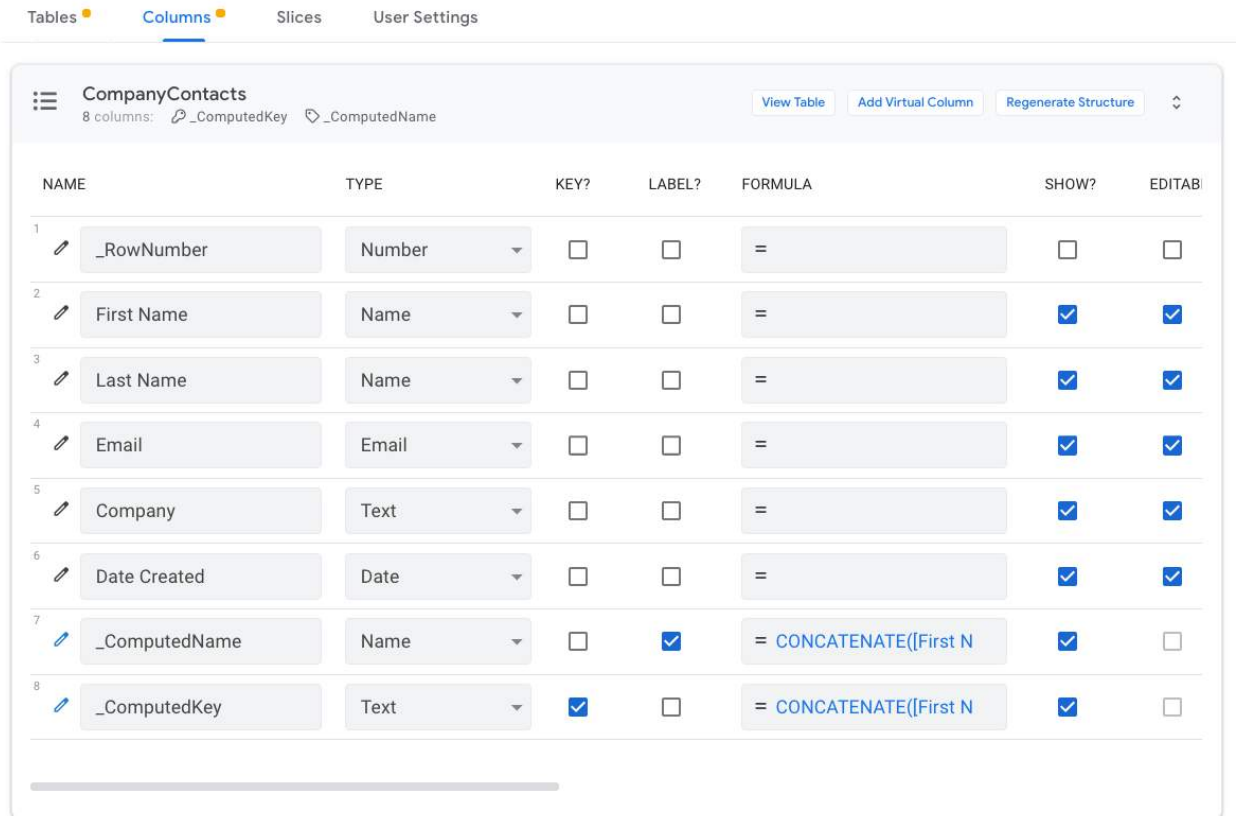
3. Select your configured Google Data Source.



4. Use the file browser to select the appropriate **ContactList** Sheet as the data source.



5. Select **Company Contacts** as the table for your entity.



Configuring Contact entity from Salesforce datasource

Follow these steps to ensure that you have correctly configured a “Contact” entity in

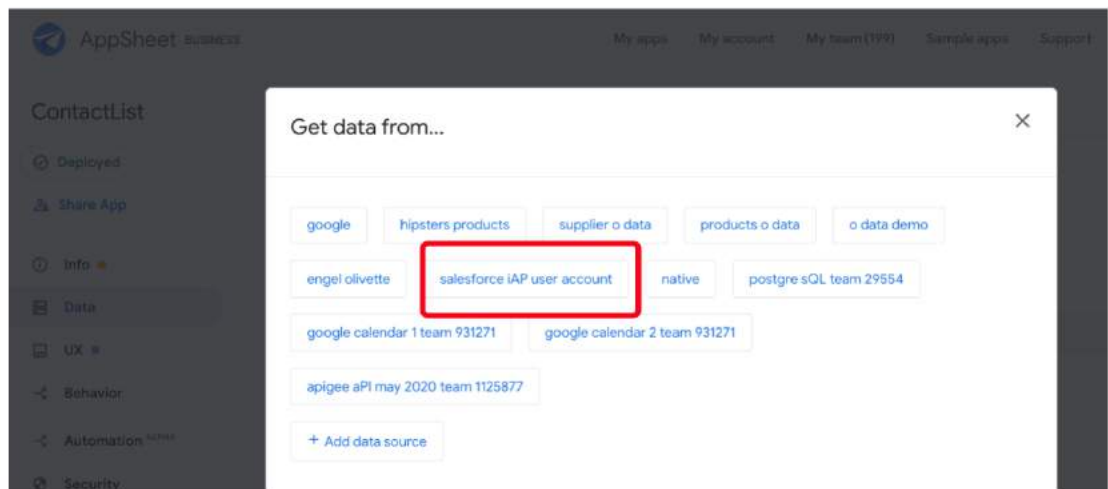
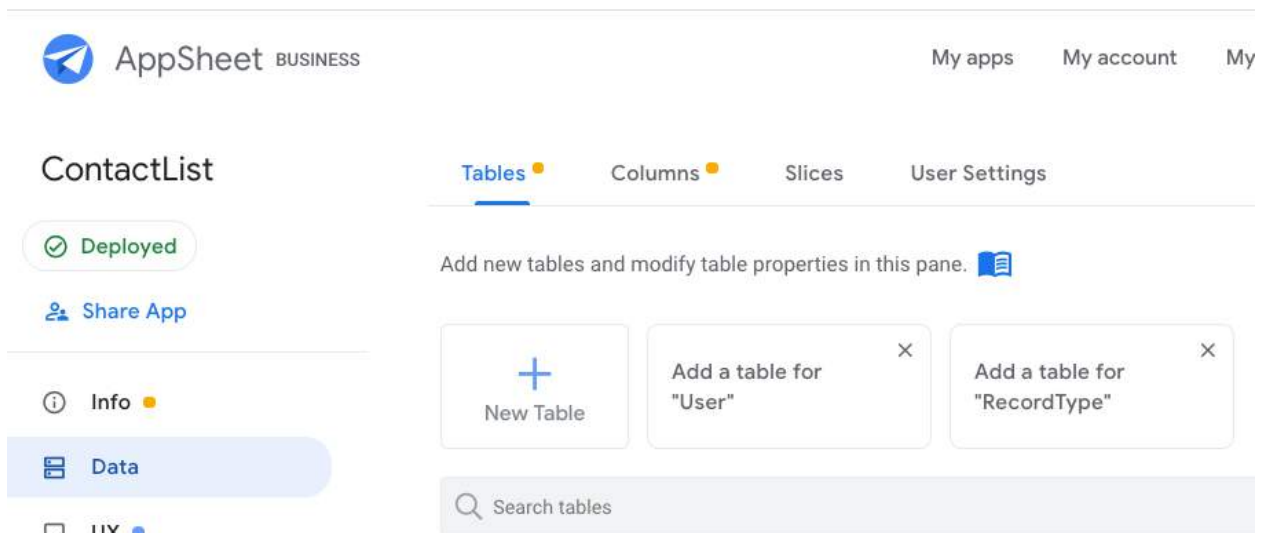
Salesforce.

Before you begin:

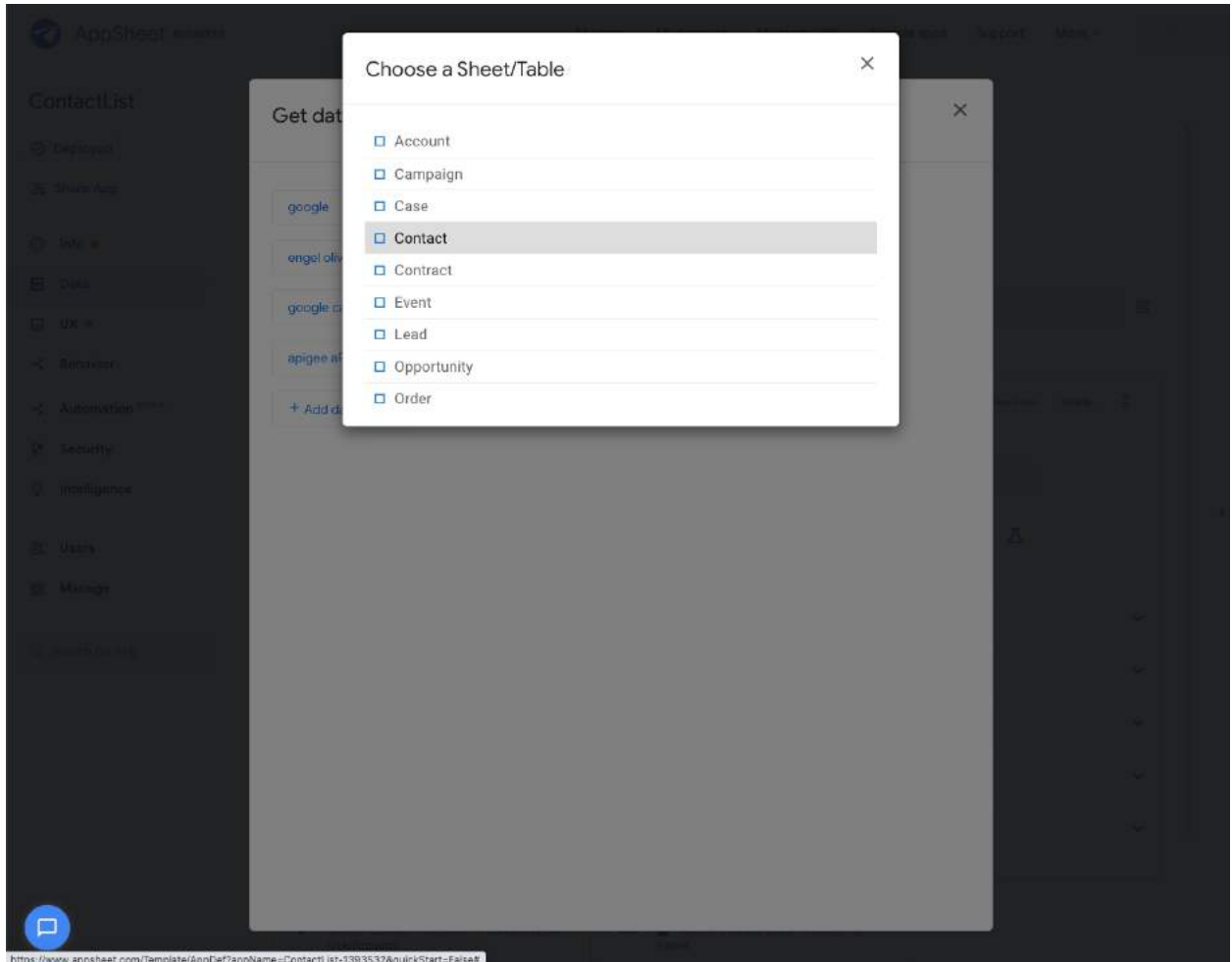
- Confirm that you have access to a Salesforce account.
- Confirm that the appropriate AppSheet package has been deployed in Salesforce. (Refer to this [help article](#) for further detail).

To configure a “Contact” entity in AppSheet using your Salesforce account data:

1. From the AppSheet UI, click **Data** in the left navigation bar.
2. Click **New Table**.
3. Select your configured Salesforce Data Source, as shown in the image below.



4. Select **Contact** as the table for your entity.



5. For **Are updates allowed?** change the selection to **Read-Only**.

The screenshot shows the configuration page for a table named "Contact". At the top, there are navigation buttons: "View Columns", "View Source", "View Data", and "Delete". Below the table name, there is a section for "Are updates allowed?" with options for "Updates", "Adds", "Deletes", and "Read-Only" (which is selected). There is also a lock icon. Below this are several expandable sections: "Storage", "Security", "Scale", "Localization", and "Documentation", each with a downward arrow.

6. Click **View Table**, as shown below:

Contact Warning
 46 columns: [Row ID](#) [Name](#) [View Table](#) [Add Virtual Column](#) [Regenerate Structure](#)

⚠ Column "OwnerId" in Contact_Schema has a reference to an unknown table or slice "User". [More info](#)

⚠ Column "CreatedById" in Contact_Schema has a reference to an unknown table or slice "User". [More info](#)

⚠ Column "LastModifiedById" in Contact_Schema has a reference to an unknown table or slice "User". [More info](#)

	NAME	TYPE	KEY?	LABEL?	FORMULA	SHOW?
1	_RowNumber	Number	<input type="checkbox"/>	<input type="checkbox"/>	=	<input type="checkbox"/> ⌵
2	Row ID	Text	<input checked="" type="checkbox"/>	<input type="checkbox"/>	=	<input type="checkbox"/> ⌵
3	IsDeleted	Yes/No	<input type="checkbox"/>	<input type="checkbox"/>	=	<input type="checkbox"/> ⌵
4	MasterRecordId	Ref	<input type="checkbox"/>	<input type="checkbox"/>	=	<input checked="" type="checkbox"/> ⌵
5	AccountId	Ref	<input type="checkbox"/>	<input type="checkbox"/>	=	<input checked="" type="checkbox"/> ⌵
6	LastName	Text	<input type="checkbox"/>	<input type="checkbox"/>	=	<input checked="" type="checkbox"/> ⌵
7	FirstName	Text	<input type="checkbox"/>	<input type="checkbox"/>	=	<input checked="" type="checkbox"/> ⌵
8	Salutation	Enum	<input type="checkbox"/>	<input type="checkbox"/>	=	<input checked="" type="checkbox"/> ⌵
9						

7. Create a new Action (under **Behavior**) by configuring the fields as follows:

The screenshot displays the AppSheet interface for configuring an automation. The top navigation bar includes 'My apps', 'My account', 'My team (233)', 'Sample apps', 'Help', and 'More', along with a 'Save' button. The left sidebar shows the 'Sample Automation App' with a 'Not Deployed' status and various management options like 'Share App', 'Info', 'Data', 'UX', 'Behavior', 'Automation', 'Security', 'Intelligence', 'Users', 'Manage', and 'Learning Center'. The main panel is titled 'Update company contact' and shows the following configuration:

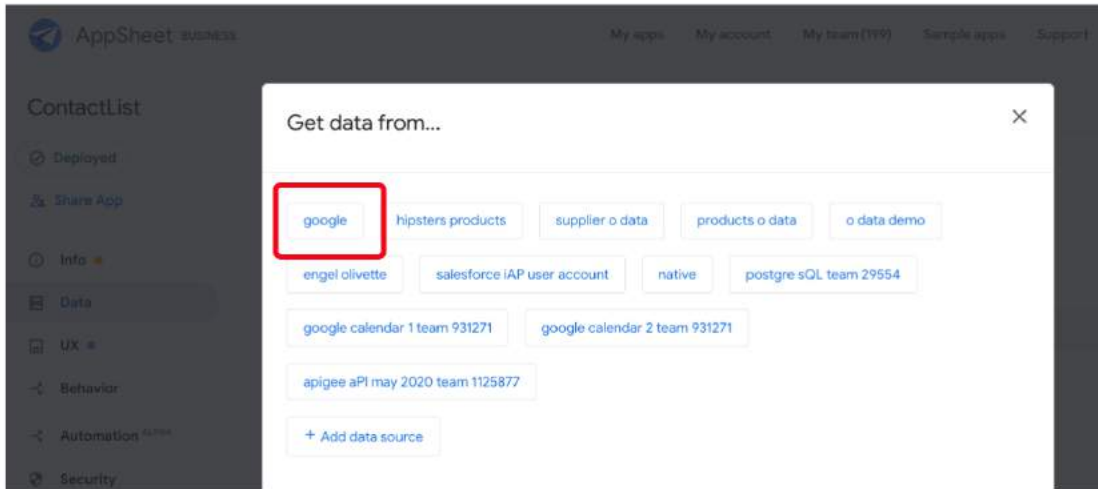
- Action name:** Update company contact
- For a record of this table:** Contact
- Do this:** Data: add a new row to another table using values from this row
- Table to add to:** CompanyContacts
- Set these columns:**

First Name	=	[FirstName]	⬆	🗑
Last Name	=	[LastName]	⬆	🗑
Email	=	[Email]	⬆	🗑
Company	=	[AccountId]	⬆	🗑
Date Created	=	[CreatedDate]	⬆	🗑

Configuring Leads entity from Google Sheets datasource

To configure your “Leads” entity from Google Sheets:

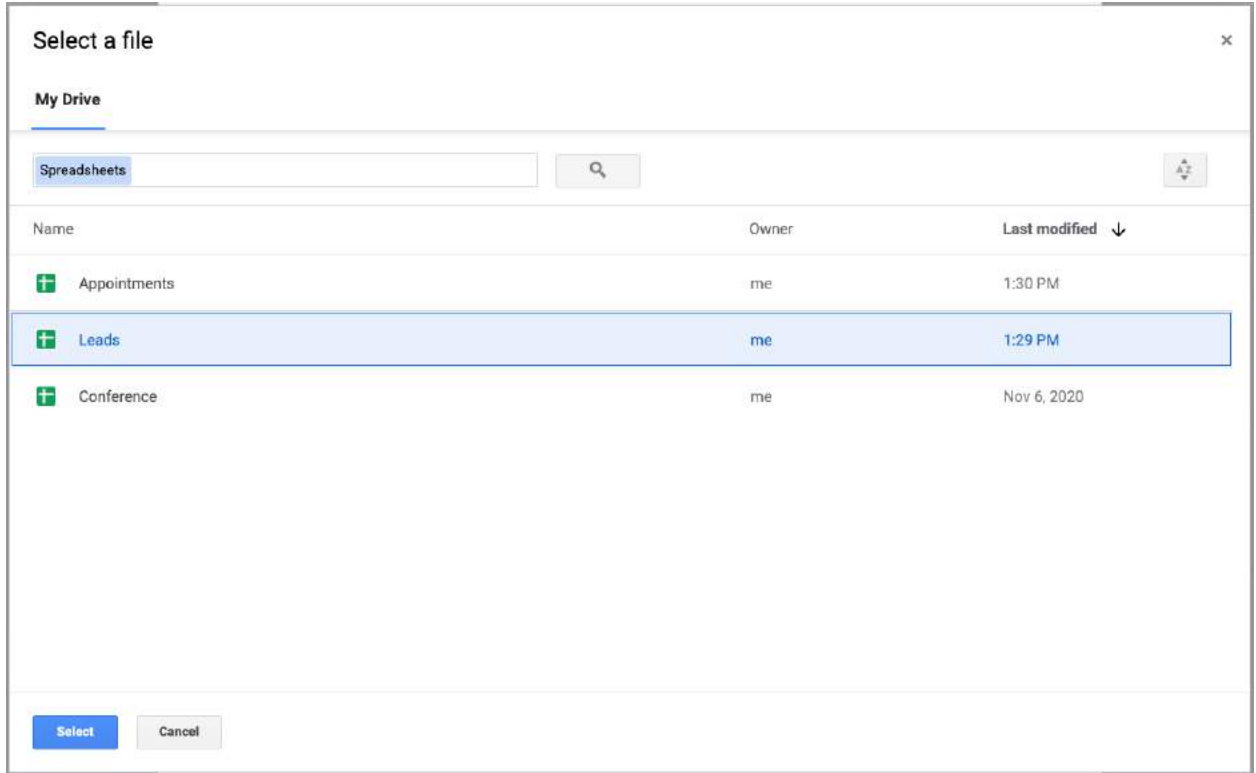
1. From the AppSheet UI, click **Data** in the left navigation bar.
2. Click **New Table**.
3. Select your configured Google Data Source.



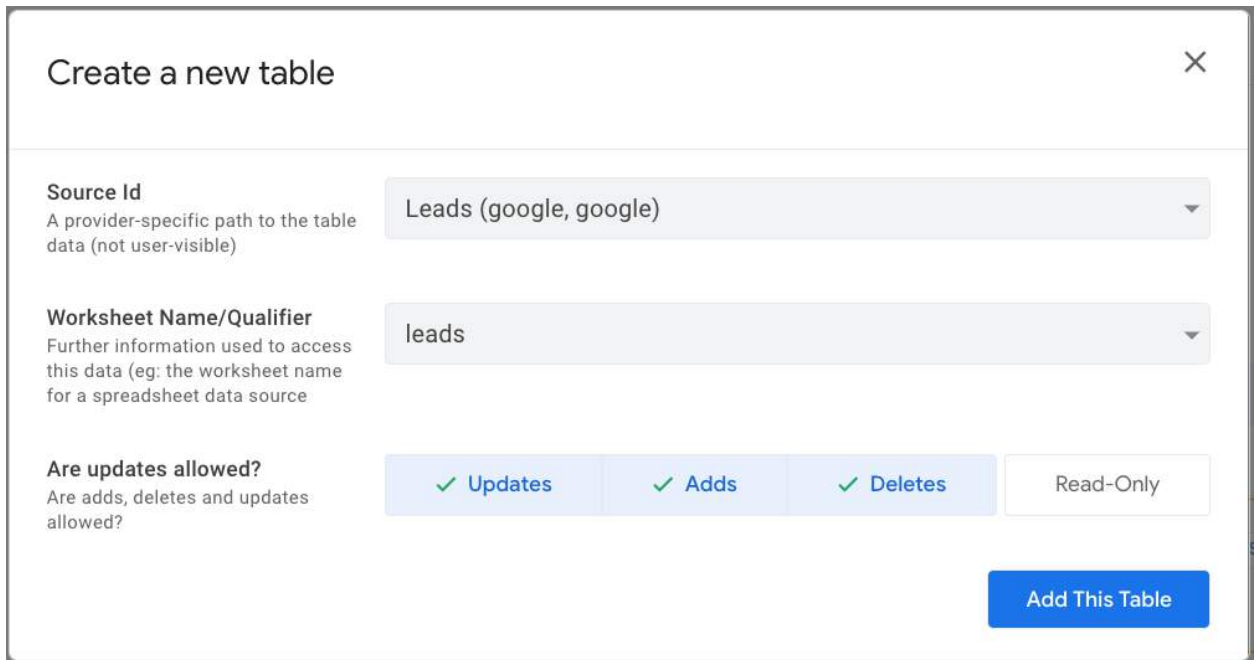
4. Create a new Google Sheet named “Leads.” Use the column names shown below and name the Sheet (tab) “leads”. The Sheet should be in the Google Drive accessible to your AppSheet account.

	A	B	C	D	E	F	G	H	I
1	First Name	Last Name	Email	Phone	Company	Date	FollowupRequested	FollowupAfterDays	Profile
2	Test	Lead1		m 3148889292	Acme Inc	11/1/2020	Y		5 High
3	Test	Lead2		m 4082224455	Acme Inc	11/1/2020	Y		6 Medium
4	Test	Lead3		m 9256667788	Acme Inc	11/10/2020	Y		7 Low
5	Test	Lead6		m 6691112222	Acme Inc	1/6/2021	N		5 Low
6									

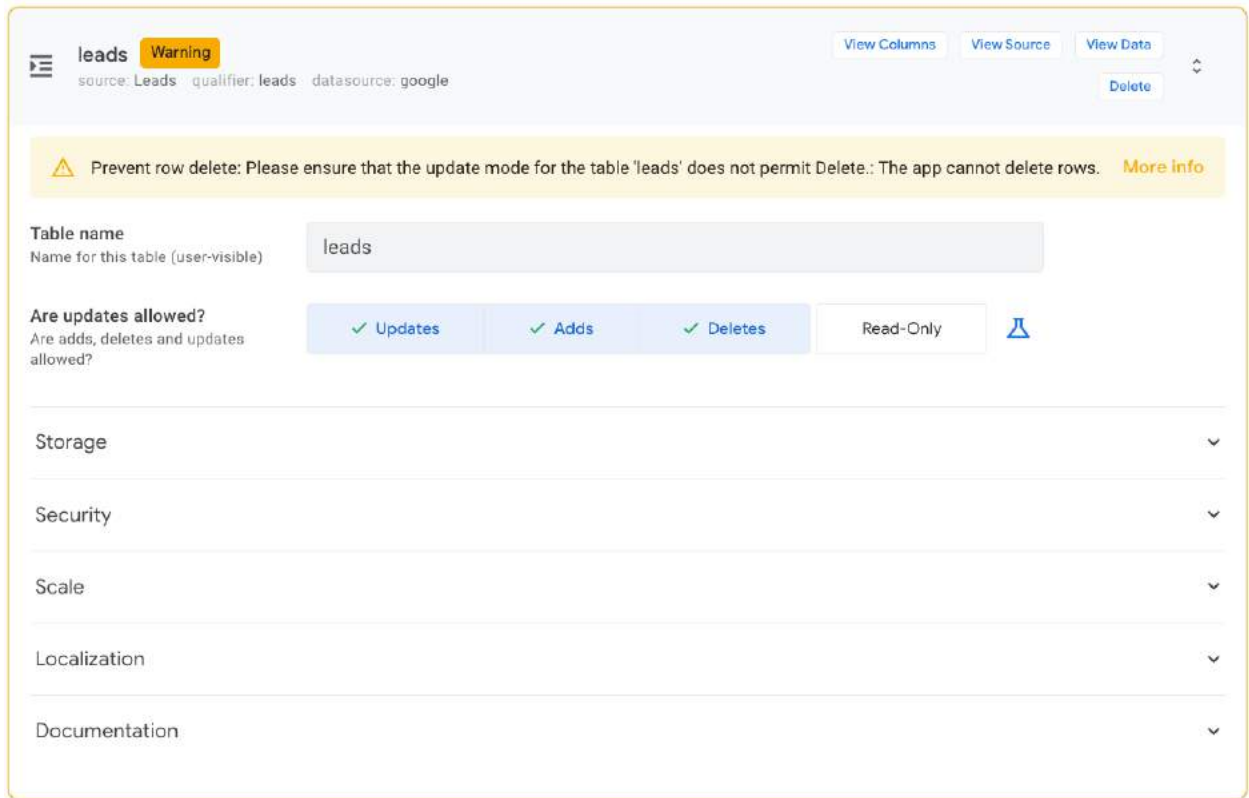
5. Use the file browser to select the appropriate **Leads** Sheet as the data source.



6. Select **leads** as the table for your entity.



7. Click **Add This Table**.

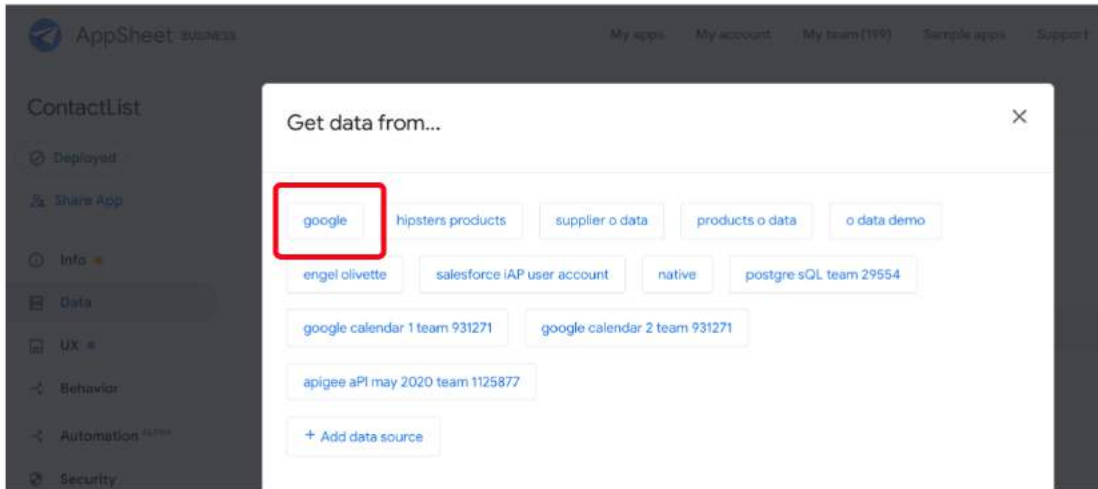


8. Click on “View Columns” and ensure that the data type of “FollowupAfterDays” column is “Number” and clear out initial entry of NOW() if it exists

Configuring appointments entity from Google Sheets datasource

To configure your “appointments” entity from Google Sheets:

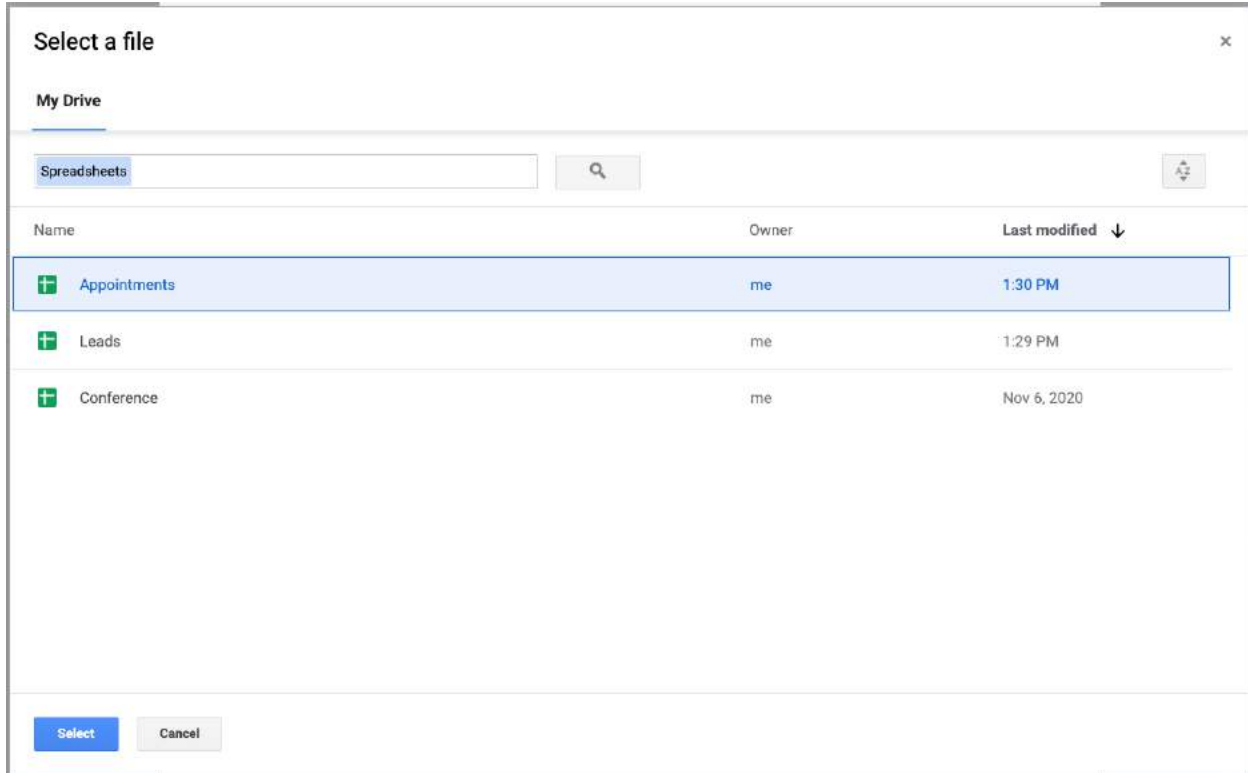
1. From the AppSheet UI, click **Data** in the left navigation bar.
2. Click **New Table**.
3. Select your configured Google Data Source.



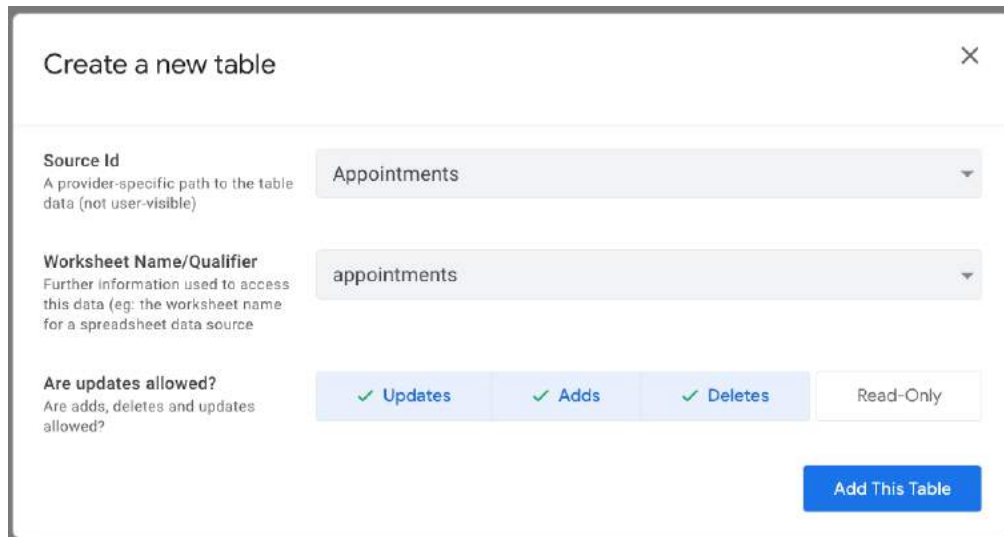
4. Ensure you create a new Google Sheet named “Appointments” (with the column names as below, the Sheet (tab) name should be “appointments”) in the Google Drive accessible to your AppSheet account

	A	B	C	D	E	F
1	Full Name	Email	Appointment Date	Confirmation Required	Confirmation Status	Confirmation Timestamp
2	Test Lead1		11/6/2020	Y	confirmed	1/20/2021 12:34:56
3	Test Lead2		11/7/2020	N	not-required	1/19/2021 14:28:43
4	Test Lead3		11/17/2020	N	not-required	1/20/2021 18:35:27
5						

5. Use the file browser to select the appropriate **Appointments** Sheet as the data source.



6. Select **appointments** as the table for your entity.



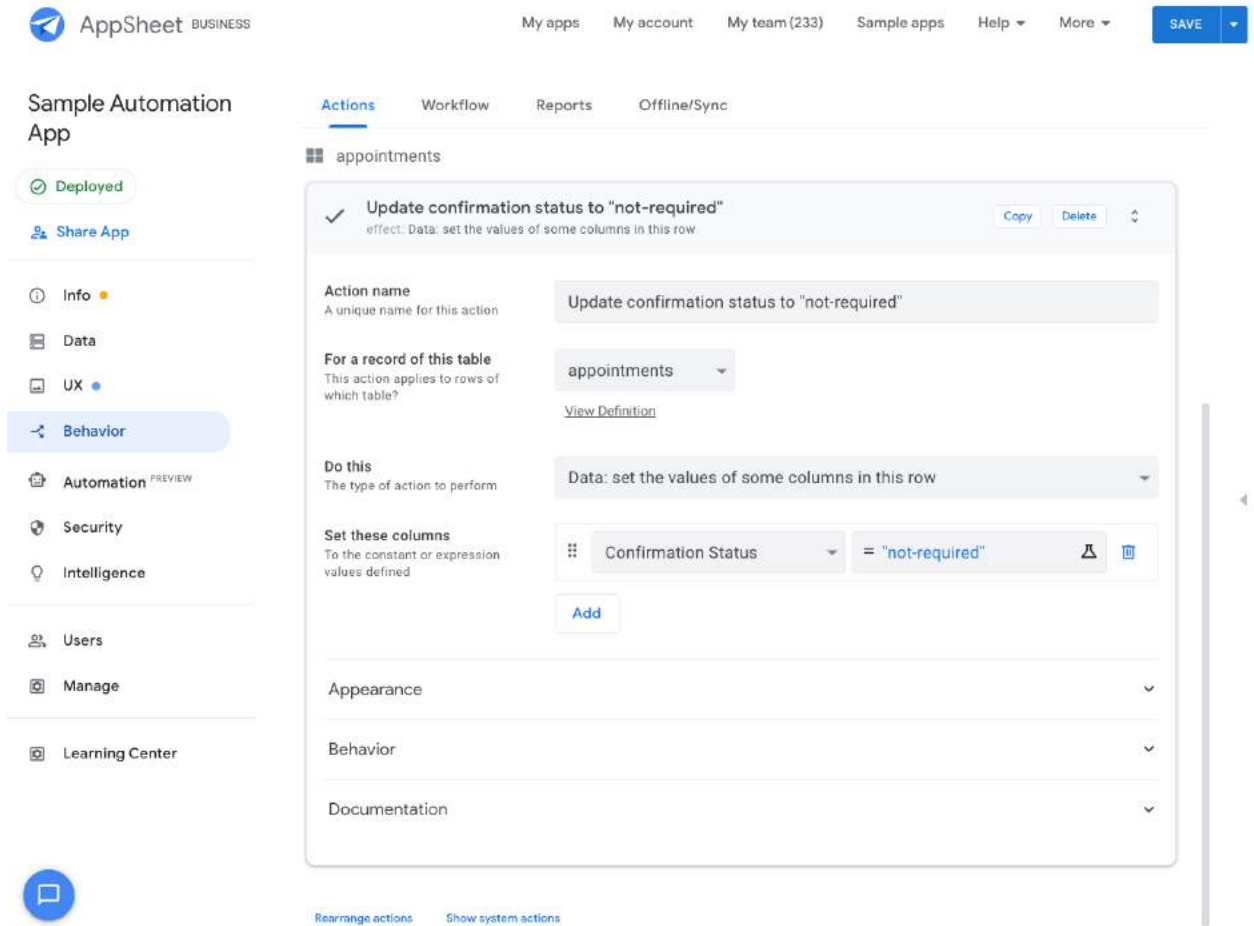
7. Make sure the **Confirmation Required** column type is **Text**.

appointments View Table Add Virtual Column Regenerate Structure

7 columns: Full Name Full Name

NAME	TYPE	KEY?	LABEL?	FORMULA
_RowNumber	Number	<input type="checkbox"/>	<input type="checkbox"/>	=
Full Name	Name	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	=
Email	Email	<input type="checkbox"/>	<input type="checkbox"/>	=
Appointment Date	Date	<input type="checkbox"/>	<input type="checkbox"/>	=
Confirmation Required	Text	<input type="checkbox"/>	<input type="checkbox"/>	=
Confirmation Status	Text	<input type="checkbox"/>	<input type="checkbox"/>	=
Confirmation Timestamp	DateTime	<input type="checkbox"/>	<input type="checkbox"/>	=

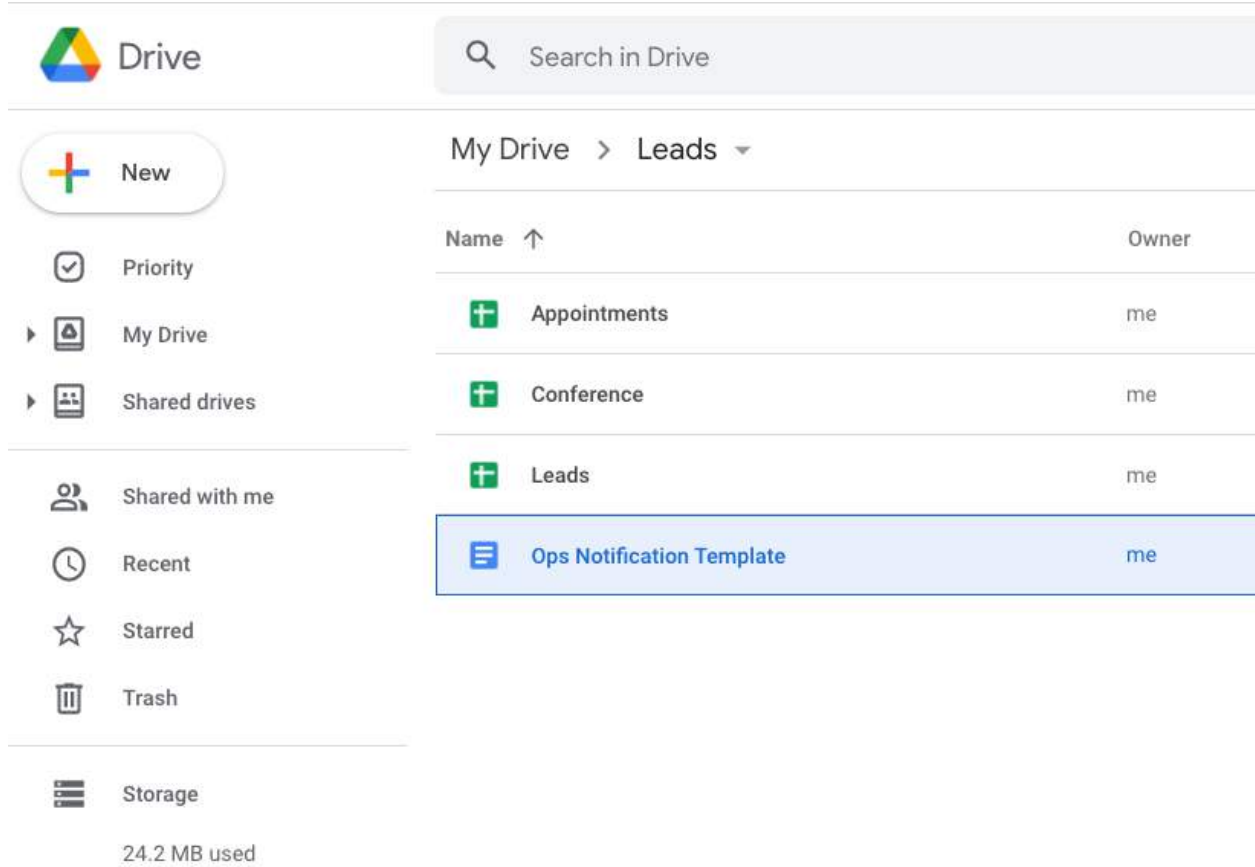
8. Create a new Action (under **Behavior**) as follows:



Configuring an email template for ops notification

To configure an email template, create a document in your Google drive. Follow the steps below to configure your email template:

1. Navigate to your Drive and create a new document named **Ops Notification Template**.

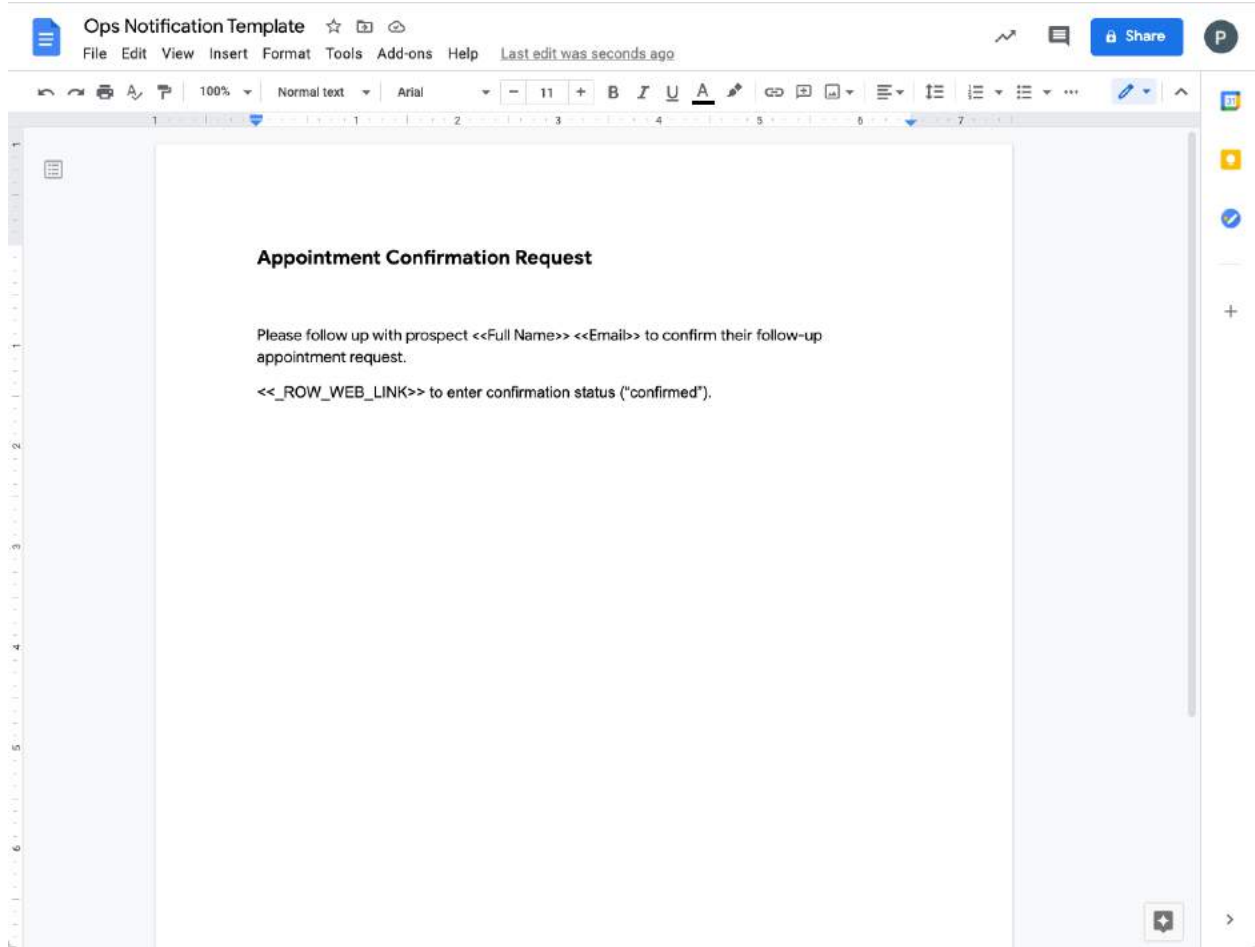


2. Enter the following text in the document, as shown in the images below:

Appointment Confirmation Request

Please follow up with prospect <<Full Name>> <<Email>> to confirm their follow-up appointment request.

<<_ROW_WEB_LINK>> to enter confirmation status (“confirmed”).



Document processing states and status codes

Document processing in AppSheet relies on AI to identify and extract content from documents. Under some circumstances, the AI may fail to extract high quality structured data from a document. To help understand the status of the content extraction, a `StatusCode` column is included in the document data. In addition to the status code, a textual description of the potential issue is included in a column named `AttentionDetails`. This is intended to help you understand what may be the issue so you can correct it.

Potential status codes and values

As noted earlier, document extraction may result in the following status codes:

StatusCode	AttentionDetails
EXTRACTION_ERROR	The document is either of an invalid type or did not appear to have the correct content.
POOR_QUALITY	Some of the values extracted have low confidence scores in the AI model. This implies that there is uncertainty in the accuracy of the data.
INCONSISTENT_CURRENCY	Extracted data from the document appears to reference multiple currencies.
EMPTY_CURRENCY	The document does not specify a currency, though one is expected.

The intended flow of document extraction is to allow you to automatically utilize the extracted data in downstream processes and actions. In order to support this, two additional indicator columns are included in the document data:

- `IsEdited`
- `NeedsAttention`

These columns can be used to understand if the document data is ready for downstream use.

When the value in the `NeedsAttention` column is `TRUE` (Yes), this indicates that a `StatusCode` is present requiring manual review. For example, if the content of a field may have low quality, or the currency code present in a field is unknown, the `NeedsAttention` flag indicates that the data should be reviewed manually, corrected, and results approved for downstream uses. When an edit is made to the data after processing, the `NeedsAttention` column is set to `FALSE` and the `IsEdited` column is set to `TRUE`. This indicates that the data is now ready for downstream use.

Below is a brief description of several common `StatusCodes` returned on data slices, and the implications for the resulting document data.

- `AND([NeedsAttention], [StatusCode] = "EXTRACTION_ERROR")`: This slice of data represents the files that could not be extracted. This usually indicates files of

the wrong file format or files with content that is not aligned with the intended use.

- `OR(NOT([NeedsAttention], [StatusCode] != "EXTRACTION_ERROR")`: This slice of data represents all of the files that had content extracted. This slice may contain potentially incomplete or invalid data. This slice is useful for viewing all the updated invoice data in one place, for example, as a view in an app.
- `AND([NeedsAttention], [StatusCode] != "EXTRACTION_ERROR"`: This slice of data represents all of the files that have been flagged as needing manual review. This slice is helpful to build a human into the process to verify and correct any potential data issues.
- `NOT[NeedsAttention]`: This slice represents all of the document data that is considered high quality and/or human verified. This slice is appropriate for integration into downstream use cases for automation or other purposes.

Why is there a separate currency code column?

For documents that have some type of monetary column, the value of the monetary columns and the currency code are separated out. This provides more flexibility in using the extracted data, such as supporting a centralized repository of documents spanning multiple currencies to allow customization of downstream processes. However, there is only ever considered a single currency for a single document; inconsistencies in currency type found in the extraction are flagged.

Where can I store my files for Document processing?

At this time, AppSheet supports processing documents found on Google Drive. Support for additional file storage providers for document processing is underway.

What does 'collection of files' mean?

In addition to document content extraction, AppSheet now supports exposing your folder contents as a table in your application itself. This feature allows app creators to expose files directly in their app, and use file metadata in their application logic (e.g. filtering by names, dates modified, etc).