Adobe Workfront

Adobe

ACTIVITY BOOK

Adobe Workfront Fusion

This workbook documents the steps of the walkthrough exercises contained in the Workfront Fusion learning program on Workfront One.

To complete these activities, you will need access to your organization's Adobe Workfront Fusion instance and an Adobe Workfront test drive. Talk with your internal Workfront team to learn how to access Workfront Fusion.

Inside this guide

Before you start 4

Create teams and users in Workfront Fusion 5

Log in as a new user 7

Initial scenario design 8

Beyond basic mapping 16

Filters 18

Access previous versions 20

Introduction to universal connectors 22

Routers 29

Routing patterns 36

Set/Get variables 39

Introduction to iterators 45

Aggregation 48

Advanced aggregation 51

Execution history 55

Exploring runs, cycles, and bundles 58

Switch function 61

Switch module 63

Webhooks 65

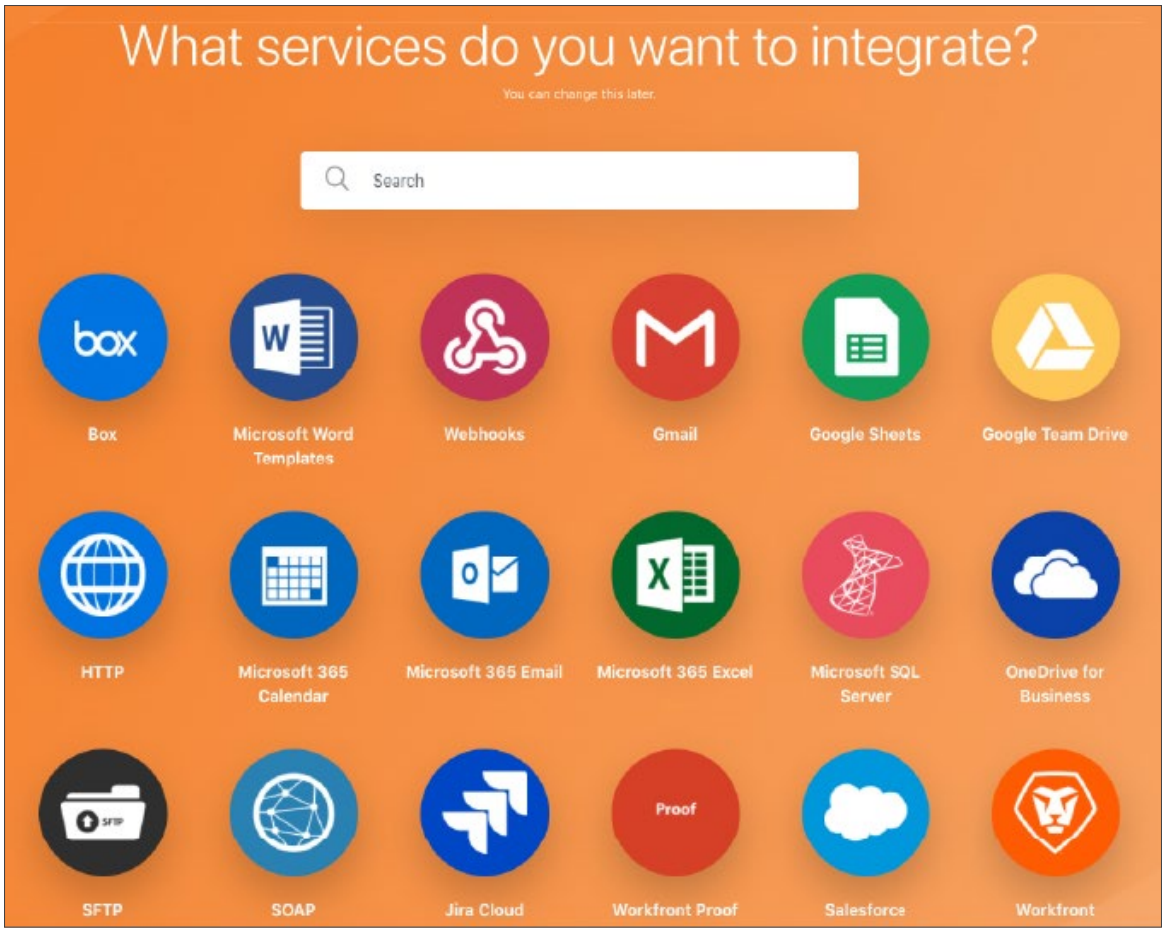
Data structures 70

Data stores 79

Working with JSON 89

DevTool 97

Before you start



1. GET SET UP

Before you can complete these exercises, you'll need to get a few things set up. Go to the **Welcome to Workfront Fusion** learning path for instructions.

2. WATCH THE VIDEOS

This workbook documents the steps of the walkthrough exercises. It is assumed that you have watched the corresponding walkthrough videos before attempting the exercises. The videos provide valuable insights and more complete explanations.

3. WORK IN ORDER

Exercises must be completed in sequential order. In many cases, you start one exercise by cloning a scenario created in a previous exercise.

4. LEARN MORE

Find additional information about **Workfront Fusion** on Workfront One.

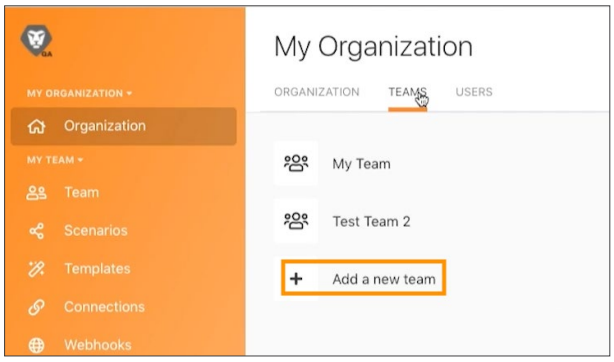
Create teams and users in Workfront Fusion

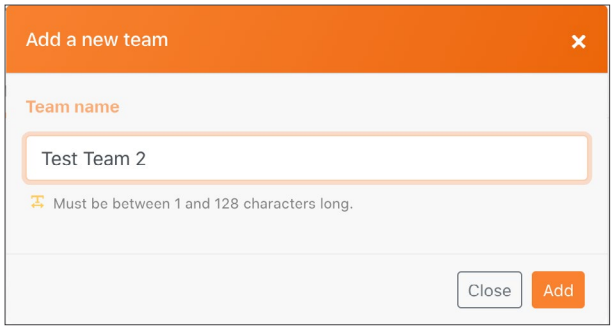
To create users in your Workfront Fusion organization you must:

- Be a Workfront Fusion administrator
- Designate or create a team for the new user

CREATE A TEAM

To create a new team, go to the Teams tab in your organization and click Add a new team.



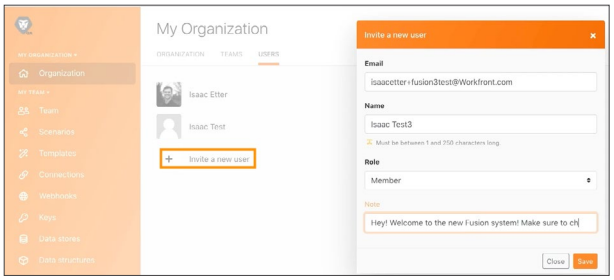


Name the team.

CREATE A USER

To create a new user, go to the Users tab and click Invite a new user. Fill out the form that appears.

In most cases you will probably want to set the Role to Member. The text you put in the Note field appears in an email to the user.

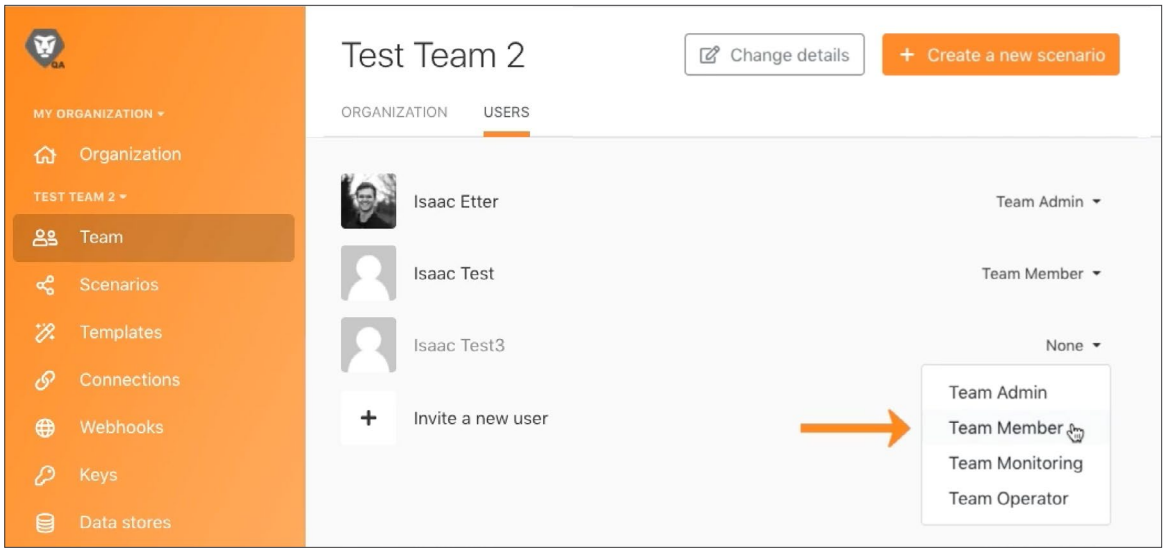


Create teams and users (cont.)

In order for a new user to create scenarios in Workfront Fusion, they need to be on a team.

ADD USERS TO A TEAM

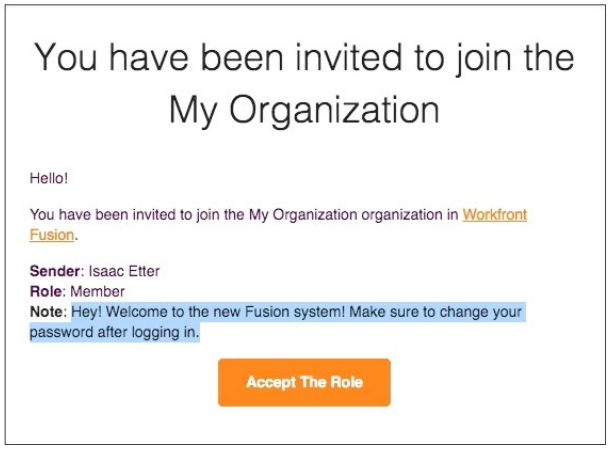
1. Select a team from the Teams tab within your organization.
2. Select the Users tab within the team.
3. Then assign the new user a role on the team, using the drop-down menu to the right of the person's name.



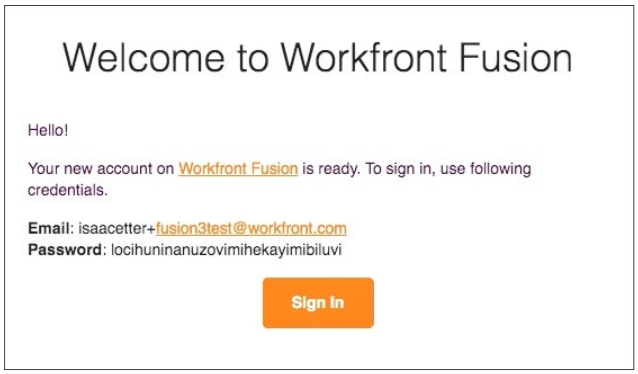
Log in as a new user

When you are invited as a new user to a Workfront Fusion instance you receive two emails.

One email has a note the Workfront Fusion system administrator added when they created your profile and invited you to the organization. At the bottom of the email is Accept The Role button. Don't click this button yet!



The other email contains your login credentials.



To start using Workfront Fusion, click the Sign In button in the second email and sign in using the password provided. After signing in the first time, you are prompted to change your password.

Once you've signed in, go back to the other email and click the Accept The Role button.

Once you do that, go back to Workfront Fusion and refresh the page. You can now see your team and the overview sections in the left panel.

Initial scenario design

Learn some basic navigation tips for when you first log into Workfront Fusion, as well as building your first scenario.

Exercise overview

Create a new project in Workfront for each row in the Project List CSV file.

Steps to follow

1. Create a folder in the Scenario section named "Fusion enablement exercises."

2. Click into the folder, then click Create a new scenario.

3. On the next page, search for Workfront and select that app. Then click Continue.

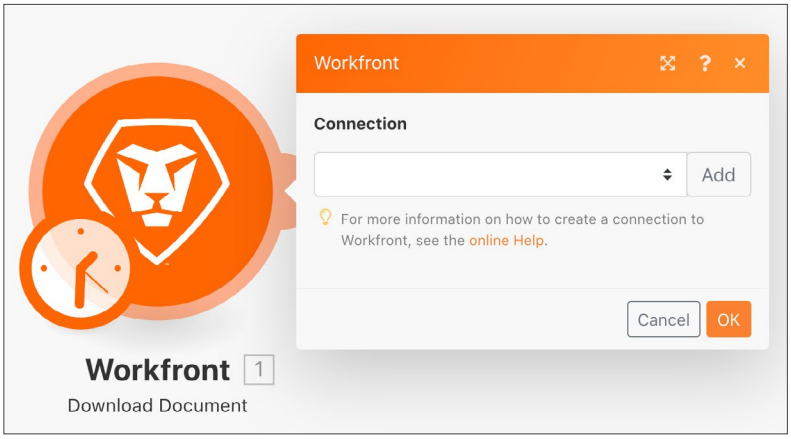
4. At the top left of the scenario designer screen, rename your scenario to "Initial scenario design."

5. Click the empty trigger module in the center of the screen and select the Workfront app, then select the Download Document module.

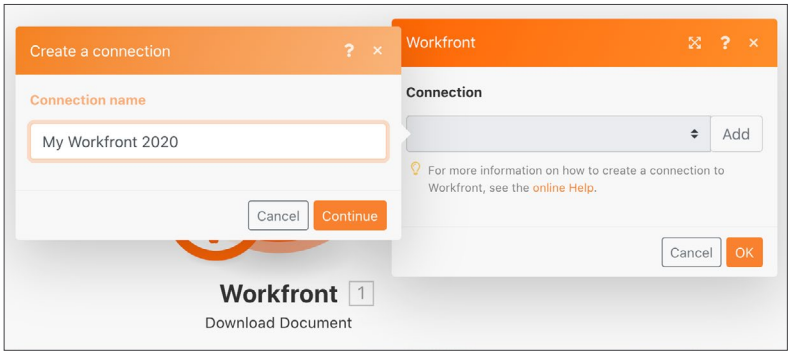
Initial scenario design (cont.)

Authenticate the module's connection to your Workfront account.

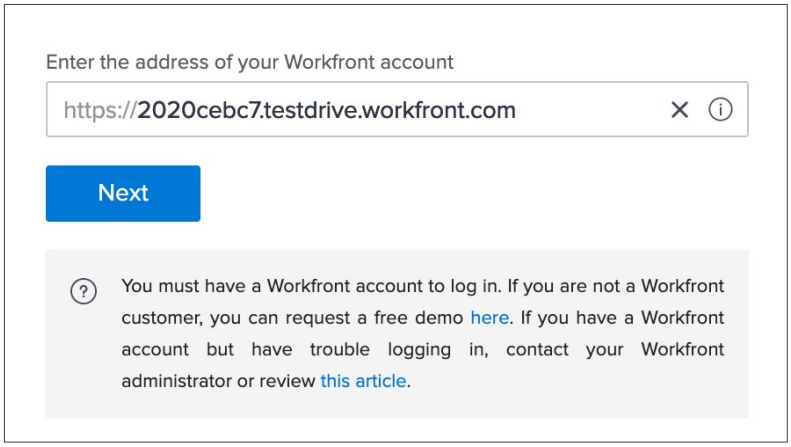
6. To create a connection for the first time, click the Add button.



7. Give the connection a name, such as "My Workfront 2020."



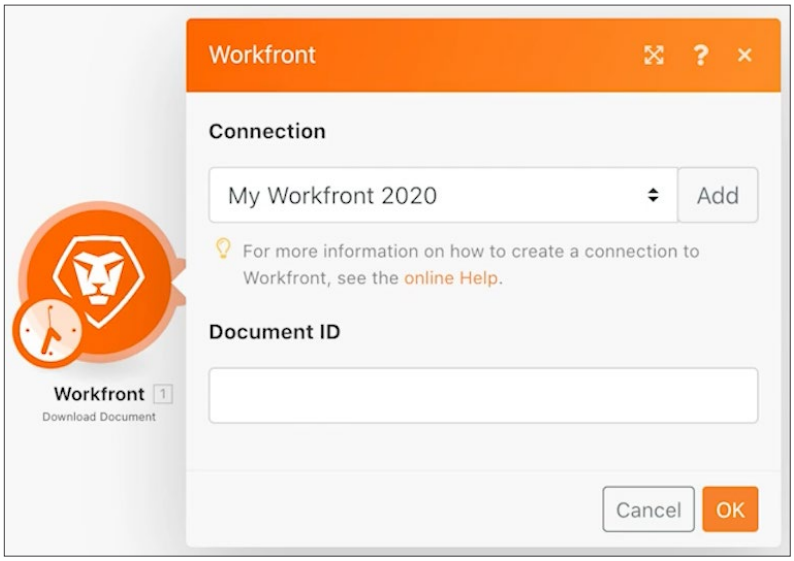
8. Enter the URL of your Workfront instance, then click Next.



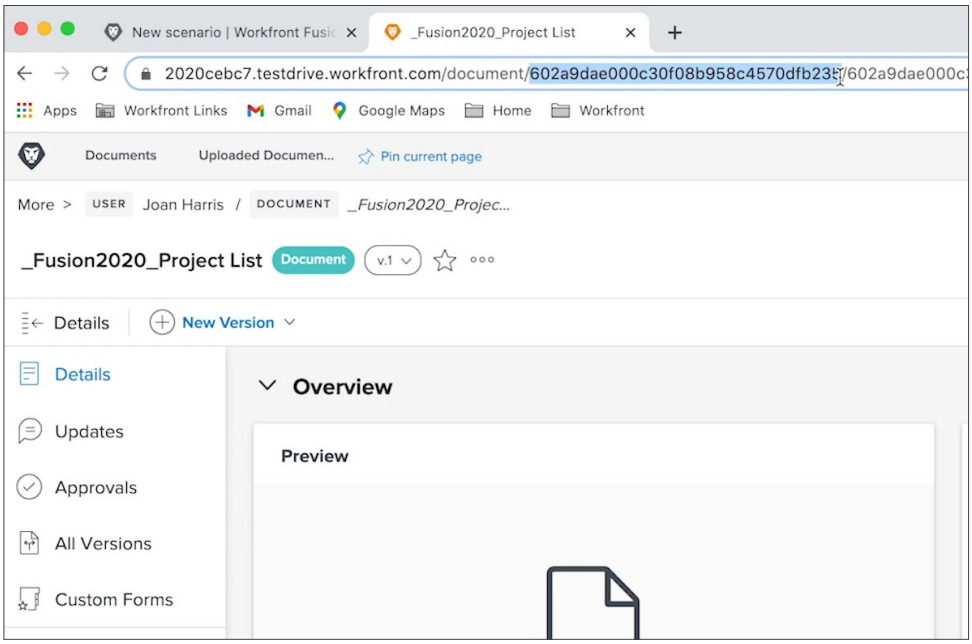
Initial scenario design (cont.)

9. Enter your password and click Log in.

The connection is established. Now enter the document ID of the document you want to download from Workfront.



10. Go back to Workfront. In the "Fusion Exercise Files" folder, select "_Fusion2020_Project List.csv" and click Document Details in the left panel. Copy the document ID number from the URL address (this is the first long number in the URL).



11. Go back to Fusion and paste the number in the Document ID field and click OK.

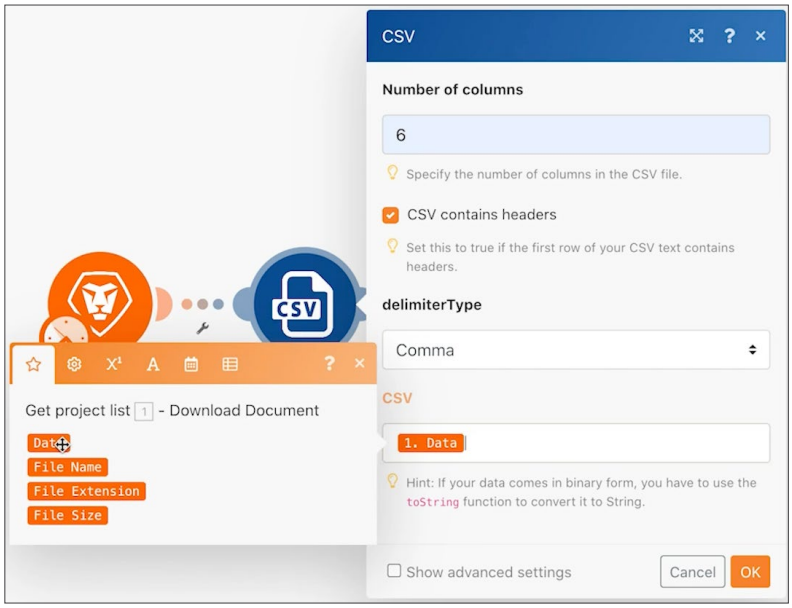
Initial scenario design (cont.)

12. Best practice is to rename modules as you create them. Right-click on the Workfront module and choose Rename. Name the module "Get project list."

Next you're going to parse the CSV file you just downloaded so you can access each row in the file. You'll use this information when you create a project from each row.

13. Click the right side of the Workfront module to add another module. Search for the CSV app and select the Parse CSV module.

14. Set up Parse CSV for 6 columns, CSV contains headers, Comma delimiterType, and put Data in the CSV field. Then click OK.



15. Rename this module "Parse project list."

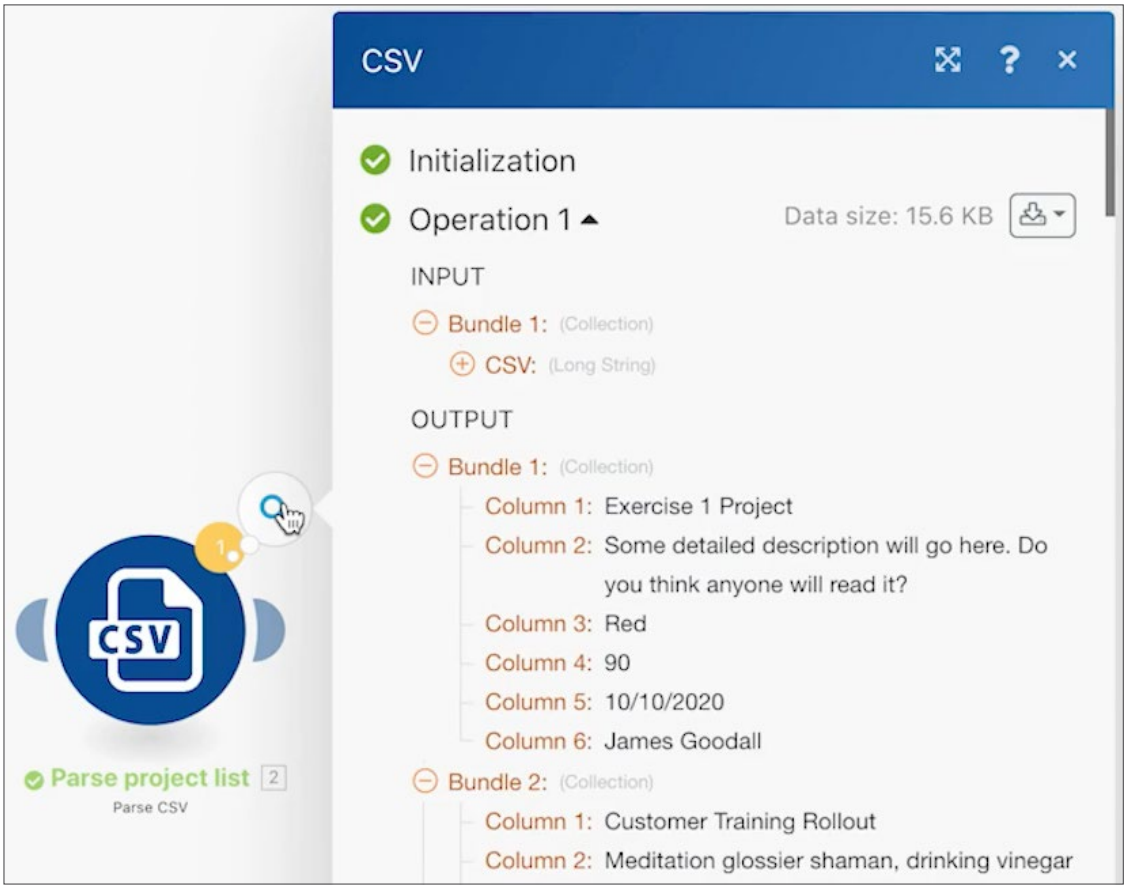
16. At the bottom of the scenario designer, click Save to save your scenario.

17. Click Run once to view the output.

- Ignore the warning that a transformer should not be the last module (this is true, but doesn't matter for this test). Click Run Anyway.

Initial scenario design (cont.)

18. Open the execution inspector on the Parse CSV module to see the inputs and the outputs of the module. There is one bundle (a CSV file) as input and several bundles as outputs (one bundle for each row in the CSV file). It should look something like this:



Add a module to create a project for each row in the CSV file.

19. Add another module. Select the Workfront app, choosing the Create Record module.

20. Set the Record type as Project.

- Search for it by starting to type a few letters, such as "proj," to get right to it.

21. Then use Cmd/Ctrl+G to find Name (project name). Check the box next to Name; the field appears below.

22. Now check the boxes next to Planned Start Date and Priority.

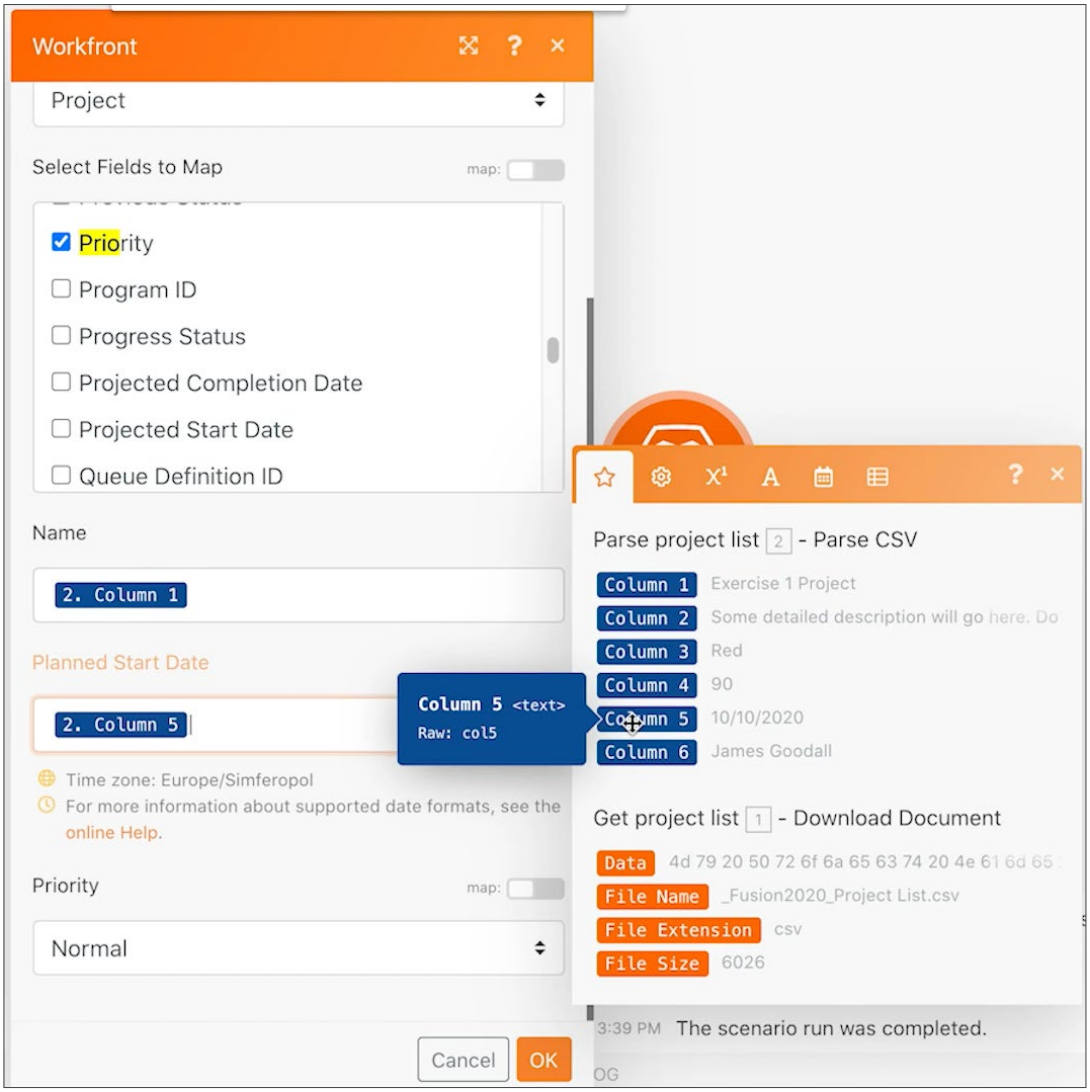
Initial scenario design (cont.)

23. Click into the Name field so the mapping panel appears. Click the Column 1 field from the Parse CSV module to add it to the Name field. This is the project name from the CSV file.

24. For the Planned Start Date, click Column 5 from the Parse CSV module.

25. For Priority, choose Normal from the drop-down menu.

Your mapping panel should look like this:



26. Click OK.

- Note: If you don't click OK and accidentally click back into the designer, your work does not save and you will have to map again.

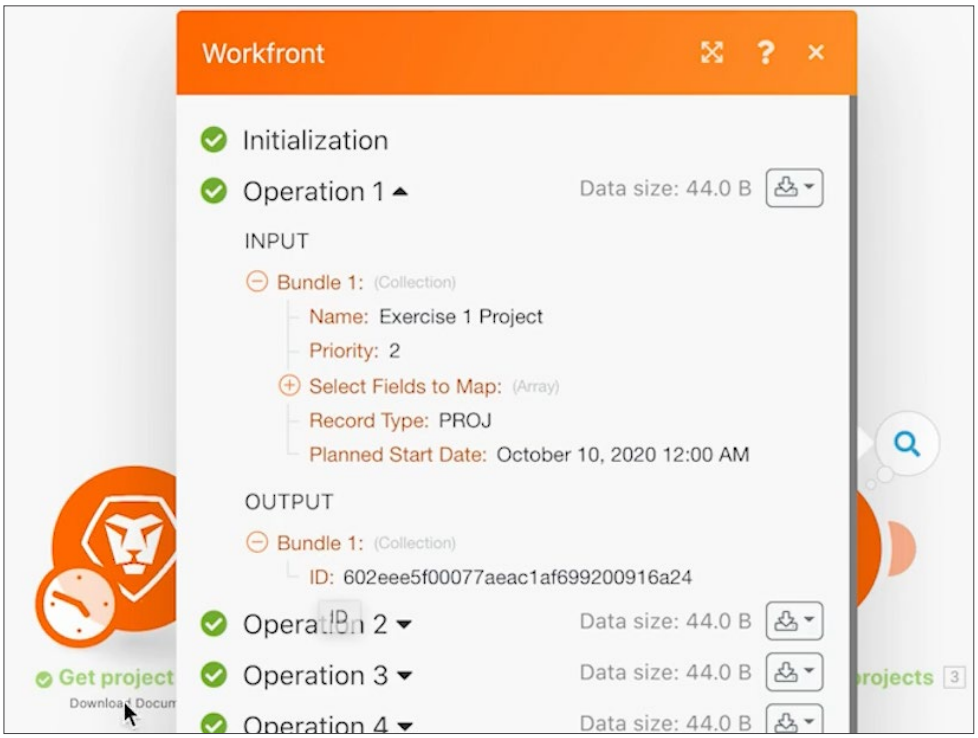
Initial scenario design (cont.)

27. Right-click the Workfront module and rename it "Create Workfront projects."

28. Save your scenario and click the Run once button.

29. Click the execution inspector at the top right of the last module.

- You'll see 20 operations were performed. Each operation took a bundle, meaning one row, from the CSV file as input and output one bundle, which was a project created in Workfront. The project ID of the project created appears with the output bundle.



Using notes

30. Notes help create more visibility into the scenario design. To add a note to the Create Workfront projects module, right-click and select Add a note. A panel at the right of the designer window pops out so you can add a note to the module. Type in "Create a project with Name, Planned Start Date, and Priority mapped from the CSV file."

31. Add another note to describe what the trigger module (the first Workfront module) is doing.

Initial scenario design (cont.)

32. Close the notes panel by clicking the X at the top-right corner.

- Access the notes again by clicking the notes button in the bottom toolbar or by right-clicking any module and adding a new note.
- Notes are sorted in reverse chronological order.
- An orange dot appears on the Notes button once notes are added.

33. Save the scenario by clicking the Save button in the controls toolbar.

34. You can view the projects created in your Workfront instance.

Beyond basic mapping

Learn how to use the mapping panel formulas to manipulate or convert field(s) sent to a module.

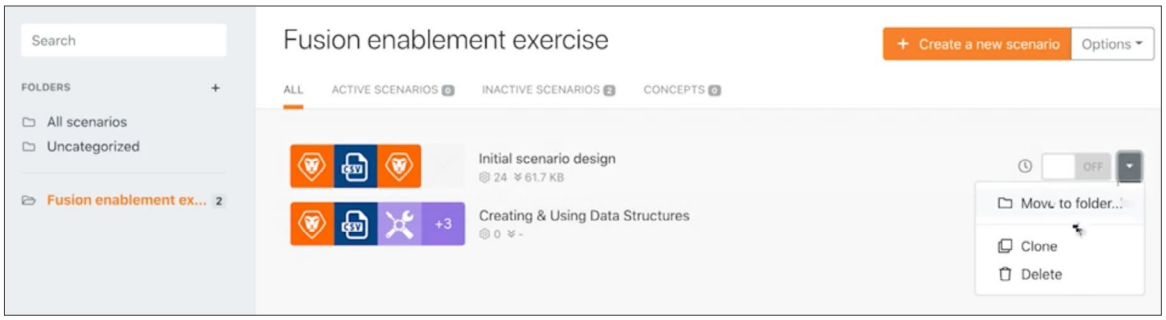
Exercise overview

Change the project name, planned start date, and priority from the Initial scenario design walkthrough exercises using the mapping panel formulas.

Steps to follow

Make a clone of your Initial scenario design scenario.

1. Select the Clone option to the right of the Initial scenario design in the scenario section, as shown below. Name it "Beyond basic mapping."



Beyond basic mapping (cont.)

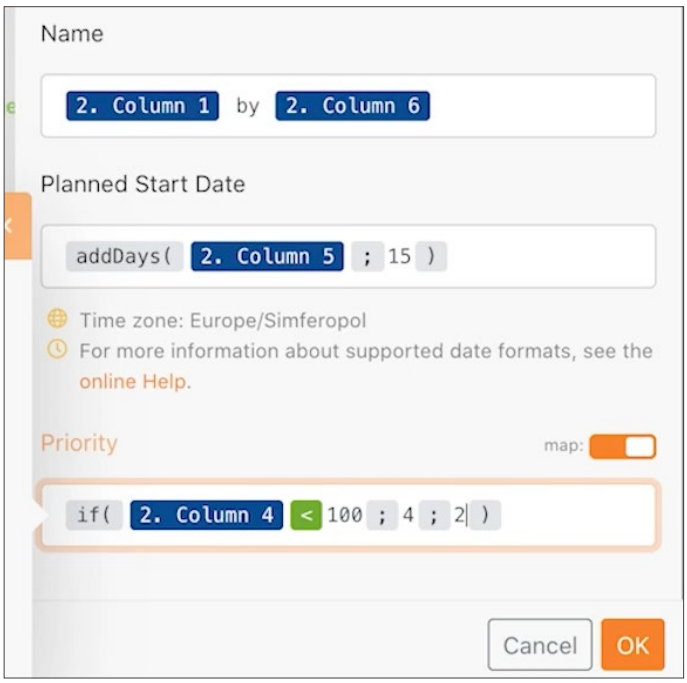
Now we're going to use the mapping panel in the Create Workfront projects module to configure the project name, planned start date, and priority fields.

2. Click the Create Workfront projects module to edit the settings. Using the mapping panel, change the Name field to be "[My Project Name] by [Sponsor]."
    - The [My Project Name] is column 1 from the Parse CSV module and [Sponsor] is column 6. The word "by" is just typed between the two.

3. Next go to the Planned Start Date and use the addDays formula to add 15 days to the field, as described in the Beyond basic mapping walkthrough video.

4. Find the Priority field and toggle the Map button at the top right of the field. The picklist menu changes to a number. Create an if statement to label a project as High(4) priority if the CSV file confidence rating is less than 100, otherwise it can be Normal(2).
    - The confidence rating is in Column 4.

At this point, your mapping panel should look like this:



5. Click OK and then click Run once.

6. Find the project in your Workfront instance to make sure everything was mapped correctly.

7. Save your scenario.

Filters

Learn how to use the filter between modules to allow only certain types of bundles through.

Exercise overview

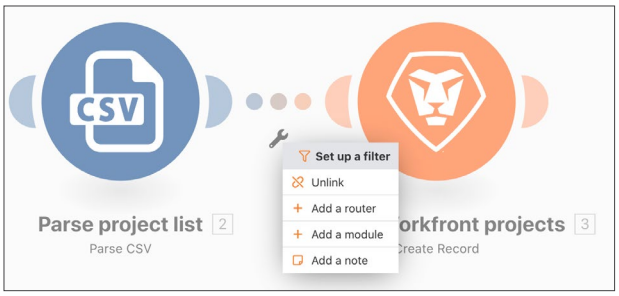
Add a filter between the two modules in the Beyond basic mapping scenario to only create projects that have a "Red" project color in the CSV file.

Steps to follow

1. Create a clone of the "Beyond basic mapping" scenario and name it "Using the mighty filter."

Add a filter before the Create Workfront projects module to only allow red projects to be created.

2. Add a filter by clicking the dotted line connecting the modules or clicking the wrench and selecting Set up a filter.

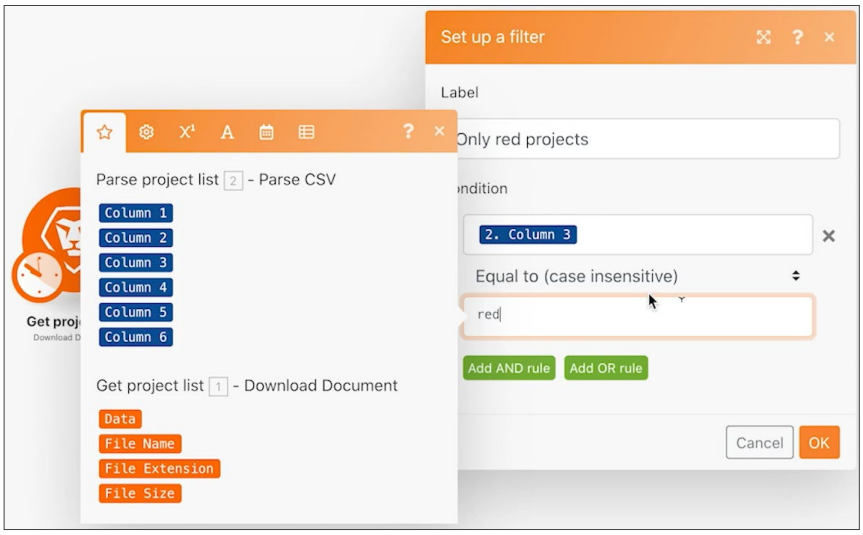


3. Use the Label field to name the filter "Only Red Projects."

4. In the Condition field, map the Project Color field (Column 3 in the CSV file). Select the Equal to (case insensitive) operator, and then type in "red."

Filters (cont.)

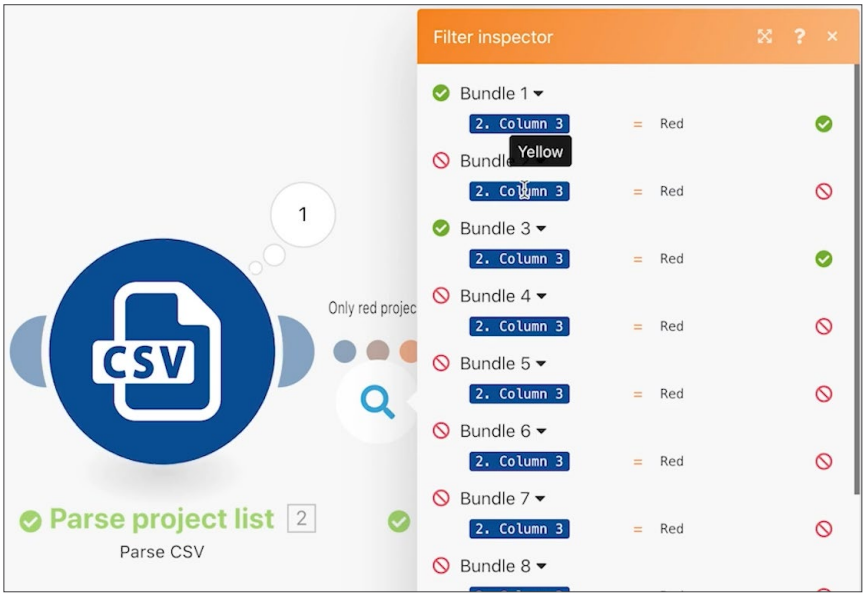
5. Click OK.



Test the filter and verify the results.

6. Click Save to save the scenario and then Run once.

7. Click the execution inspector for the filter to see how each bundle was examined by the filter and either passed or failed to move on to the Create Workfront projects module.



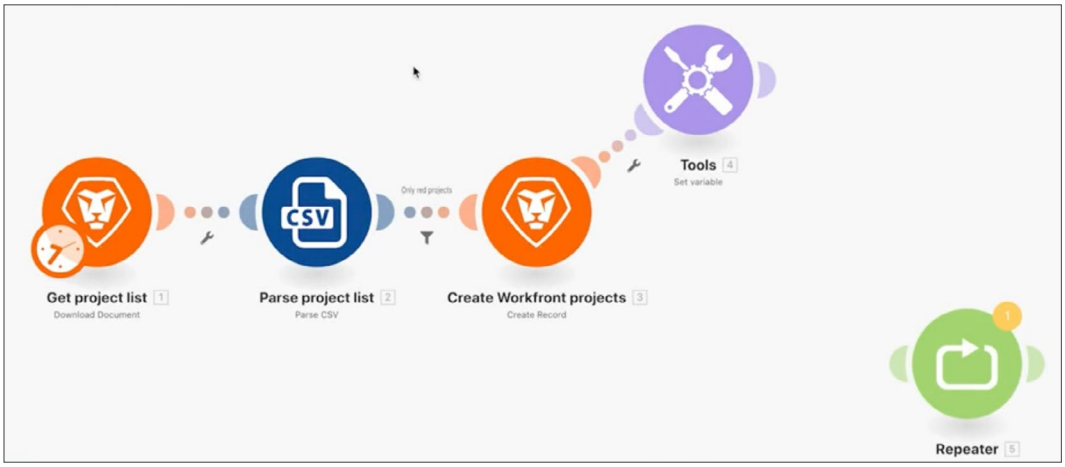
8. Find the projects created in your Workfront instance.

Access previous versions

Learn how to return to a previous version of a scenario.

Exercise overview

Discover how you can restore previous versions after you've made changes to a scenario and saved it multiple times.

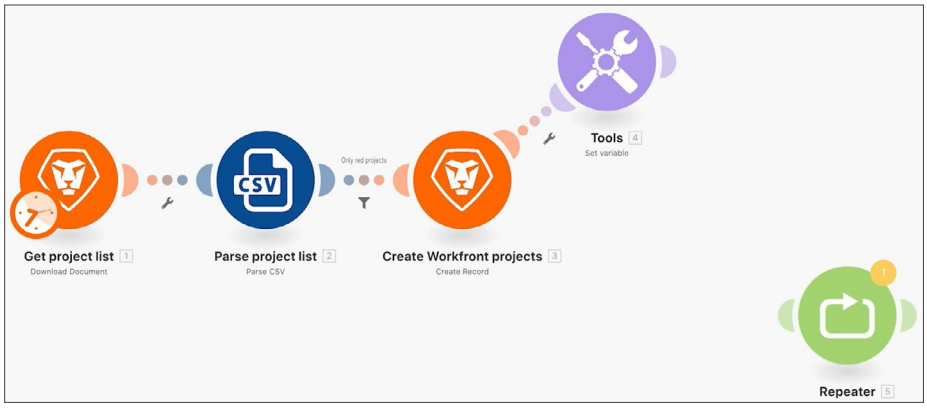


Steps to follow

1. Clone your Using the mighty filter scenario and name it "Accessing previous versions."
2. Add a Set variable module after the Create Workfront projects module. Name the variable "Test."
3. Drag it to a new position and save the scenario.

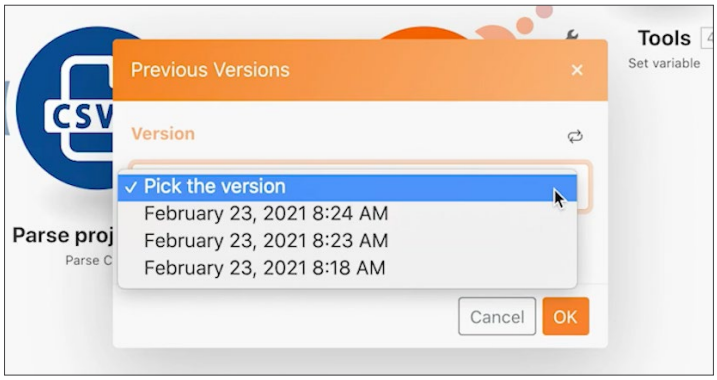
Access previous versions (cont.)

4. Add a Repeater module, unlink it from the previous module, and save the scenario again.



5. Now delete all of your modules and save.

6. In the toolbar, click the three-dot menu and click the Previous Versions option. The picklist shows the date and time stamps for each version saved.



7. Choose a previous version and notice how the scenario in the designer returns to where you saved.

Introduction to universal connectors

Expand your understanding of working with REST universal connectors and working with the data returned.

Exercise overview

Using a Pokemon character in a spreadsheet, call the Poke API through an HTTP connector to gather and post more information on that character.

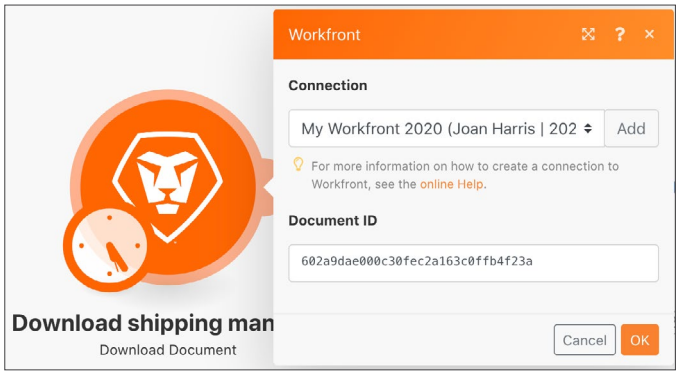
Steps to follow

Download the CSV file from Workfront.

1. In the Workfront "Fusion Exercise Files" folder, select "_Fusion2020_Shipping Manifest.csv" and click Document Details.
2. Copy the first ID number from the URL address.
3. Create a new scenario in Workfront Fusion. Name it "Using universal connectors."
4. Start with the Download Document module from the Workfront app.
5. Set up your Workfront connection and include the Document ID you copied from the Workfront URL.

Introduction to universal connectors (cont.)

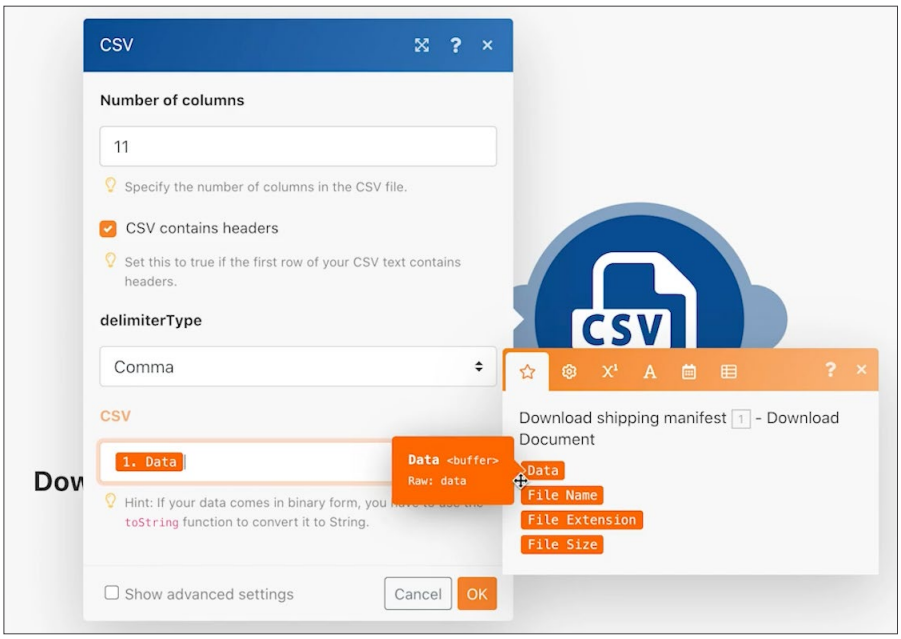
6. Rename this module “Download shipping manifest.”



Parse the shipping manifest data.

7. Add another module, selecting Parse CSV.

8. Set up Parse CSV for 11 columns. Check the CSV contains headers box. Choose the Comma delimiterType, and put Data from the Download Document module in the CSV field.



9. Rename this module “Parse shipping manifest.”

10. Save the scenario and click Run once so you can see data from the CSV file in the next steps.

Introduction to universal connectors (cont.)

Get the Pokemon data using the universal connector.

11. Add an HTTP Make a Request module.

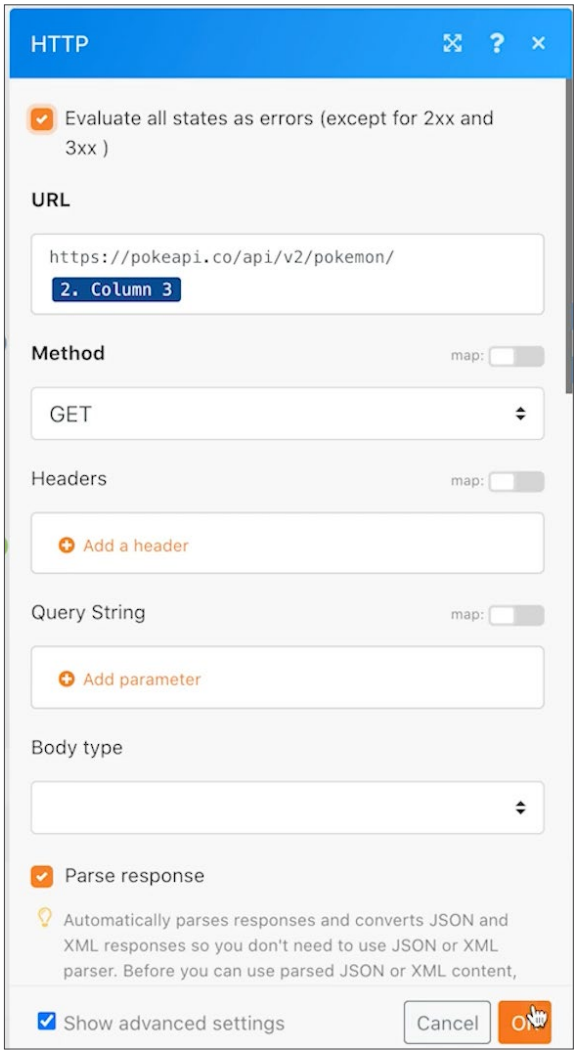
12. In the URL field use https://pokeapi.co/api/v2/pokemon/[Character], where [Character] is mapped to Column 3 from the Parse CSV module.

13. Select the Parse response check box.

14. Select Show advanced settings and then check the box next to "Evaluate all states as errors."

15. Click OK and rename the module "Get Pokemon info."

Your mapping panel should look like this:

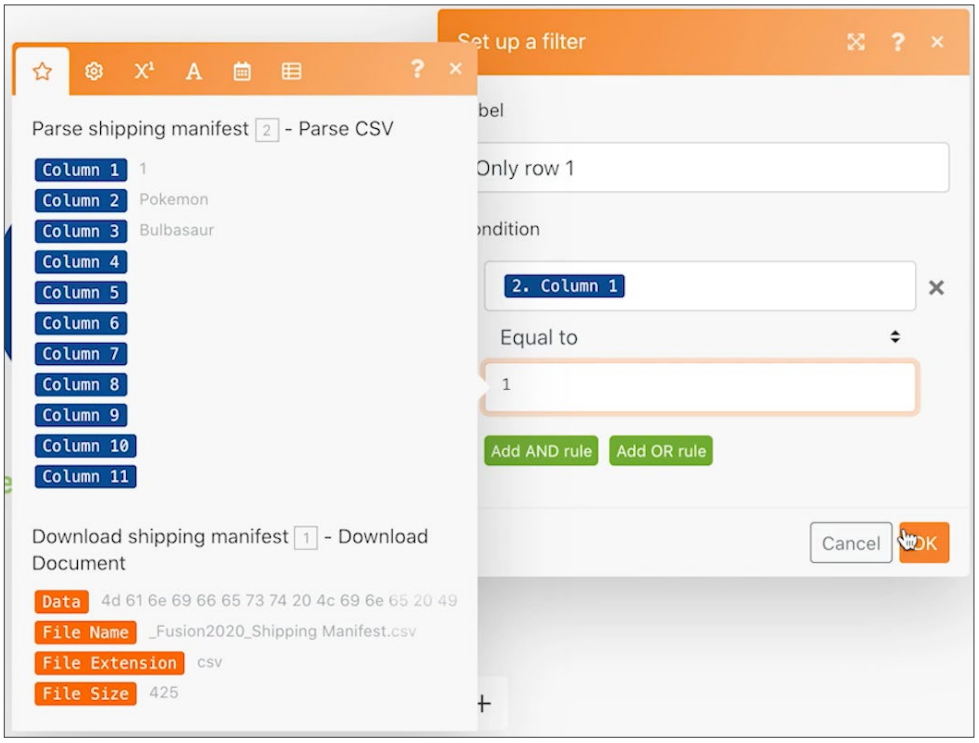


Introduction to universal connectors (cont.)

In this part of the exercise, you only want to process row 1 in the CSV file.

16. Add a filter before your Get Pokemon info module. Name it "Only row 1."

17. Set the condition to only allow ID number 1 to pass. ID number 1 is in row 1, and the ID field is in Column 1 in the CSV file.

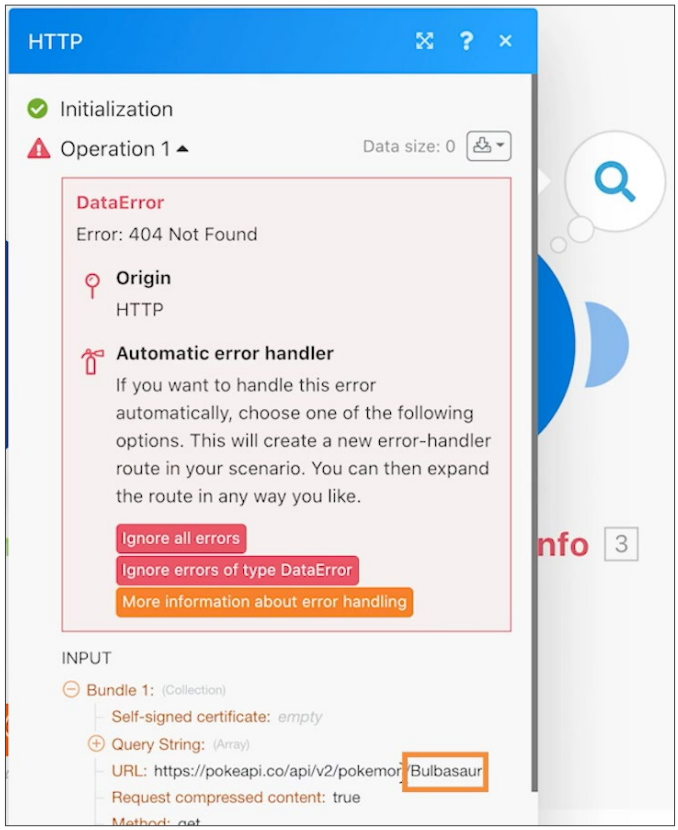


18. Save the scenario.

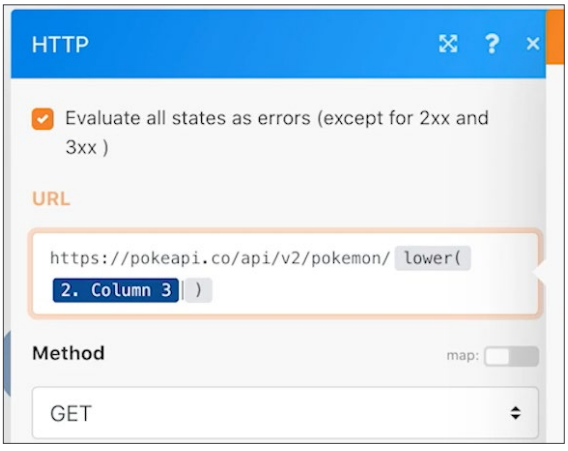
Introduction to universal connectors (cont.)

19. Click Run Once and observe the error message you receive in the HTTP Make a request module.

    - Notice in the input data URL field the character name is capitalized. This won't work for making that API call because character names need to be lowercase.



20. Use the mapping panel in the HTTP Make a request URL field to make the [Character] field all lowercase letters using the **lower** function.

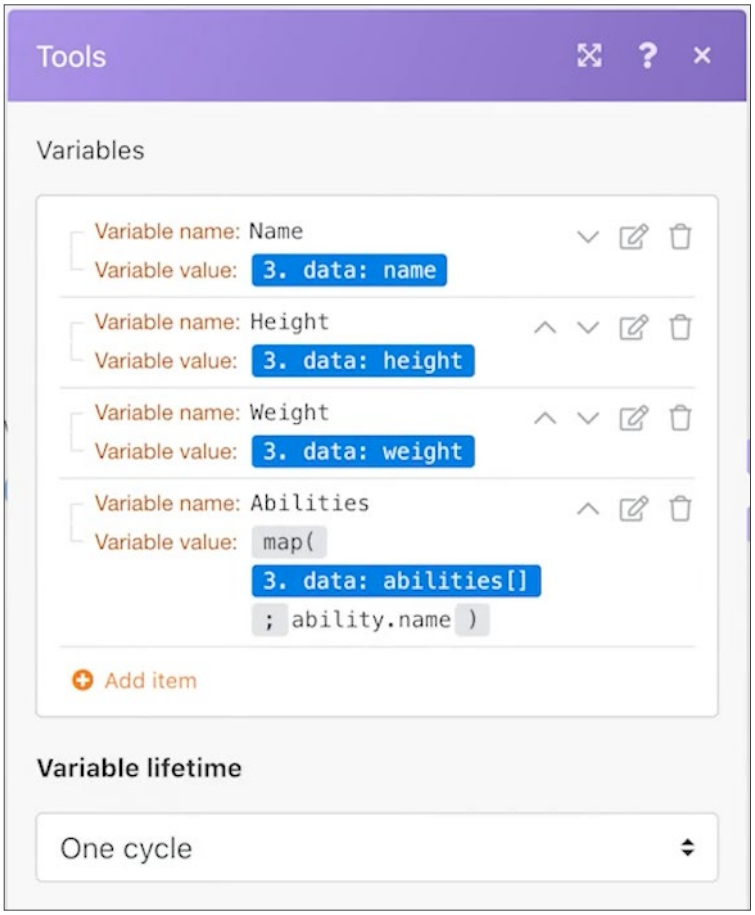


Introduction to universal connectors (cont.)

Map information back from the API using the Set multiple variables module.

21. Add the Set multiple variables module after Get Pokemon info. Map name, height, weight, and abilities.

22. Since the Abilities field is an array, remember to use the map function to access the name of each ability in the array.



Run the scenario without the filter to uncover another error.

23. To process all the rows in the CSV file, delete the filter named Only row 1:

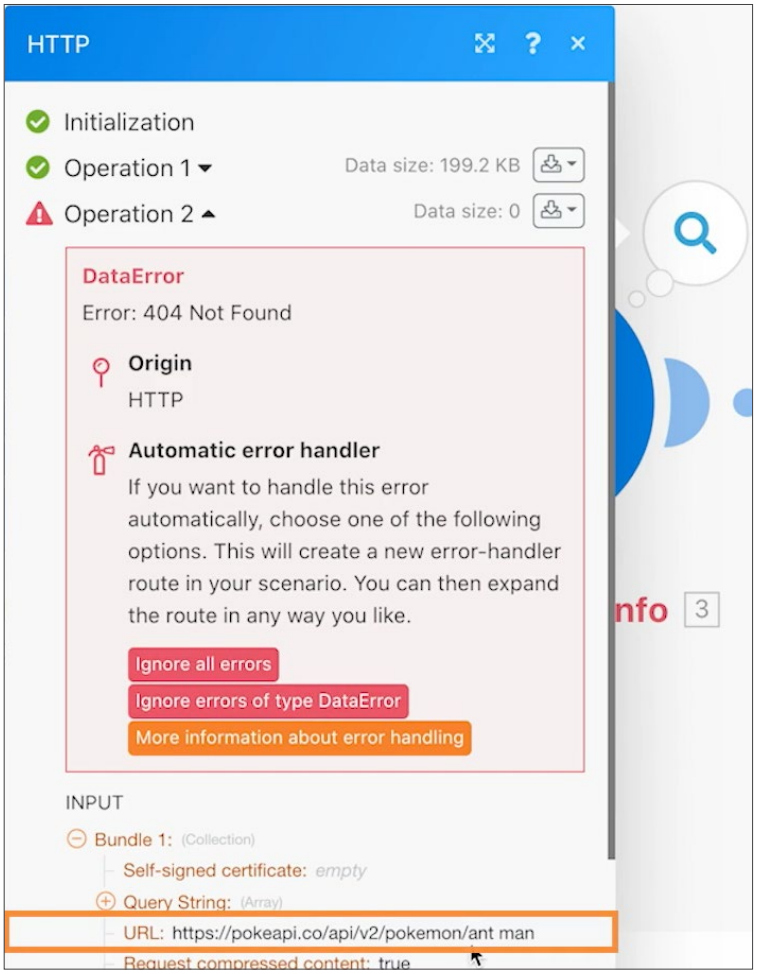
- Click the filter icon to edit it.
- Delete the filter label.
- Delete the Condition.
- Click OK.

Introduction to universal connectors (cont.)

24. Save the scenario and click Run once.

25. An error occurs in the Get Pokemon info module. You see a superhero character has been passed to the Pokemon API.

- In the Routers walkthrough, you'll see how to resolve this error by creating a separate path to process superheroes.

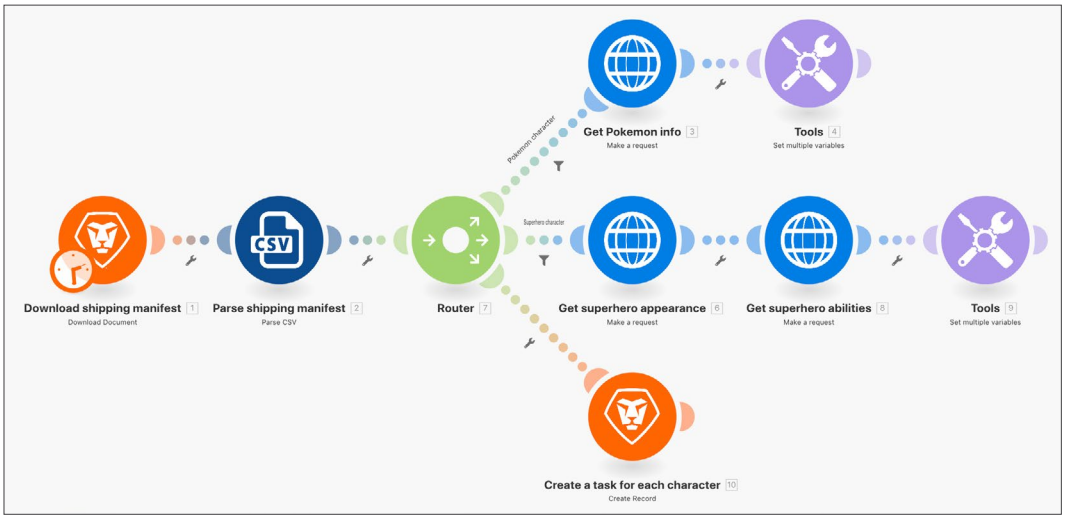


Routers

Understand the importance of routers and how they can be used to conditionally process different modules.

Exercise overview

Use a router to pass Pokemon vs. superheroes bundles down the correct path, then create a task for each character.

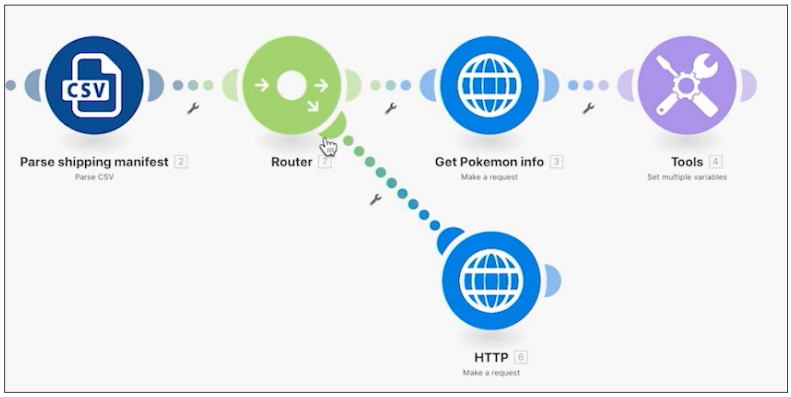


Steps to follow

1. Clone the Using universal connectors scenario from the previous exercise. Name it "Creating different paths using routers."

Create a new path for superheroes by cloning modules and adding a router.

2. Right-click the Get Pokemon info module and choose Clone. Once cloned, drag and connect it to the line between the new HTTP module and the Parse CSV module.
    - Notice how it automatically adds a router with two paths.



Routers (cont.)

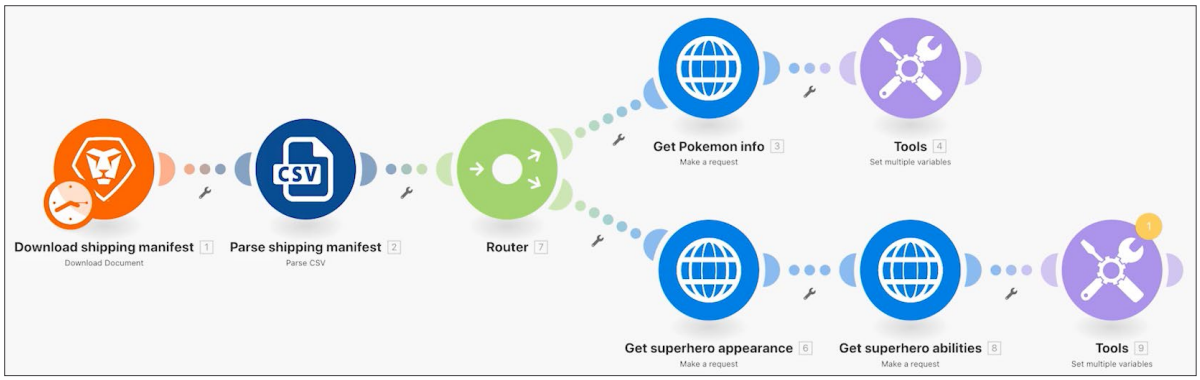
3. Name this module “Get superhero appearance.”

4. Clone this module, move the clone to the right, and name it “Get superhero abilities.”

5. Clone the Tools module and move it to the end of the second path.

6. Click the wand icon—the Auto-align button—in the toolbar.

Your scenario should look like this:



Next, you’re going to change the mapped values in the new cloned modules.

7. Go to **https://superheroapi.com/** and use your Facebook account to get an access token.

   - If you have trouble accessing your own superhero token, you can use this shared token: 10110256647253588. Please be considerate of how many times you call to the superhero API so this shared token continues to work for everyone.

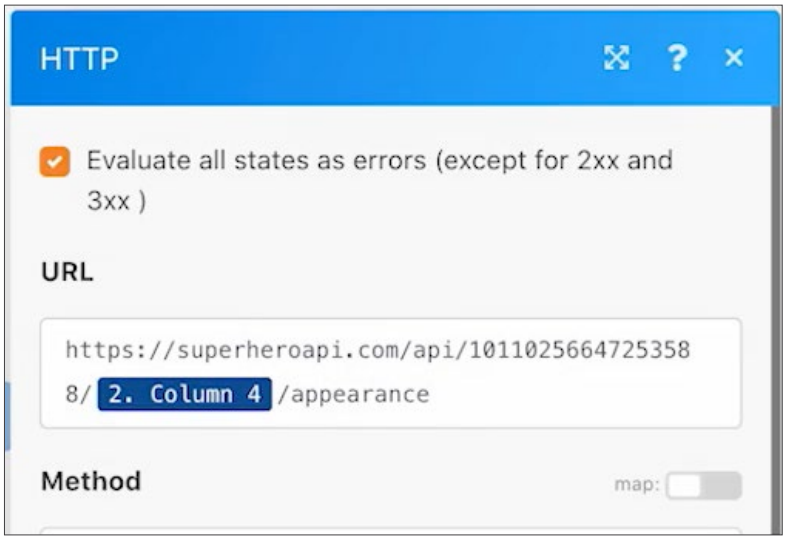
8. Open the settings for the Get superhero appearance and change the URL to https://superheroapi.com/api/[access-token]/332/appearance. Be sure to include your access token in the URL. Click OK.

9. Open the settings for the Get superhero abilities and change the URL to https://superheroapi.com/api/[access-token]/332/powerstats. Be sure to include your access token in the URL. Click OK.

10. Right-click each superhero module and select Run this module only. This will generate the data structure you need to see for mapping.

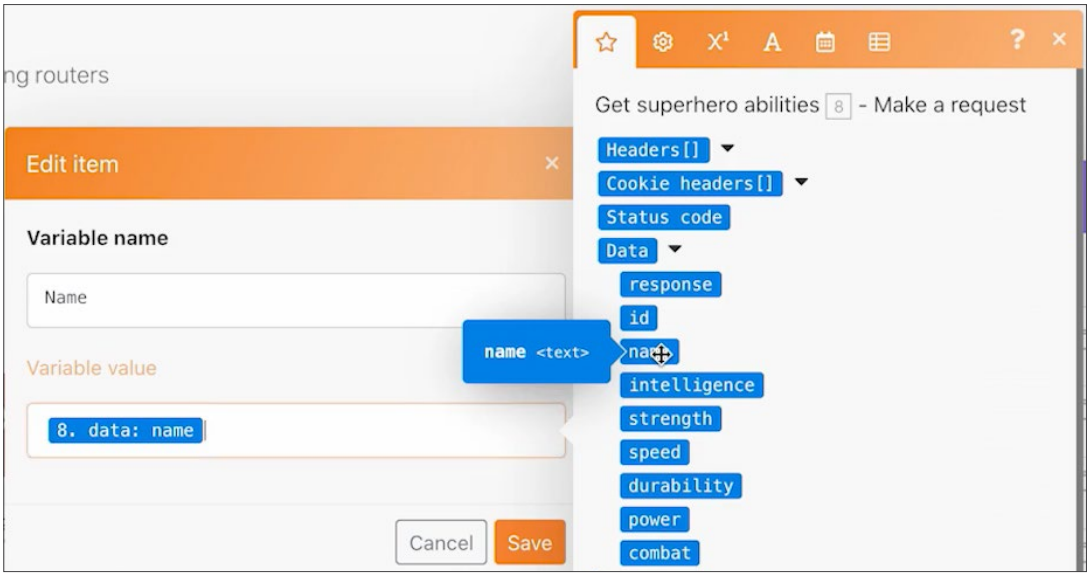
Routers (cont.)

11. After you run both, change the number "332" in each URL field to Column 4 mapped from the Parse CSV module.



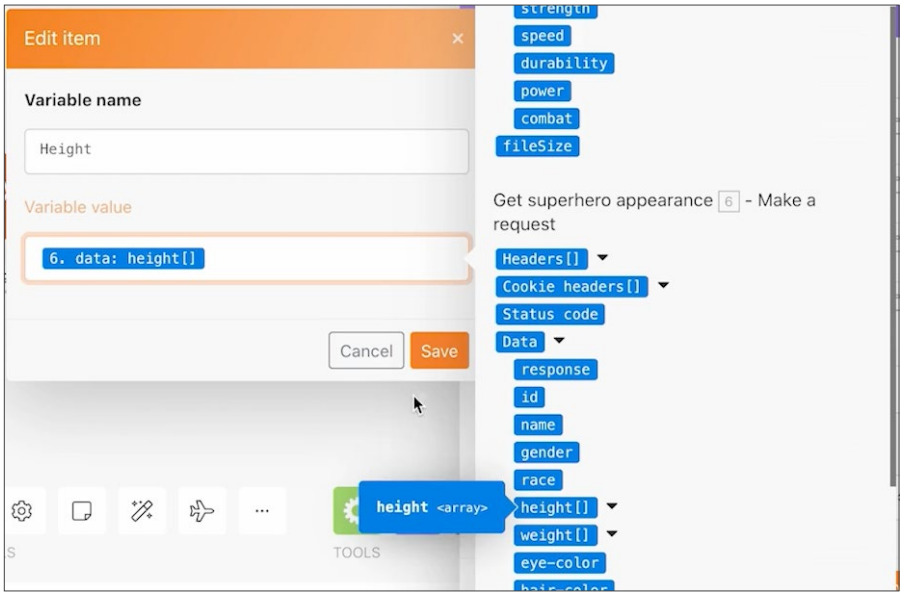
Now you can click into the Set multiple variables module in the superhero path and update the name, height, weight, and abilities.

12. Update the Name and Abilities fields from the Get superhero abilities module—Module 8.

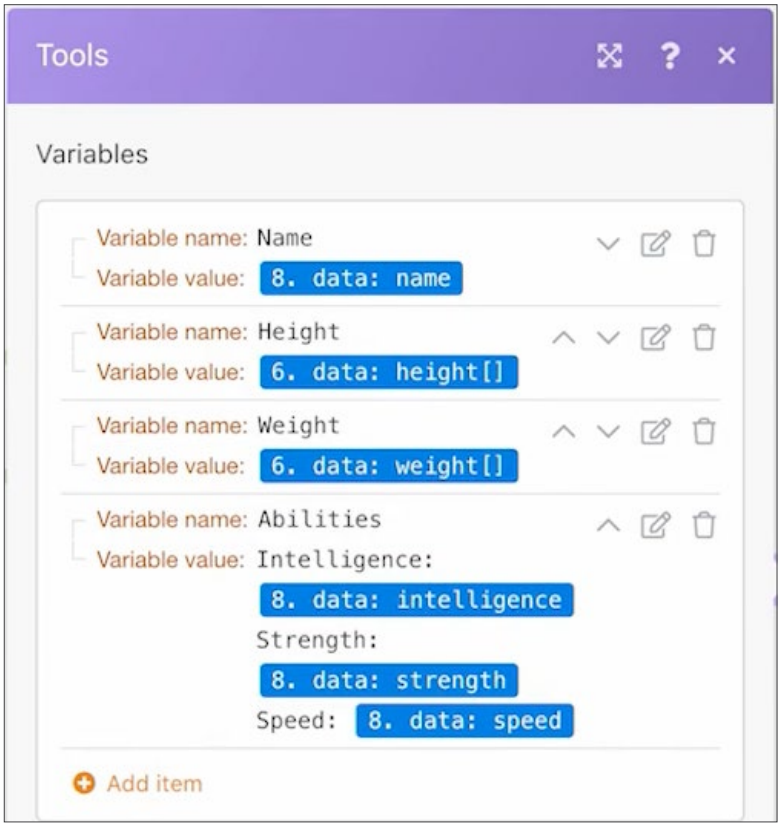


Routers (cont.)

13. Update the Height and Weight fields from the Get superhero appearance module—Module 6.



When you're done, your variables should look like this. Note that the module numbers appear in the field values.



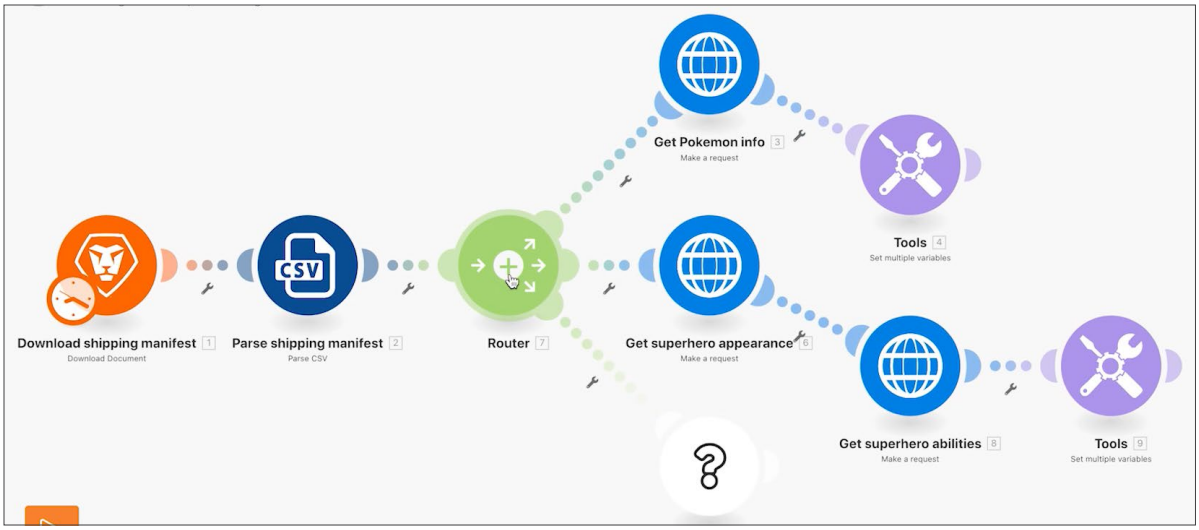
Routers (cont.)

14. Click OK, then save your scenario.

Create another path to create a task per character.

15. In Workfront, create an empty project. Name it “Shipping Manifest Project” and copy the project ID from the URL.

16. Return to Workfront Fusion and click in the center of the router to create another path.



17. Click in the center of the empty module that appears and add a Create record module from the Workfront app.

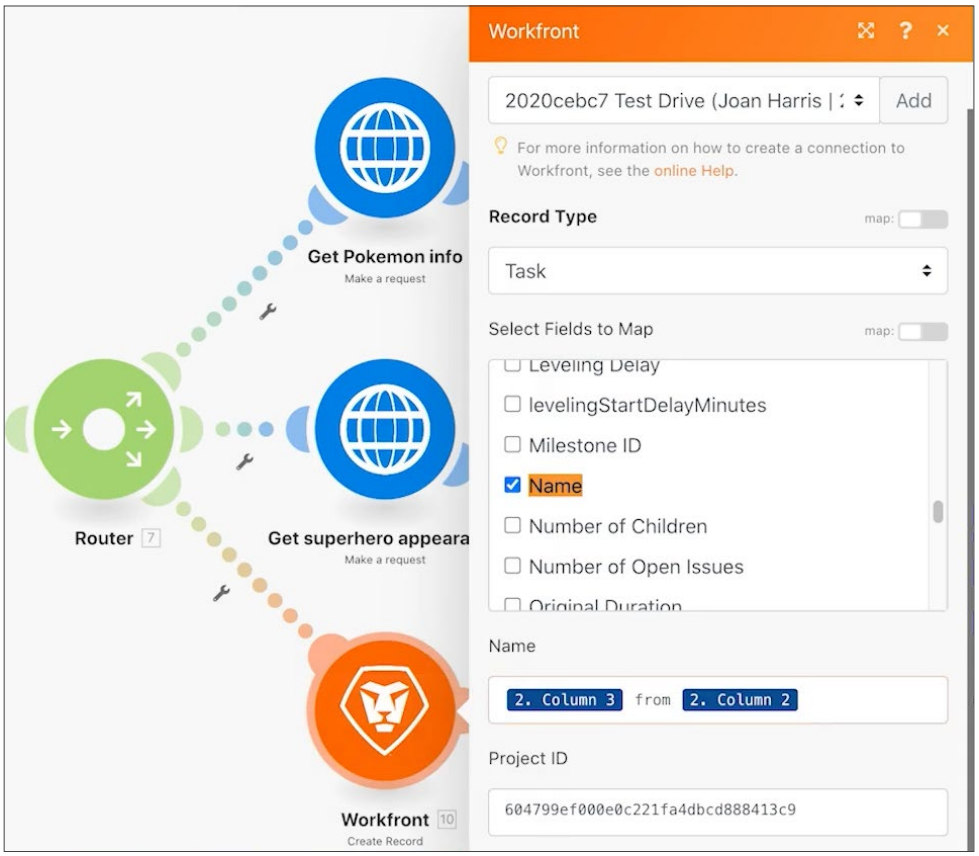
18. Set the Record Type to Task and select Project ID from the Fields to Map section.

19. Paste the project ID you copied from Workfront into the Project ID field.

20. Now, select the Name field from the Fields to Map section.

Routers (cont.)

21. Name the task "[Character] from [Franchise]," taking the character name and the franchise name from the CSV file. Column 3 is the character name and column 2 is the name of the franchise.



22. Click OK, and rename this module to "Create a task for each character."

Add filters so the scenario can run without errors. You want only Pokemon characters to go down the top path, only superhero characters to go down the middle path, and all characters to go down the bottom path.

23. Click the dotted line to the left of the Get Pokemon info module to create the first filter. Name it "Pokemon character."

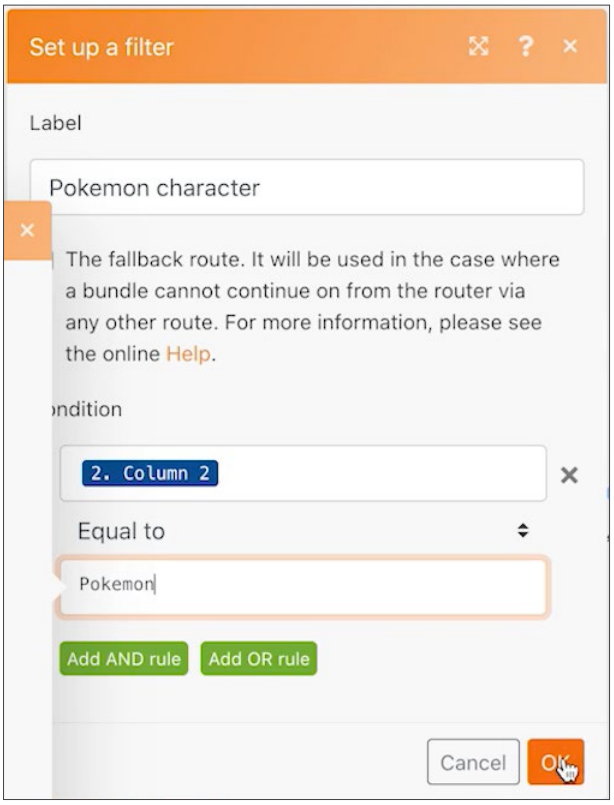
24. For the condition, only allow records where the franchise (Column 2) is equal to "Pokemon." Choose the text "Equal to" operator.

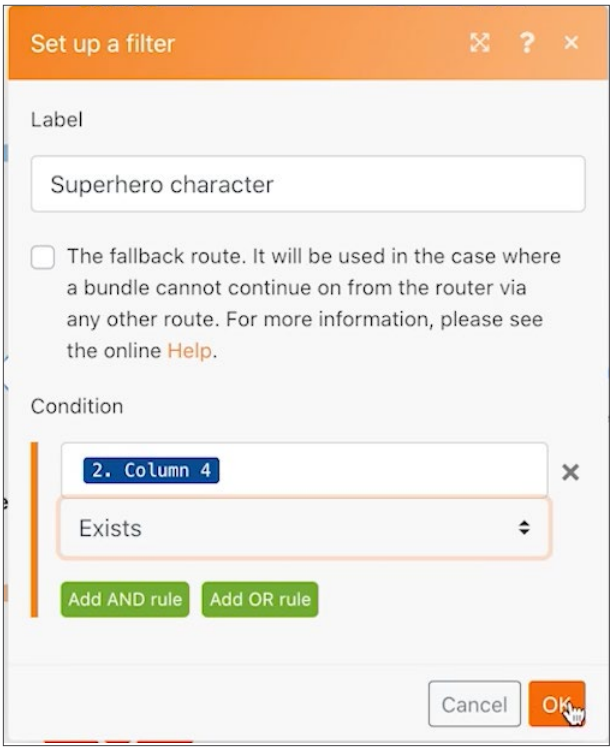
25. Click the dotted line on the left of the Get superhero appearance module to create the next filter. Name it "Superhero character."

26. Because superheroes can come from various franchises, use the Superhero ID field (Column 4) to determine if a character is a superhero or not.

Routers (cont.)

Your filters should look like this:





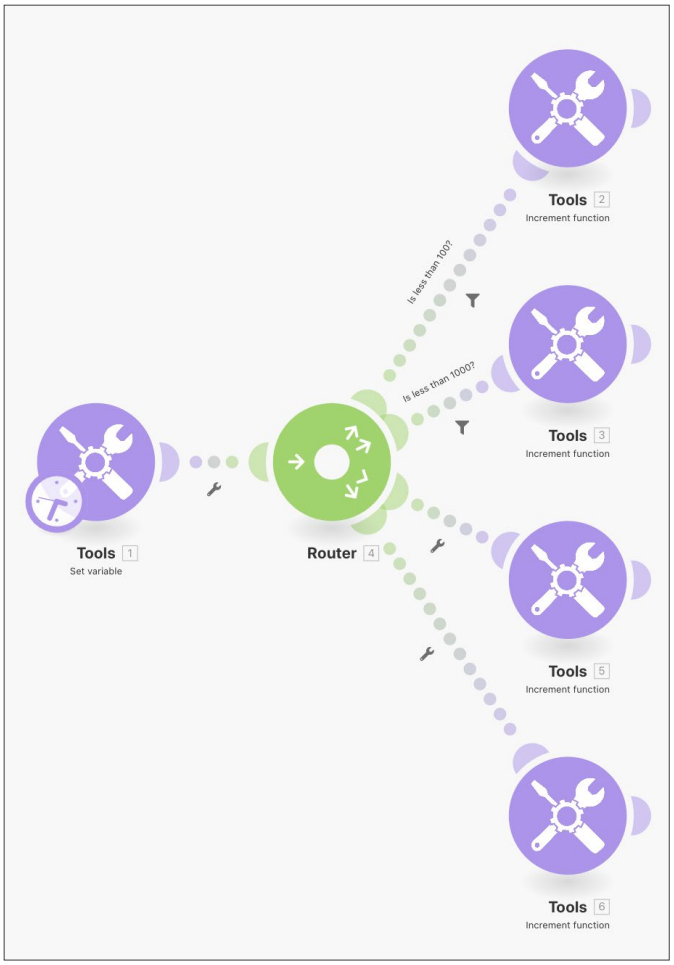
27. Save the scenario and click Run once. Use the execution inspectors to verify all operations were successful, and check tasks that were created in your Workfront project.

Routing patterns

Reinforce your concept of routing and fallback routes without actually dealing with any other APIs.

Exercise overview

Use the Set Variable module to send a number through multiple paths to see how filters and fallbacks behave when routing.

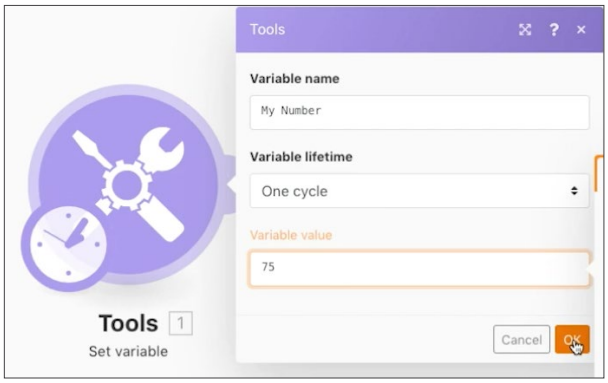


Routing patterns (cont.)

Steps to follow

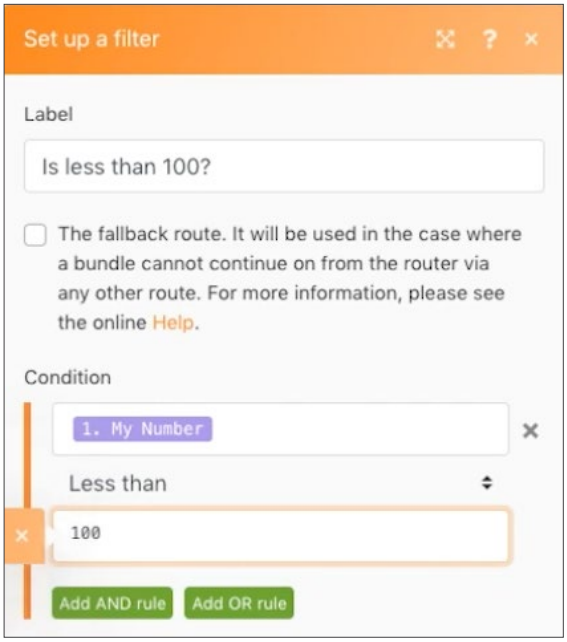
1. Create a new scenario and call it “Routing patterns and fallbacks”.

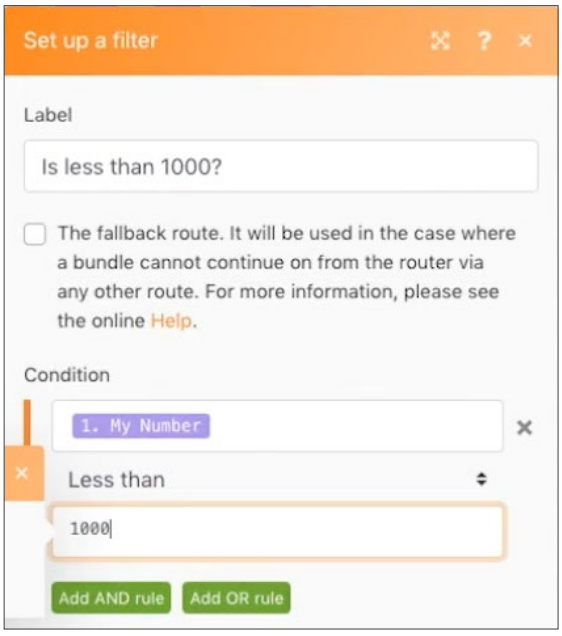
2. For the trigger, add the Set Variable tool module. Put “My Number” for the Variable name, leave the Variable lifetime as One cycle, and set the Variable field to “75.”



3. Add another module and choose the Router module. For both paths, choose the Increment function tool and click OK without making any changes for each.

- For the first path, create a filter, name it “Less than 100,” and set the condition to [My Number] Less than 100.
- For the second path, create a filter, name it “Less than 1000,” and set the condition to [My Number] Less than 1000. Make sure you use the Numeric operator for both.



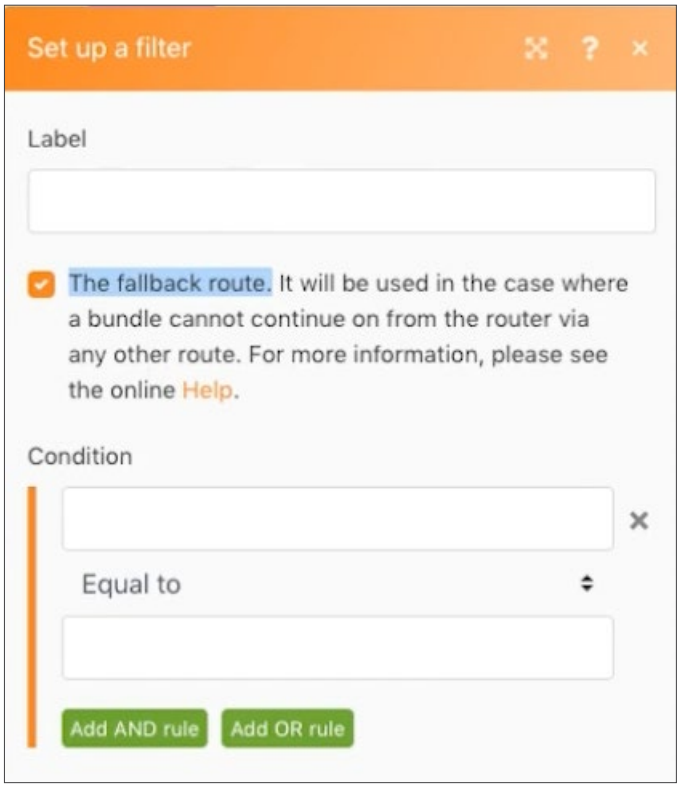


Routing patterns (cont.)

4. Click Run once and watch the bundle pass down the “Less than 100” path.

5. Then change the Set Variable module field to 950 and Run once again. Watch it run down the second path.

6. Click the router and add one more path. Add the Increment function tool module. For the filter, click “The fallback route” checkbox. Notice how the arrow pointing to that path changes to a caret, indicating it’s the fallback route.



7. Change the Set variable number to 9500 and Run once. Because the number is not less than 100 or less than 1000, the bundle travels down the fallback route.

If you add one more path with an Increment function tool module, but set no filter, what will happen when you click Run once again? Will a bundle ever go down the fallback route with the fourth route added?

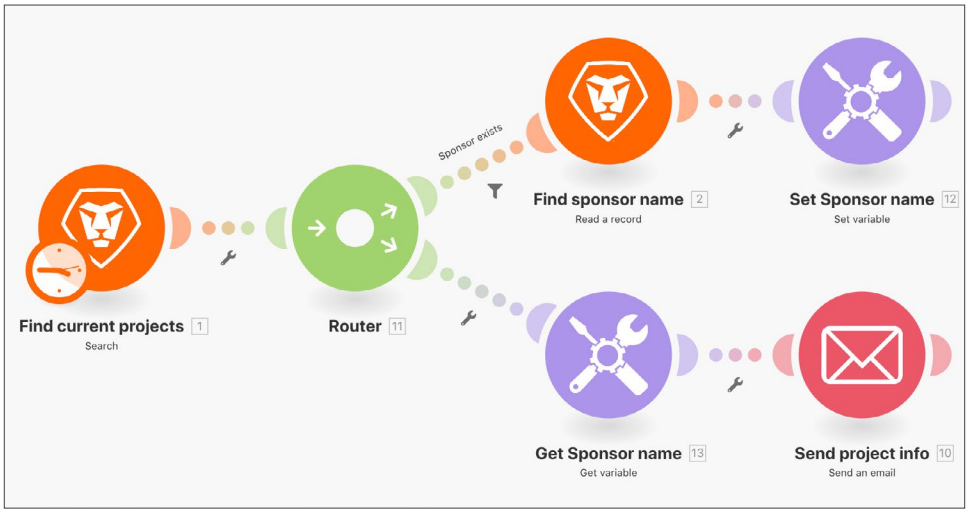
- No, because with no filter set, every bundle will always go down this path instead of the fallback route.

Set/Get variables

Learn how to use the Set and Get Variable modules to use the fields available in one path in a different path.

Exercise overview

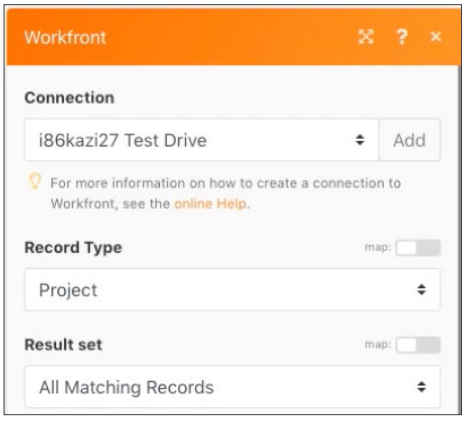
Look up information about a project in Workfront and send an email with related information.

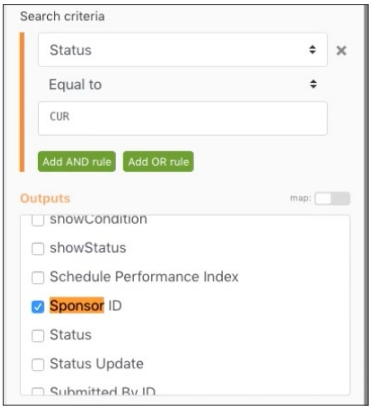


Steps to follow

1. Create a new scenario and name it "Sharing variables between routing paths."

2. For the trigger, select the Search module in the Workfront app.





   - Set the Record Type to Project.
   - For the Result set, choose All Matching Records.
   - For Search criteria, set it to Status Equal to CUR.
   - For Outputs, choose ID, Name, Description, and Sponsor ID.

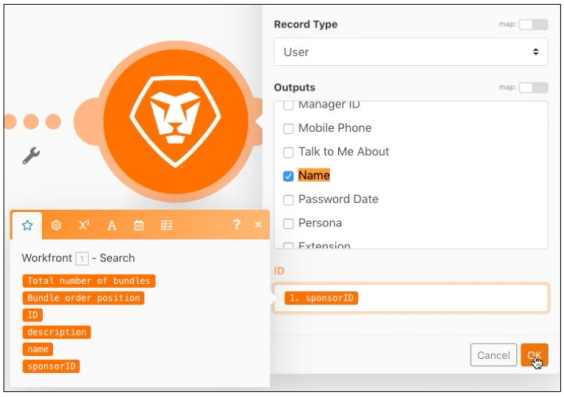
Set/Get variables (cont.)

3. Click OK and rename this module “Find current projects.”

4. Add another module and select the Workfront Read a record module.
   - For the Record Type choose User.
   - For Outputs choose Name.
   - Map the Sponsor ID from the Search module to the ID field.

5. Click OK.

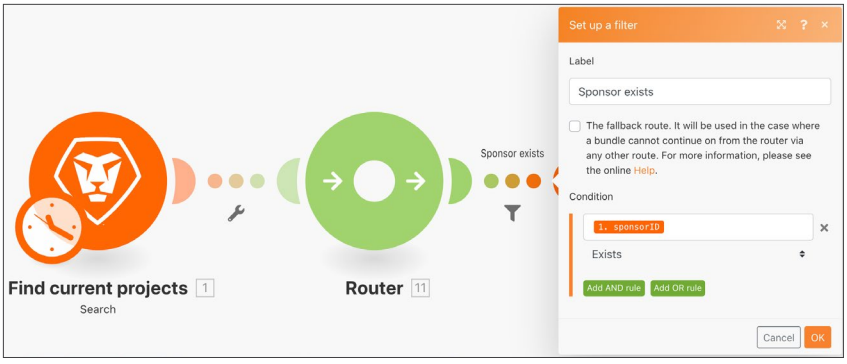
6. Rename the module “Find sponsor name.”



7. Save the scenario and click Run once.
   - If you receive an error on the Read a record module, it’s likely due to the Search module finding a project without a sponsor listed.

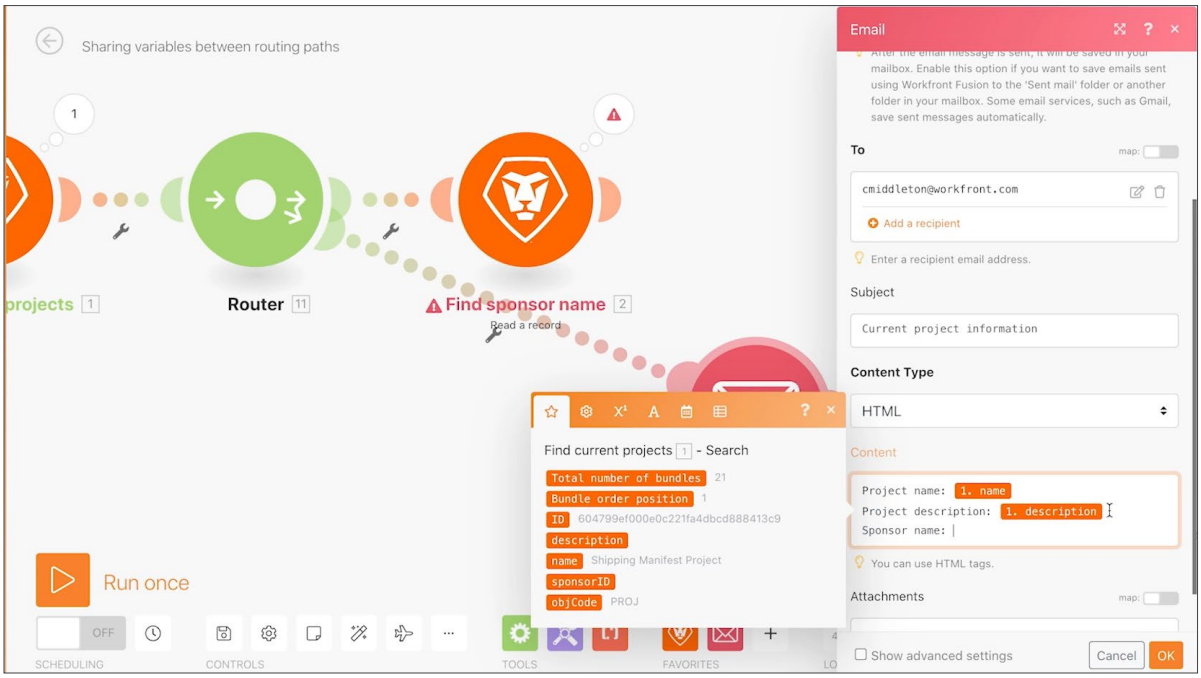
To avoid this error, create two paths: one for projects that have a sponsor ID and one for projects that don’t.

8. Add a router between the two modules by clicking the wrench icon between the router and the Read a record module. Set up a filter named “Sponsor exists” and set the Condition to Sponsor ID Exists.



Set/Get variables (cont.)

9. Click the router to create another path. Add a Send an email module from the Email app.
   - Put your own email address in the To field.
   - In the Subject field, type "Current project information."
   - In the Content field, put the project name, description, and sponsor.
   - You cannot pull the sponsor name output from the Read a record module. You can only access the sponsor ID from the search module before the router. You'll need to find a way to access the sponsor name from the other router path.



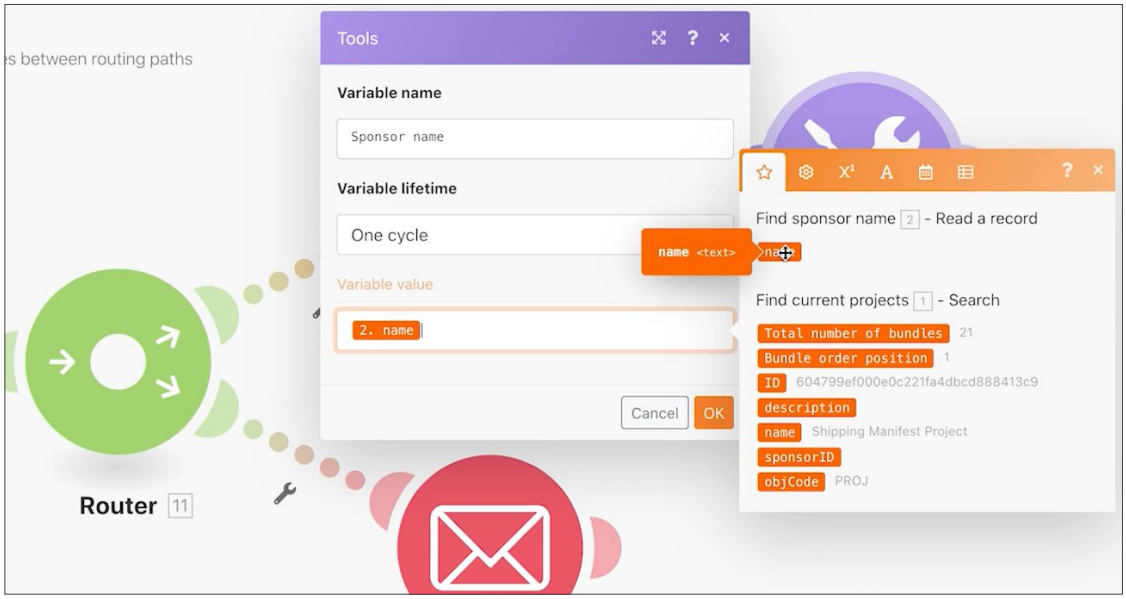
10. Click OK for now, and rename this module "Send project info."

Use Set/Get variables to share data between different paths.

11. After the Find sponsor name module, add a Set variable tool module.
    - Put "Sponsor name" as the Variable name.
    - Leave the Variable lifetime at One cycle.
    - Map the field to the name output from the Find sponsor name module.

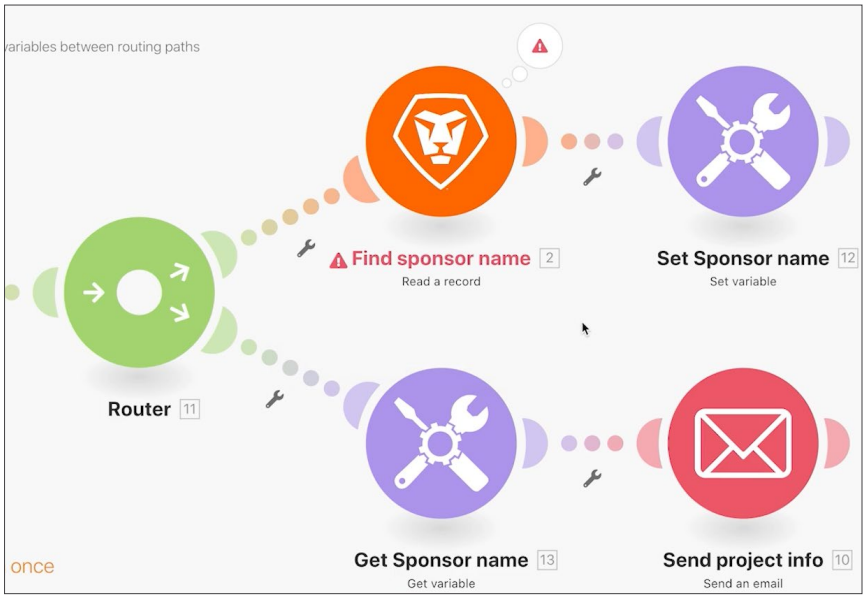
Set/Get variables (cont.)

12. Click OK, then rename the module "Set Sponsor name."



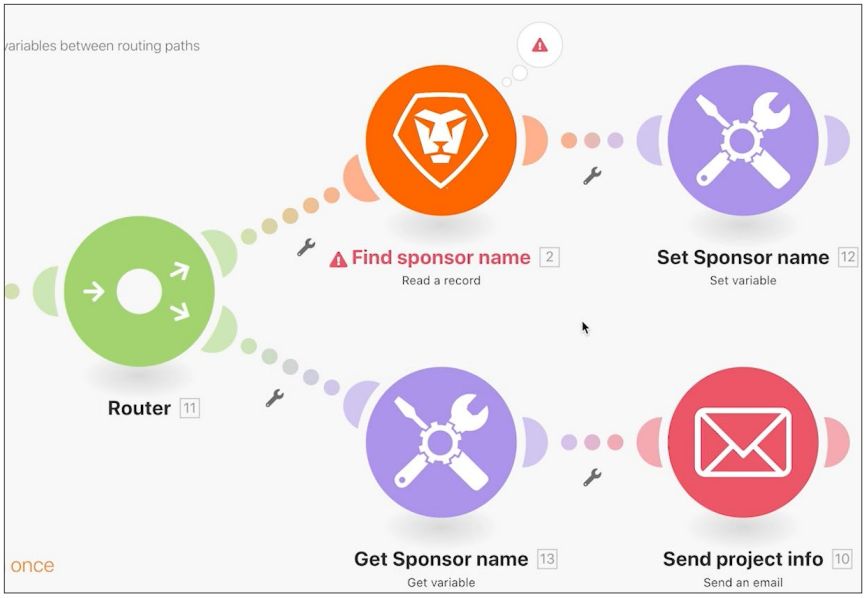
13. Next, right-click between the router and the Send an email module to add a Get variable tool module. Enter "Sponsor name" in the Variable name field.

14. Click OK. Rename the module "Get Sponsor name."



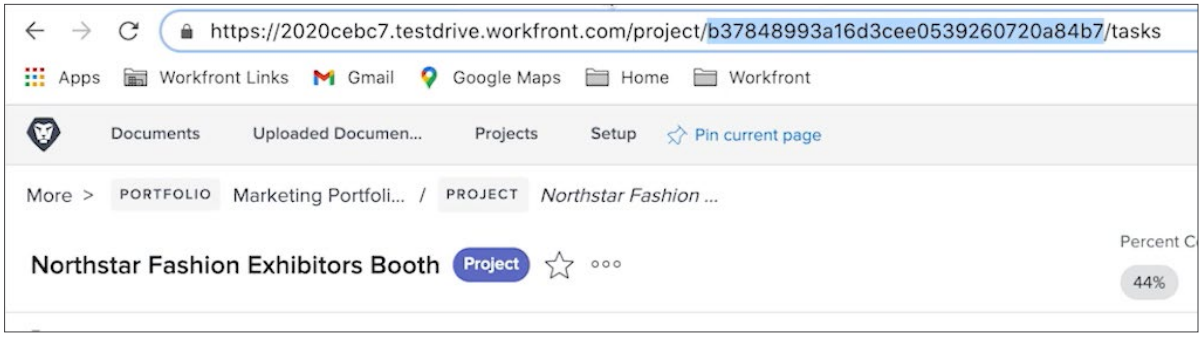
Set/Get variables (cont.)

15. Go back into the Send an email module and map the value from the Get Sponsor name module into the content field. Click OK.



Before you test the scenario, we recommend restricting the number of projects you process to avoid getting a flood of emails.

16. Go to your Workfront test drive and locate the Northstar Fashion Exhibitors Booth project. This is a current project that has a sponsor. Copy the project ID from the URL.

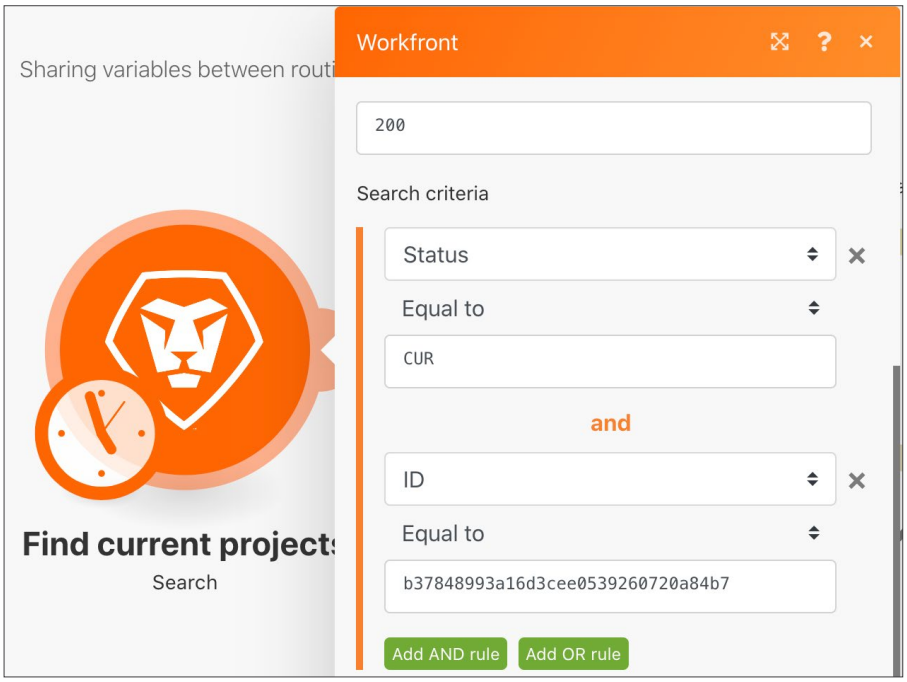


17. In your scenario, click the Find current projects module. Add another condition to the search criteria by clicking the green "Add AND rule" button. Specify that the ID must equal the project ID you copied. Click OK.

Set/Get variables (cont.)

18. Save your scenario and click Run once.

19. Review the execution inspectors and the email you receive.

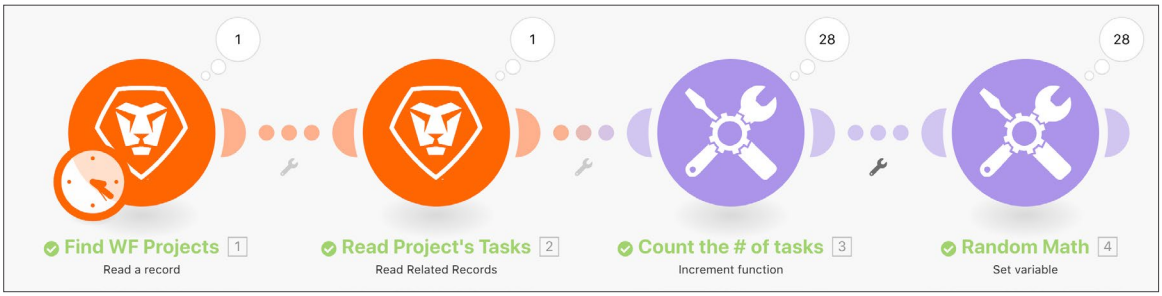


Introduction to iterators

Learn to use iteration-type apps and perform actions on each bundle of information.

Exercise overview

Look at a specific project in Workfront, then look at all the tasks within that project. You will use the increment tool module to count the number of tasks within the project. Finally, you'll use the Set variable module to subtract the Number of Children from the Number of Open Issues to produce a numeric value for each of the task bundles.



Steps to follow

Read a project and related tasks.

1. Start a new scenario. Name it "Introduction to iteration."
2. Choose Workfront as the trigger module, Read a record.
3. For Record Type, choose Project.
4. For Outputs, choose ID, Name, and Description.
5. In the ID field, put the project ID of the Northstar Fashion Exhibitors Booth project from your Workfront test drive instance.
6. Rename this module "Find WF Projects."
7. Add another Workfront module to read the tasks related to this project. Choose the Read Related Records module.
8. For Record Type, choose Project.

Introduction to iterators (cont.)

9. For the Parent Record ID, choose the ID from the Read a record module.

10. For Collections, select Tasks.

11. For Outputs, select ID, Name, Description, Number of Children, Number of Open Issues, and Work.

12. Rename this module "Read Project's Tasks."

13. Save the scenario, then click Run once to see the outputs.

    - Click on the execution inspector and you see one bundle as input (the project) and 28 bundles as output (the tasks).

Count and process iterated bundles.

14. Add another module after Read Related Records. Choose an Increment function tools module.

    - Leave the Reset a value field as Never and click OK.

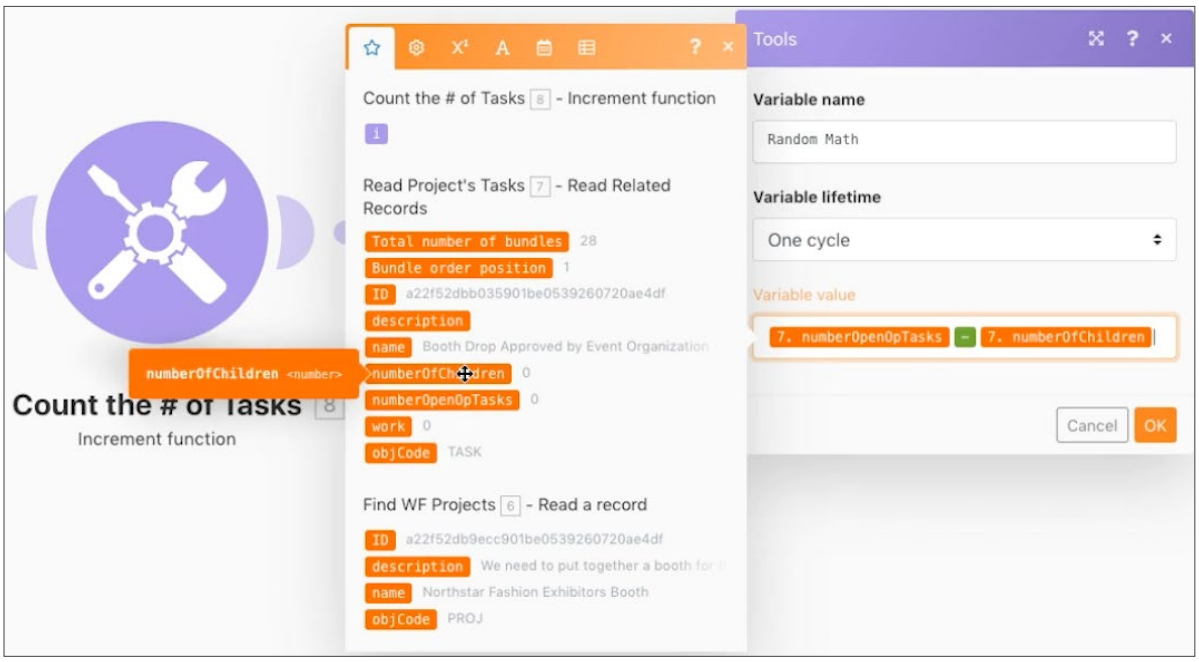
15. Rename this module "Count the # of tasks."

16. Add a Set variable module. Set the Variable name to "Random Math."

17. In the Variable value field, subtract the number of open children from the number of open opTasks.

Introduction to iterators (cont.)

It should look like this:



18. Rename this module "Random Math."

19. Save the scenario and click Run once.

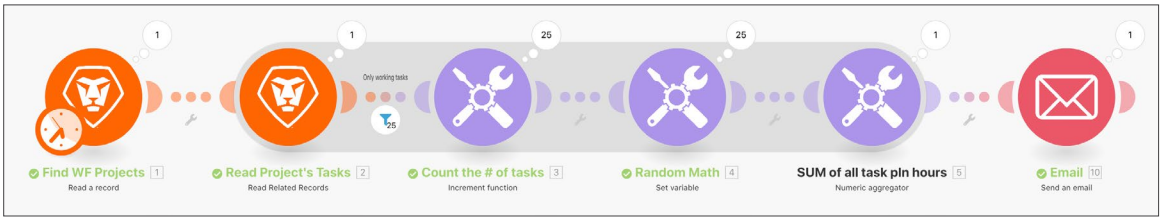
For each of the tasks produced by the Read Related Records iterator module, Workfront Fusion performed 28 executions. These 28 bundles will continue to be processed throughout the scenario unless an aggregator is added to close the loop.

Aggregation

Learn to aggregate multiple bundles of information into a single value.

Exercise overview

Using the "Introduction to iteration" scenario you built in the last exercise, aggregate the planned hours on every working task in the project and send an email to yourself with that information.



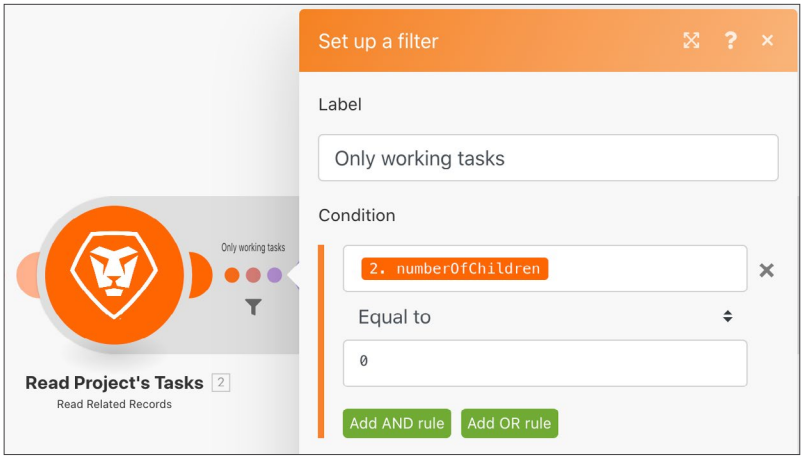
Steps to follow

Add a filter and SUM the planned hours.

1. Clone the "Introduction to iteration" scenario you created in the previous exercise, and name it "Introduction to aggregation."

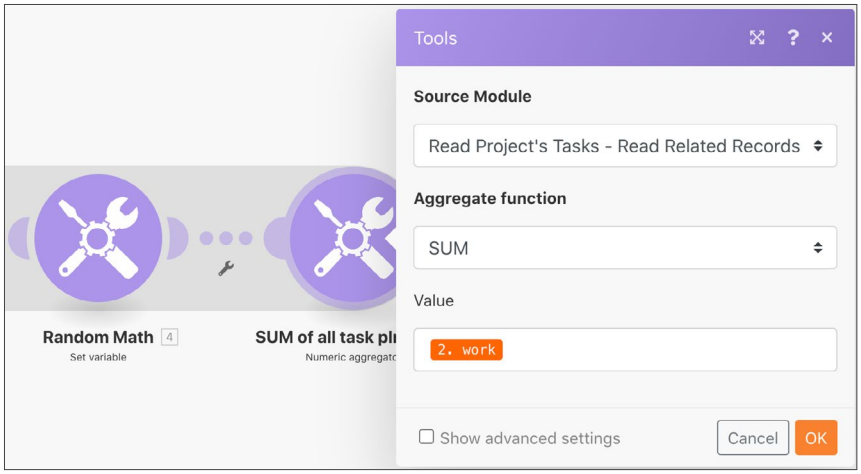
2. Add a filter between the Read Project's Tasks module and the Count the # of tasks module. Name the filter "Only working tasks."

3. Set the condition to Number of Children <Numeric Operator: Equal to> 0.

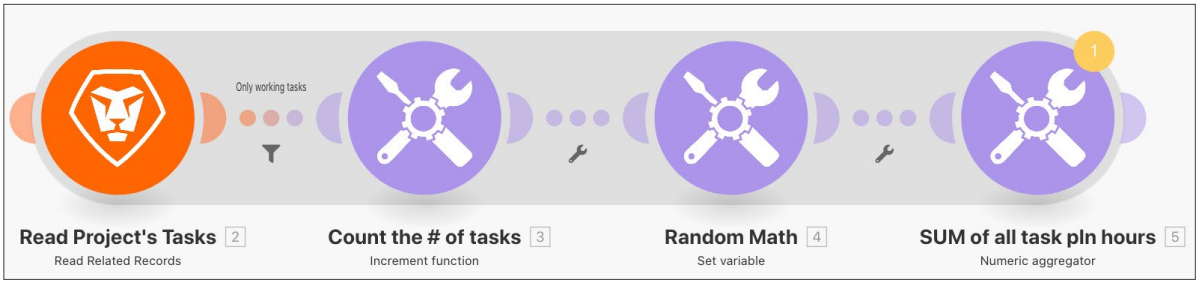


Aggregation (cont.)

4. After the Random Math module, add a Numeric Aggregator tool module.
5. Set the source module to Read Project's Tasks.
6. Set the Aggregate function to SUM.
7. Set the Value to the Work field from the Read Project's Tasks module.
8. Rename this module "SUM of all task pln hours."



Note the shadow that shows that the aggregation ends the iteration.



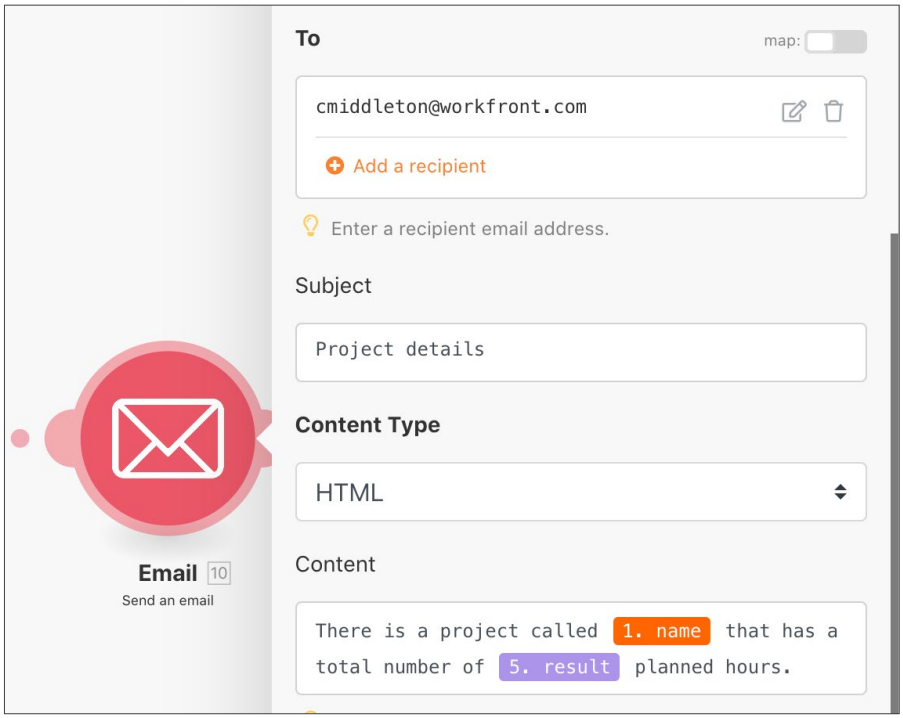
Send an email with aggregated hours.

9. Add a Send an email module from the Email app, after the numeric aggregator.
10. Send the email to yourself.

Aggregation (cont.)

11. Subject line is "Project details."

12. In the Content field, put "There is a project called [project name] that has a total number of [result] planned hours." The "[project name]" is taken from the Read a Record module and "[result]" is taken from the aggregator module.



13. Save and Run once. Find the email in your inbox.

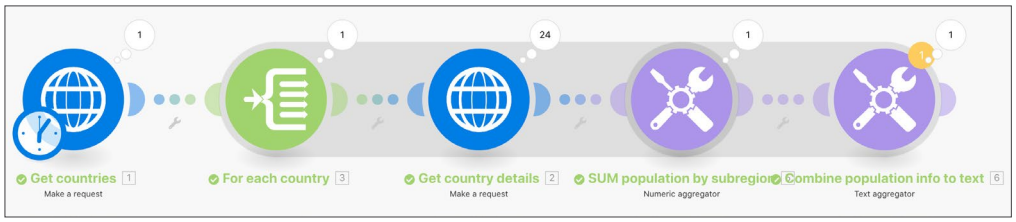
Within the iteration, the individual bundles can be accessed. But outside of the iteration, in the Send an email module, only aggregated fields can be accessed.

Advanced aggregation

Understand how to use groupings when aggregating.

Exercise overview

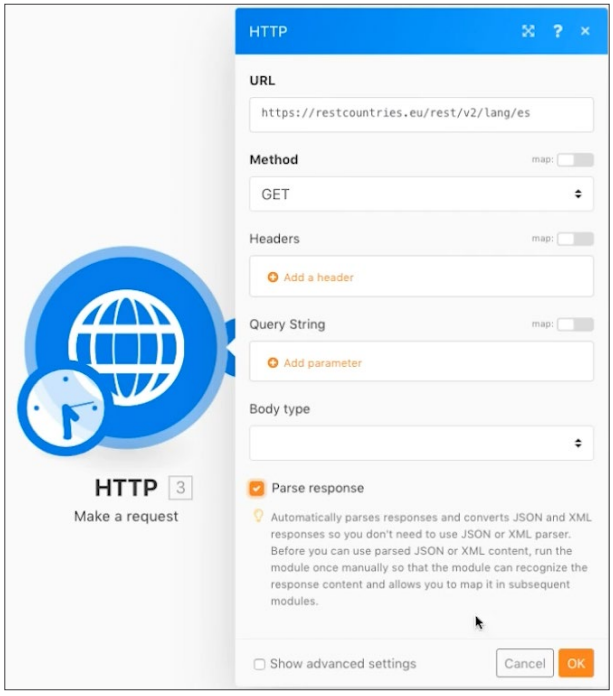
Call a web service to return details about multiple countries and identify the total population of all countries, grouped by subregion.



Steps to follow

Get country details.

1. Create a new scenario and name it "Advanced aggregation."
2. Set the trigger module to an HTTP - Make a request module.
3. Use this URL, https://restcountries.eu/rest/v2/lang/es, which gives you a list of all countries where Spanish is spoken.
4. Leave the Method as Get.
5. Click the Parse response checkbox.
6. Rename this module "Get Countries."
7. Click Save and Run once.



Advanced aggregation (cont.)

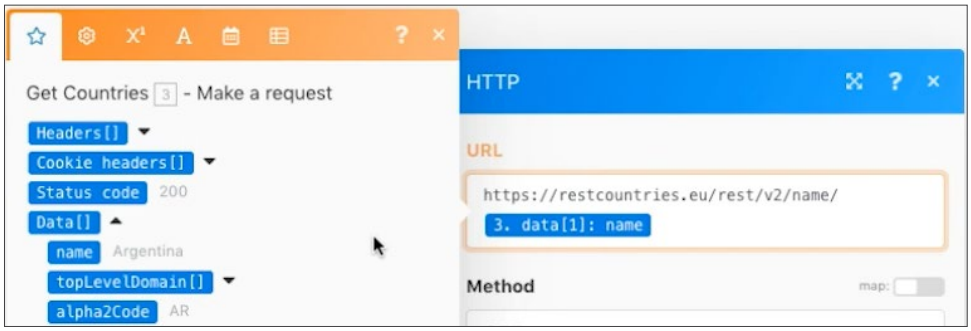
The output is a single bundle, but it comes in an array with 24 collections, one for each Spanish speaking country.

You need to gather subregion information for each of the countries, so you'll need to make an additional HTTP request.

8. Add another request to get subregion information. It will only return the first country, but that's OK for now. Add another HTTP Make a request module and use the URL https://restcountries.eu/rest/v2/name/.

9. To get the name of the first country, go to the mapping panel and click Data, then Name in the array. The [1] in the data field means it will return the first item in the array.

- Click the number and change the index if needed, but in this case you just want the first item.



10. Check Parse response in the mapping panel, then click OK.

11. Rename this "Get Country Details."

12. Click Save, then Run once.

- The output is information for a single country.

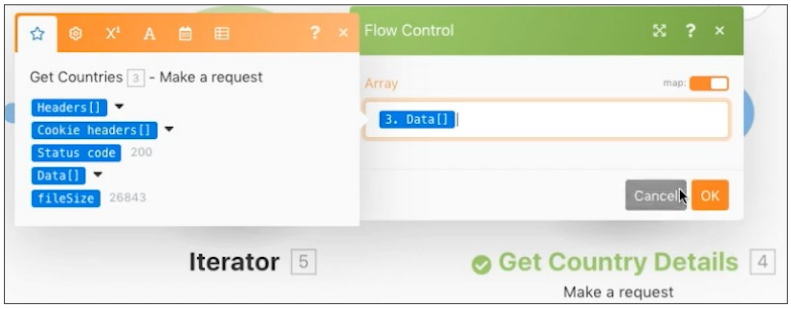
Advanced aggregation (cont.)

13. To get the other countries, you need to iterate through the array. Add an iterator, which takes a list of things and outputs a bundle for each item on the list.

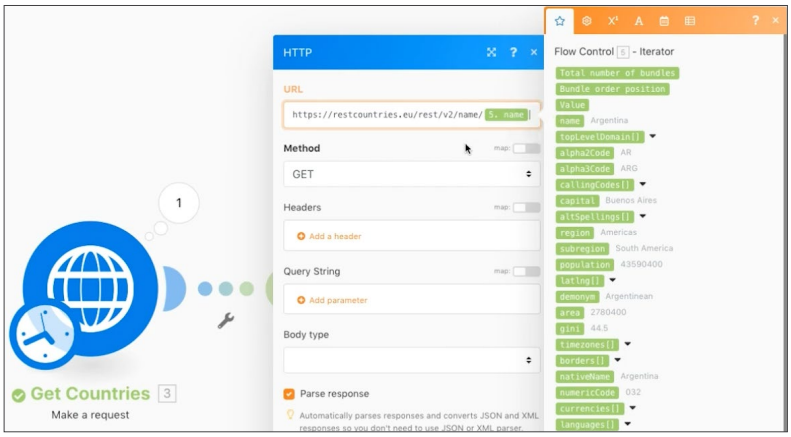
Add the iterator and aggregator.

14. Right-click between the HTTP modules and add the Iterator Flow Control module.

15. In the Array field, select Data from the Get Countries module.



16. In the Get Country Details module, update the URL field to take the name field from the iterator instead of from the Get Countries module.



17. Now add a numeric aggregator after Get Country Details to group and sum the populations.

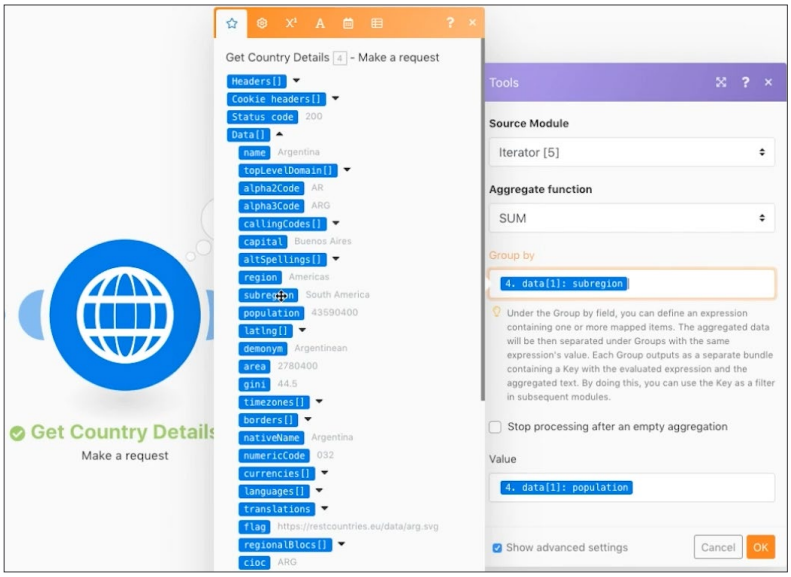
18. The source module is the iterator module.

19. The aggregate function is SUM.

20. The value is [data:population] from the Get Country Details module.

Advanced aggregation (cont.)

21. Click the Show advanced settings option at the bottom and group by [data:subregion] from the Get Country Details module.

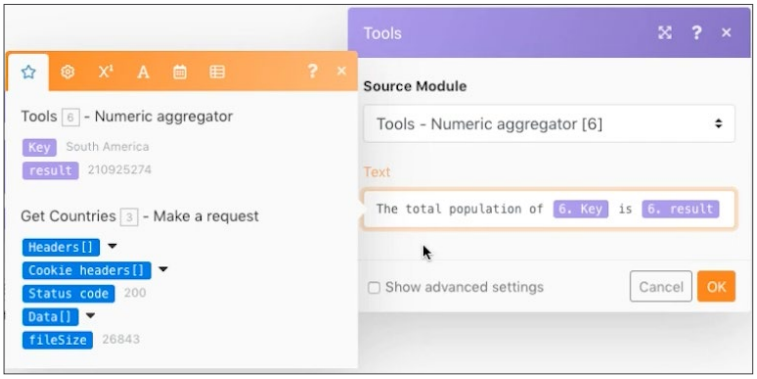


Finish with a text aggregator to aggregate what you grouped within the numeric aggregator.

22. Add a text aggregator to the end.

23. The source module is the numeric aggregator.

24. In the Text area, insert "The total population of [KEY] is [result]."



25. Save and Run once.

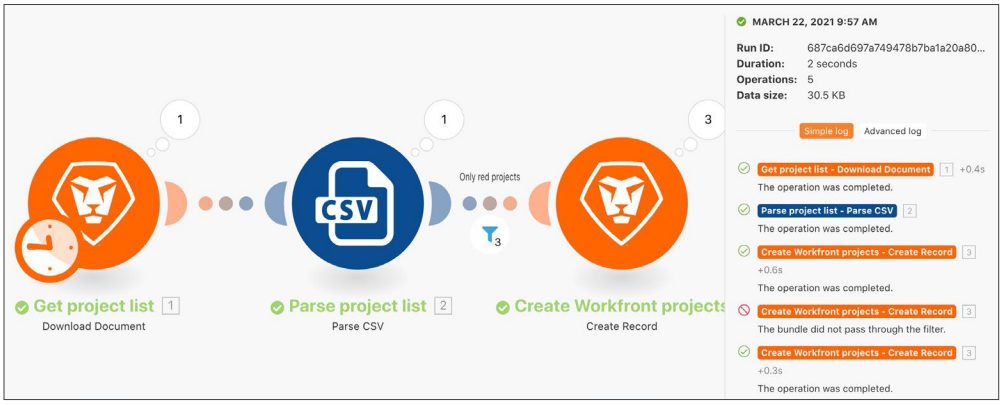
- Review the output from the final module.

Execution history

Review and audit details about past executions and scenario configurations.

Exercise overview

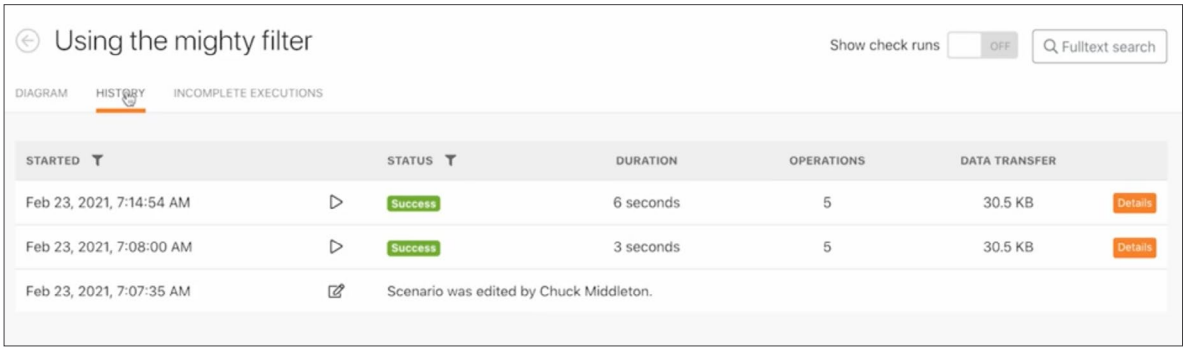
Review the execution history for the "Using the mighty filter" scenario to understand what happened when executions occurred and how they were structured when they were run.



Steps to follow

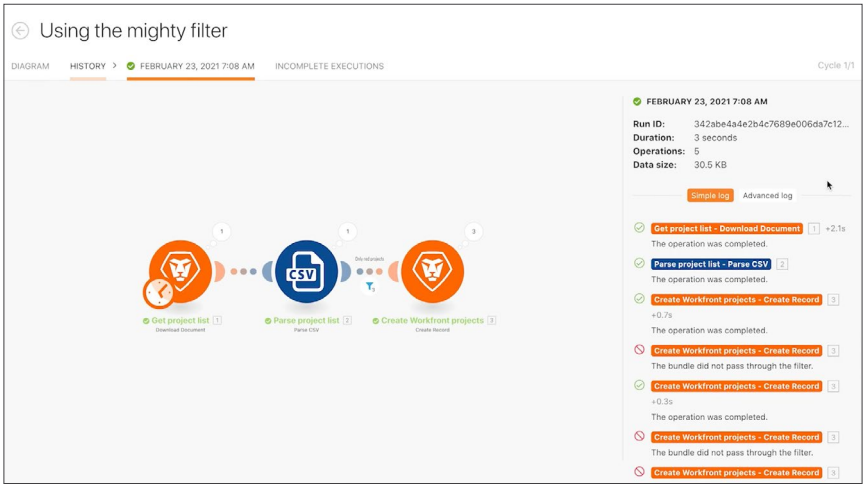
1. Open your "Using the might filter" scenario.

2. From the overview page, click the History tab (at the top, under the scenario name).

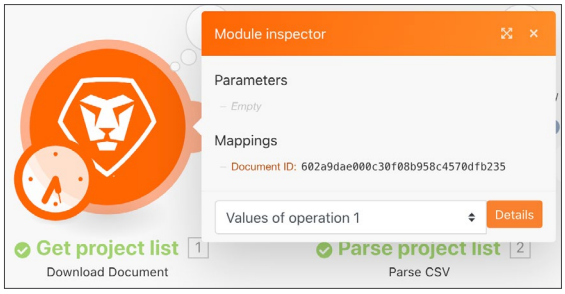


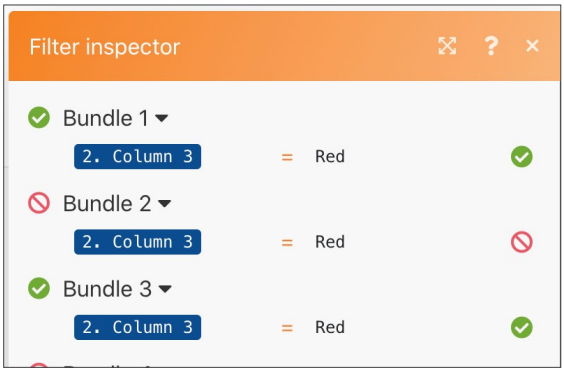
Execution history (cont.)

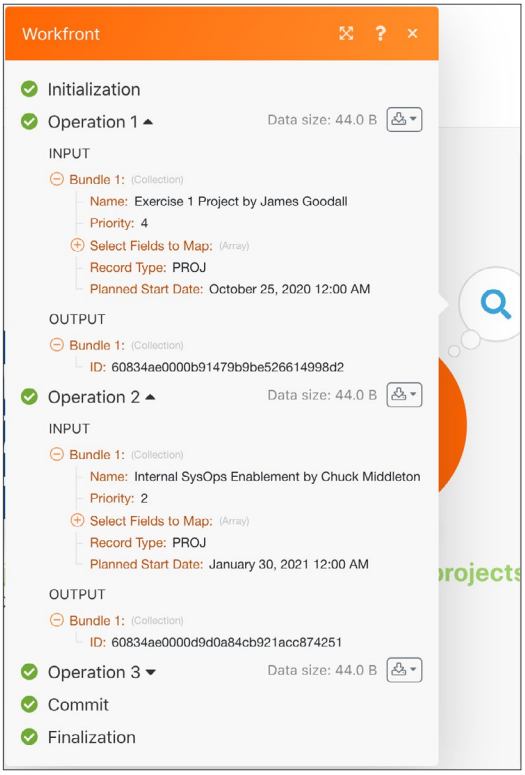
3. Find an execution and click the details button to open a page that shows the specific operations performed (or not performed) in the right panel. In the left panel, you can examine the scenario as it was at the time of execution.



4. When you click a module in the scenario panel, a module inspector panel appears, displaying information on the settings of the module. Click on the execution inspector next to a module or filter to see which bundles of information were executed on.





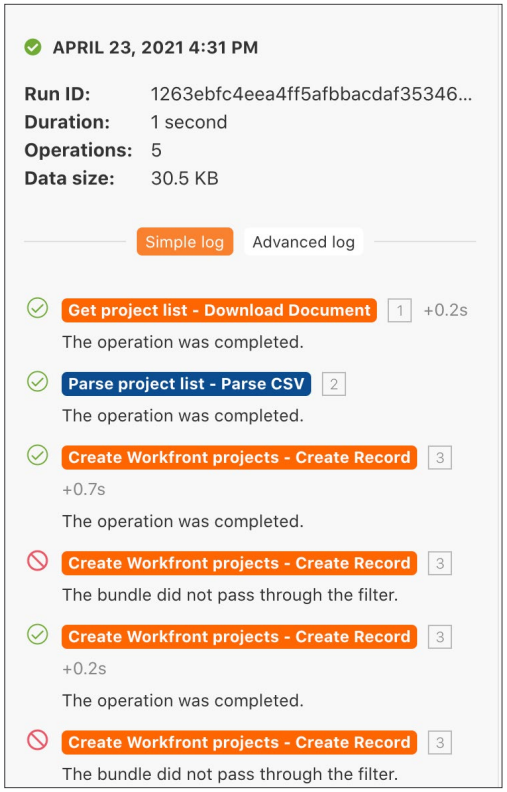


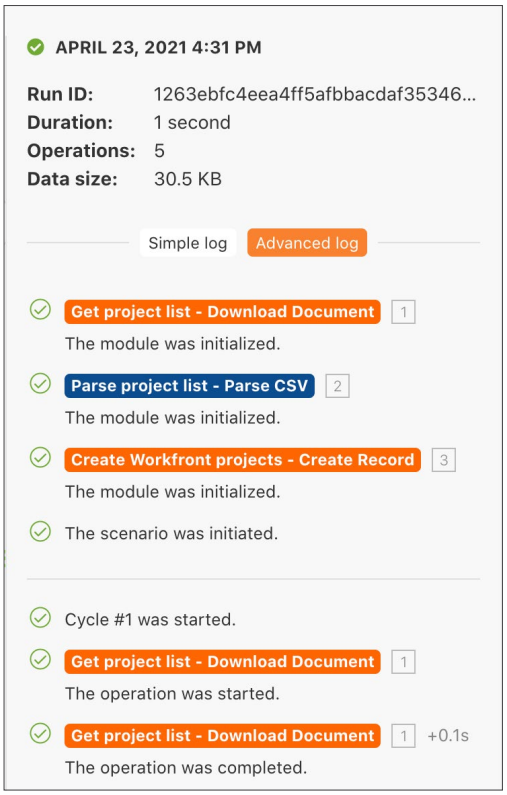
Execution history (cont.)

5. In the right panel, scroll through or click through the Simple log to view details of the execution's "play-by-play."

- You can see when operations were completed in modules and when bundles passed (or did not pass) through filters.
- Click a log item to open the operation panel in the scenario panel. The logs are listed in chronological order of when they occurred.

6. The Advanced log shows similar information. However, it provides more information on how many cycles were executed per run and lets you dig deeper into which bundles of information were processed in each cycle.



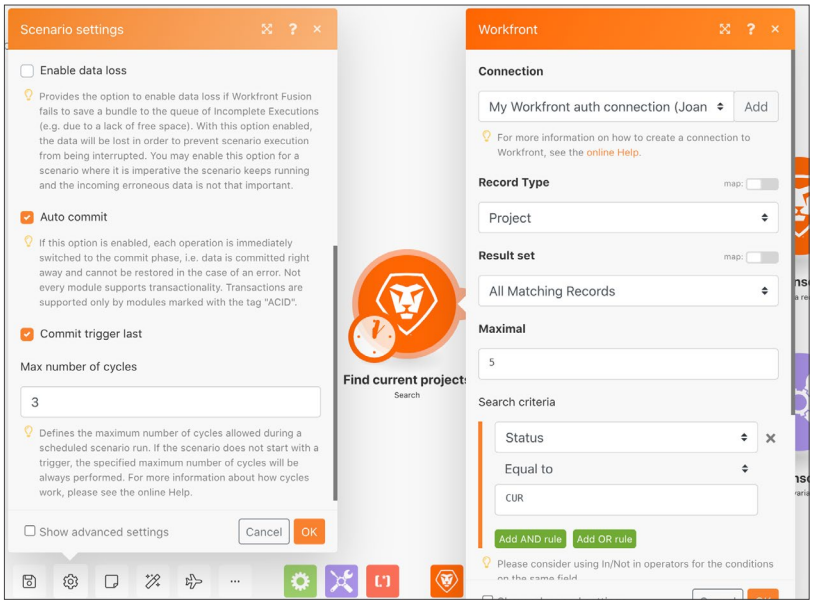


Exploring runs, cycles, and bundles

Understand how runs, cycles, and bundles behave using the execution history of a scenario.

Exercise overview

Practice with different scenario configurations to explore using runs and cycles.



Steps to follow

1. Clone the scenario named “Sharing variables between routing paths.” Name the new scenario “Sharing variables between routing paths - Cycles test.”

2. Remove the Send an email module, as it’s not needed for this test.

Set up your scenario to process 3 cycles per run. Process 5 projects in each cycle.

3. Click the trigger module and change the Maximal field to 5, so only 5 projects are processed in each cycle.

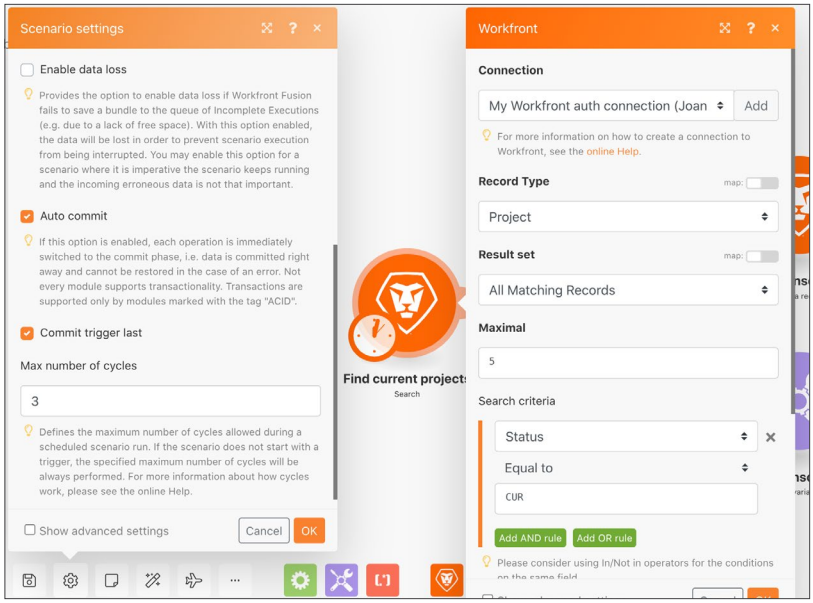
4. In the Search criteria, remove the second filter that restricts the search to a single project.

Exploring runs, cycles, and bundles (cont.)

5. Click OK.

6. In the Fusion toolbar, open Scenario settings and change the Max number of cycles field from 1 to 3.

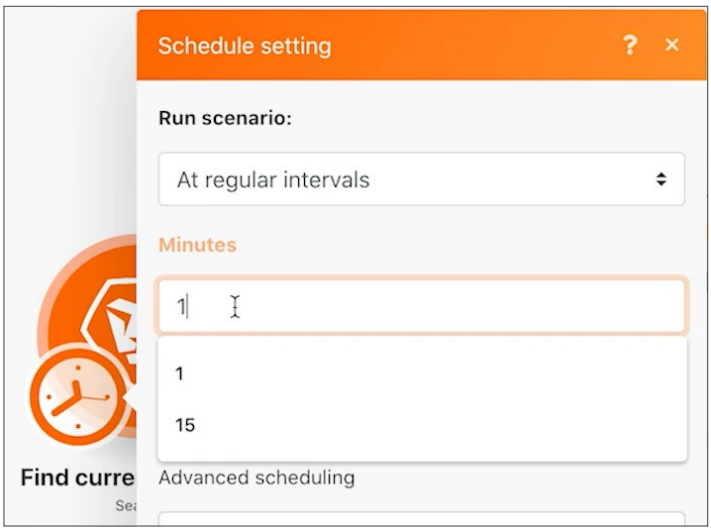
7. Click OK.



Schedule the scenario to run every minute.

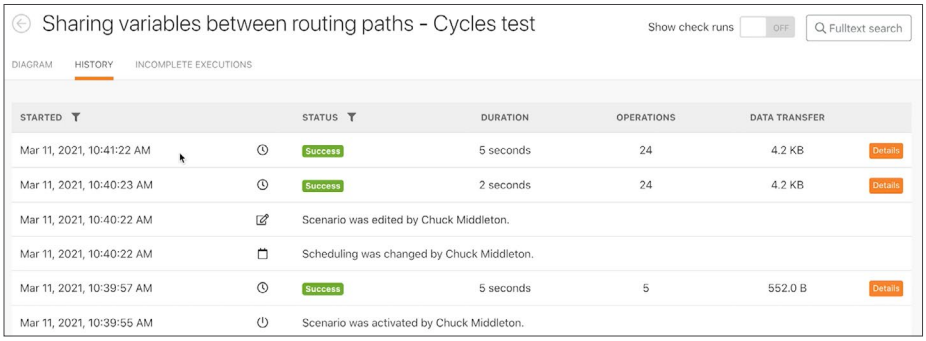
8. Click the clock icon by the trigger module and change the Minutes field to 1 minute.

9. Next, switch the Scheduling toggle under the Run once button to On. Save your scenario.



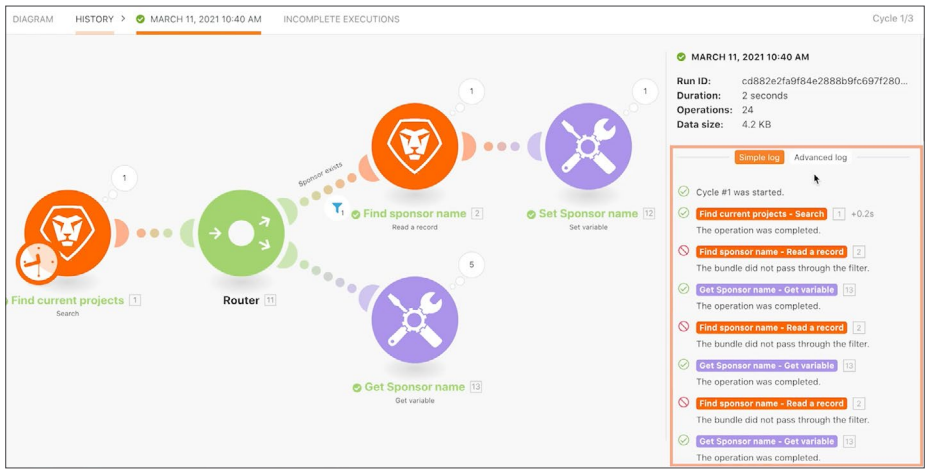
Exploring runs, cycles, and bundles (cont.)

10. Go to the Execution History for the scenario and watch as a new history record appears within the next minute. You may need to refresh the page.

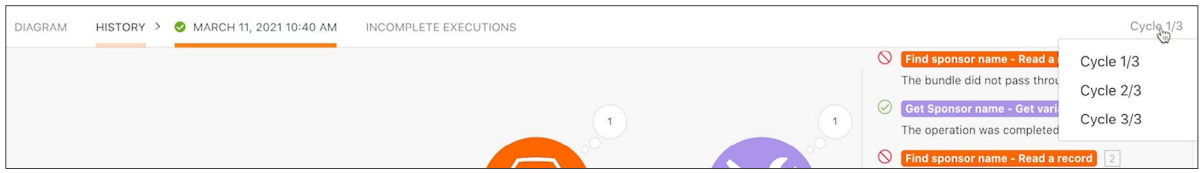


11. Click the Details button of a run. Click through the Simple log in the right panel, similar to what you did in the execution history portion of the Workfront Fusion training.

12. Records of processed operations are sectioned into cycles.



13. A drop-down menu at the top-right of the window allows you to select any of the 3 cycles you set up to run every time.

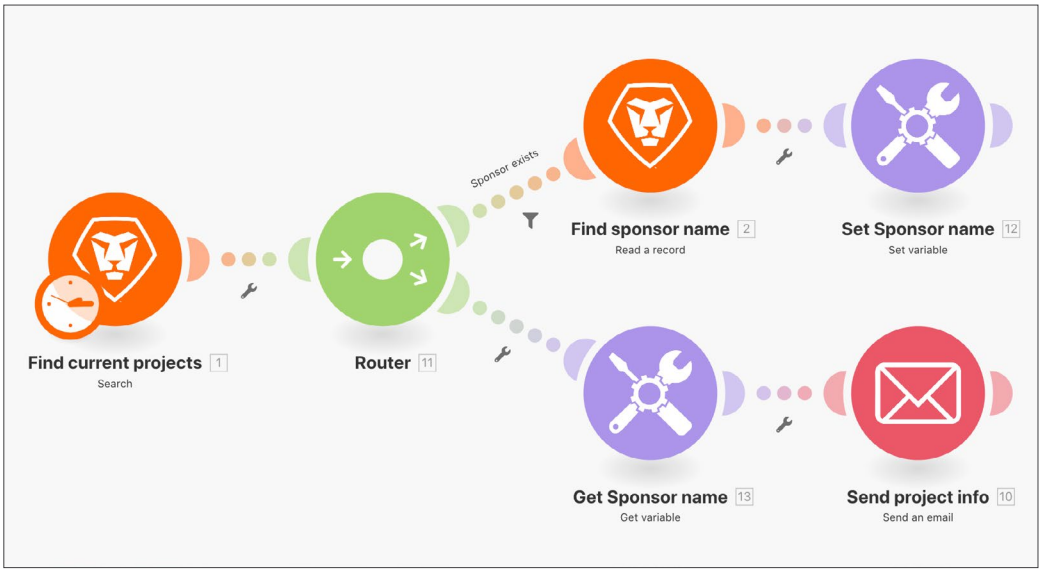


Switch function

Learn how to use the switch functionality using the Switch function.

Exercise overview

For simple data changes, use the Switch function to transform one value to another within a module field. In this exercise, change the two-letter key to the actual name for the project progress status to send in an email.



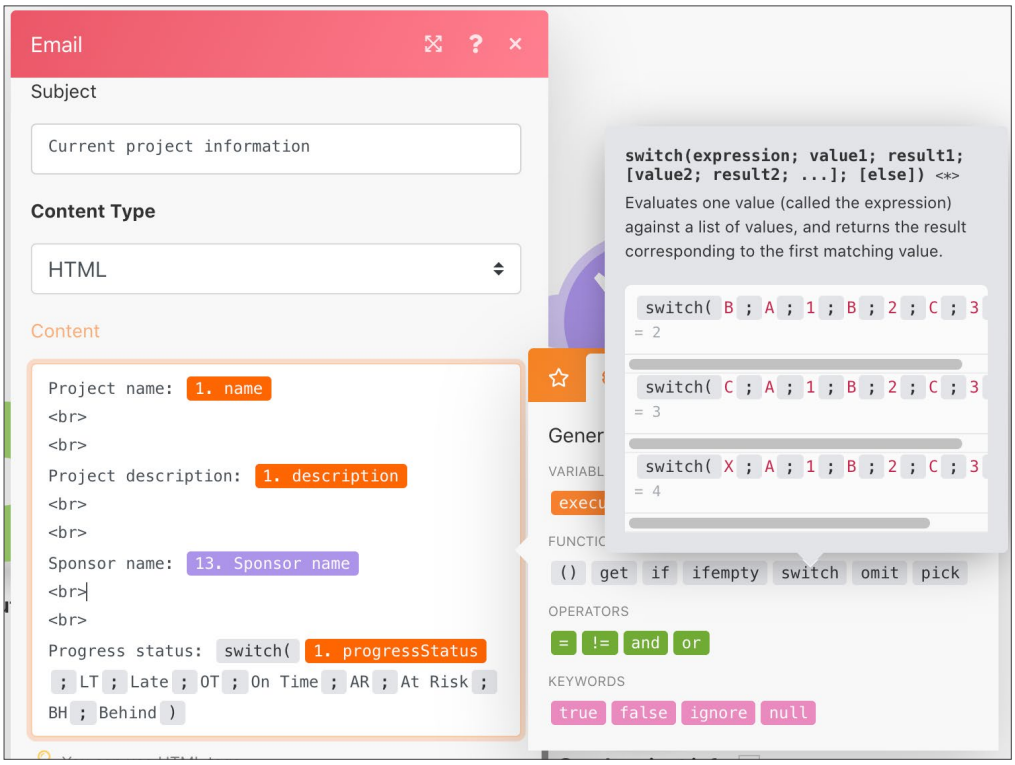
Steps to follow

1. Clone the scenario named "Sharing variables between routing paths."
2. Name the new scenario "Sharing variables between routing paths - Switch."
3. Click the trigger module and add Progress Status to the Outputs section.
4. In the Send an email module, add Progress Status to the Content field.
   - If you just map over the value coming from the Search module, there's a two-letter code for the progress status.
   - To "switch" the code for the full name of each possible progress status, use the "switch" function from the General functions tab.

Switch function (cont.)

5. The switch function uses the Progress Status value or expression as a key, then returns the output value based on that key.

   - A key value is defined in the first position after the Progress Status (“LT”) with the corresponding output defined in the second position (“Late”).
   - The next key value is defined in the third position, with the corresponding output defined in the fourth position, etc., for as many keys as desired.

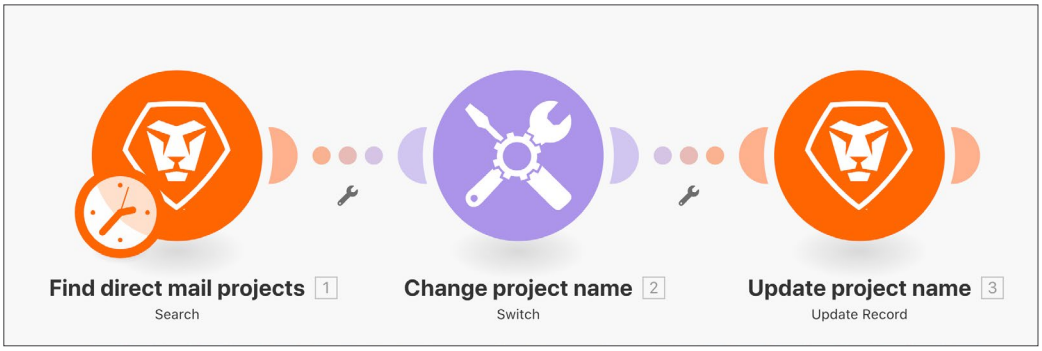


Switch module

Understand how to use the Switch module when you need to perform more complex or dynamic data transformations.

Exercise overview

Search for direct mail projects in your test drive, then change the name of each project based on a value selected in a custom field attached to the project.



Steps to follow

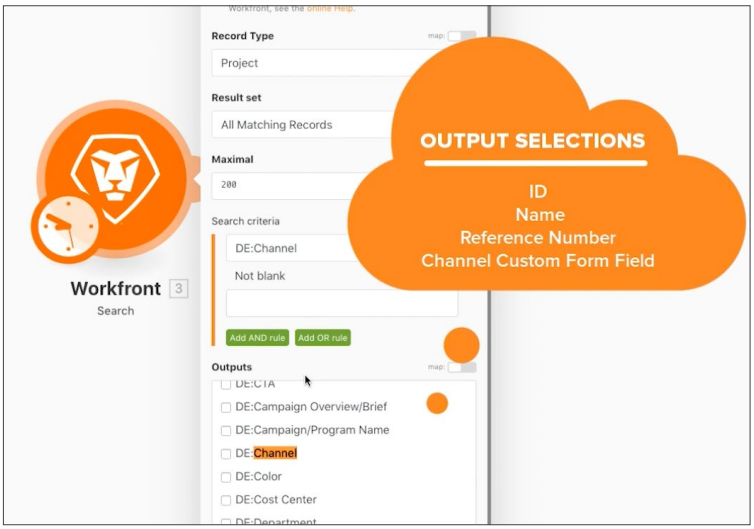
1. Create a new scenario and name it "Using the Switch module."

2. For the trigger module, use the Workfront Search module.

3. Set up your Workfront connection, and set the record type to Project.

4. In the Search criteria, specify that you only want to see projects that have a value in the Channel custom field.

5. For outputs, select ID, Name, Reference Number, and the Channel custom field.

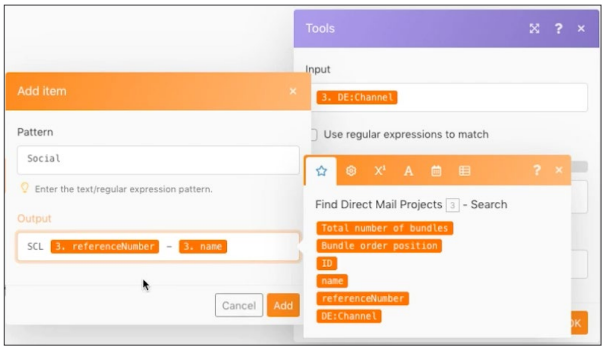


Switch module (cont.)

6. Add the Switch module from Tools.

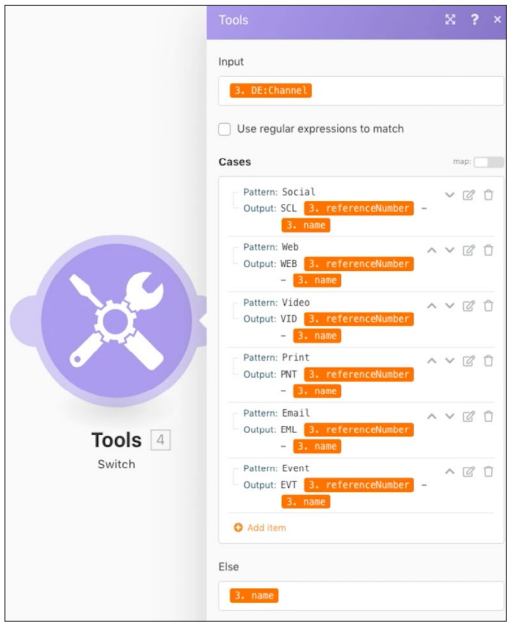
7. For the Input field, map the Channel custom field from the Search module.

8. Next add cases for each possible value coming from the Channel custom field. The possible value goes in the Pattern field. You want the output field to include a specific 3 letter code followed by the project reference number, then the project name.



Your first item should look like the image at top right:

9. You can add as many additional cases as you want. Notice the Else field at the bottom (middle image at right). This will be used if the input value doesn't match any of the cases.



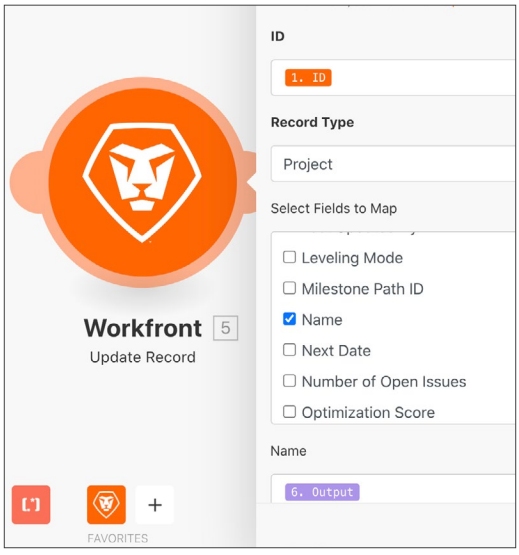
Update the project name in Workfront.

10. Add a Workfront Update Record module.

11. In the ID field, map to the ID from the trigger module.

12. Set the Record Type to Project.

13. Select the Name field from the Select Fields to Map section, and map it to the output from the Switch module. (bottom image)



14. Save your scenario and Run once. View the updated project names in your test drive.

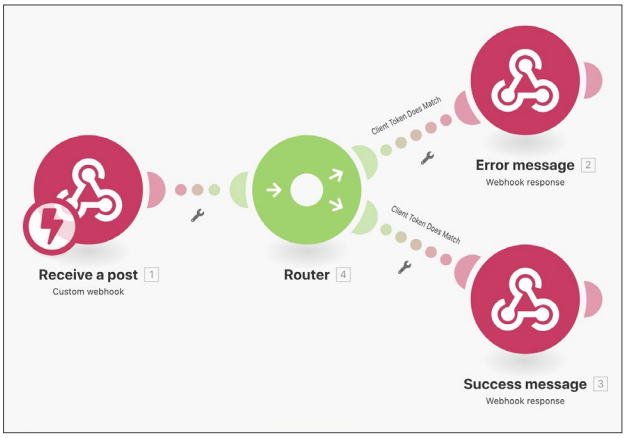
Webhooks

Learn how to create, trigger, and manage webhook-initiated scenarios.

Exercise overview

The purpose of this scenario is to create an app to sell to convenience stores so they can easily determine whether or not a customer is old enough to purchase alcohol. The cashier simply needs to post the name and birthdate of the customer to a URL they have been provided. That post will trigger the scenario which will calculate the answer and return it to the requestor.

1. The scenario consists of three webhooks.
2. The trigger module is a custom webhook that listens for a post.
3. When it receives a post it will output it to one of the next modules.
4. The next module returns a response to the requestor.



Steps to follow

Set up the trigger webhook.

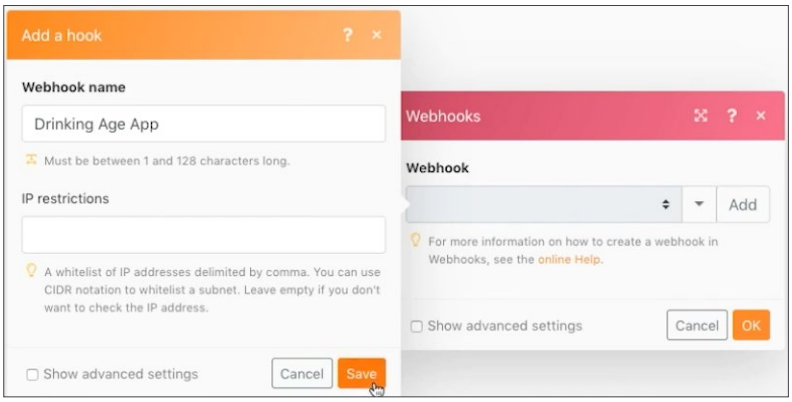
1. Create a new scenario and name it "Using webhooks."
2. For the trigger, add the Custom webhook module from the Webhooks app.
3. Click on Add to create a new Webhook.

Webhooks (cont.)

4. Enter the Webhook name of "Drinking age app."

5. Leave IP restrictions blank, which means that anyone can send data into it.

6. Click Save.



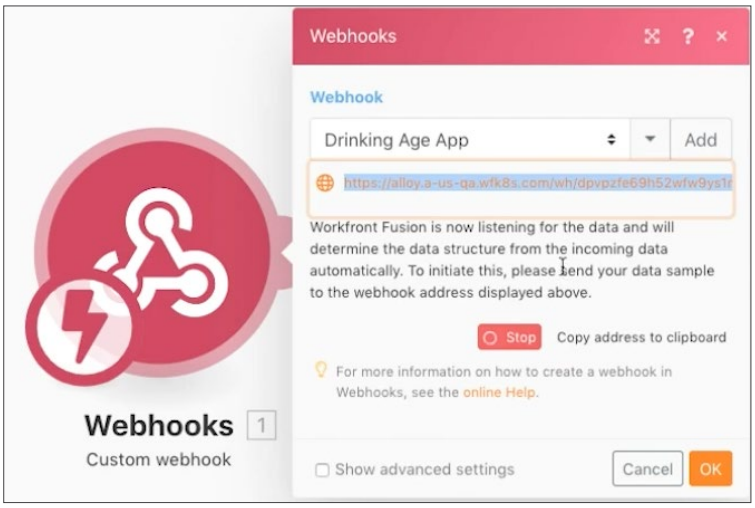
7. Back in the Webhooks mapping panel a URL has been created for this specific webhook. Click on "Copy address to clipboard" to copy that URL.

8. Click OK.

9. Click Run once.

10. Use the URL in Postman to send a name and birthdate to your custom webhook. For instructions on setting up Postman, see section three of the **Beyond basic modules in Workfront Fusion** learning path.

The Webhooks module panel should look like this:

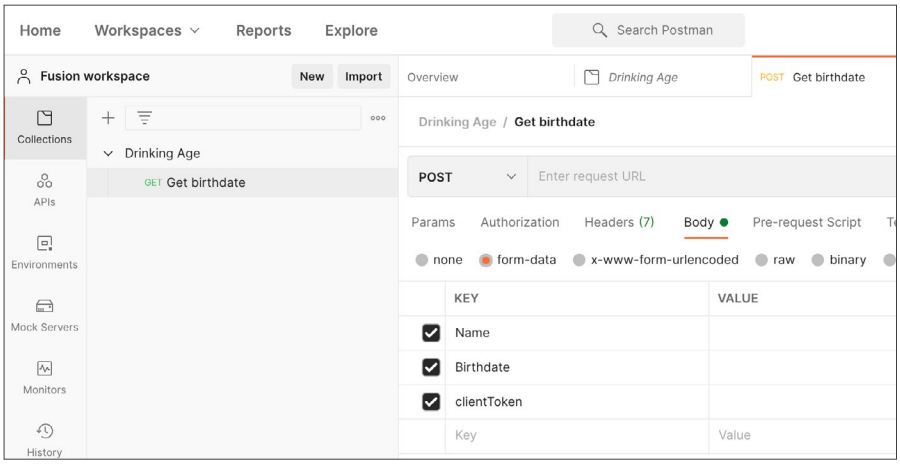


Webhooks (cont.)

The webhook is now in a state where it's listening for data to determine the data structure.

11. You can define the data structure of the payload that you expect to get (data structures will be discussed later). If you don't define a data structure Fusion will determine the data structure automatically when the post is sent.

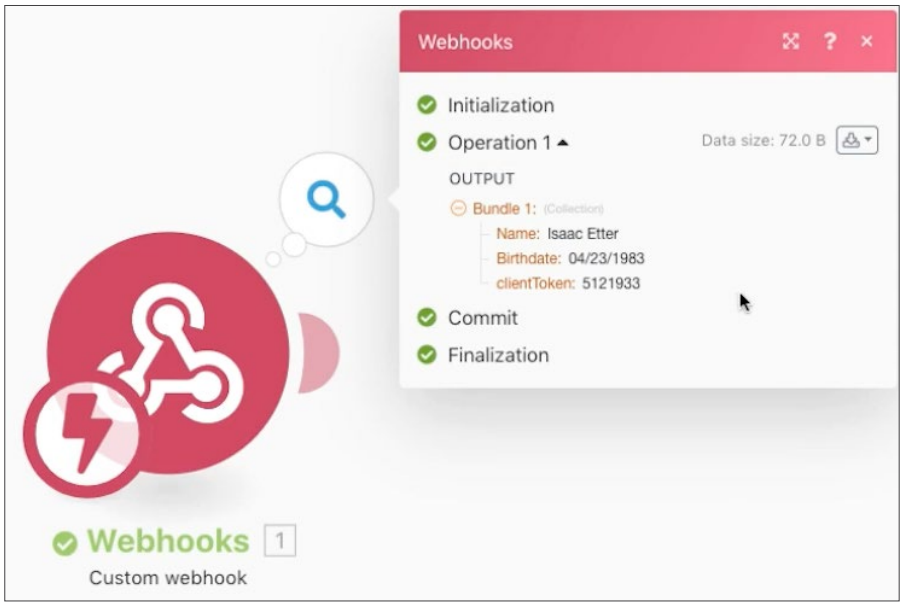
12. On the Postman side you want to send to the copied URL. The post should include basic form data. For this example you need three fields: Name, Birthdate, and clientToken.



13. After you click Send from Postman you should get an indication that the post has been accepted.

14. This is the point where your scenario will show that the data structure has been successfully determined.

15. You can see that the data has been received by opening the execution inspector.



Webhooks (cont.)

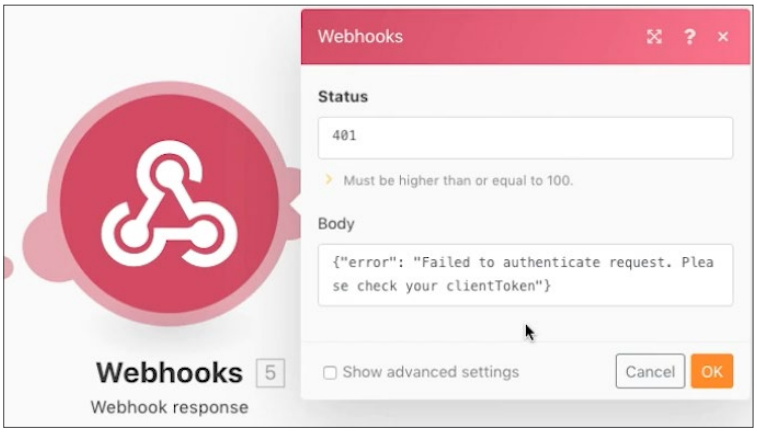
Set up routing for client tokens.

16. Add a router to the trigger module.

17. In the upper path, add a Webhook response module. This will be our path for when the client token does not match.

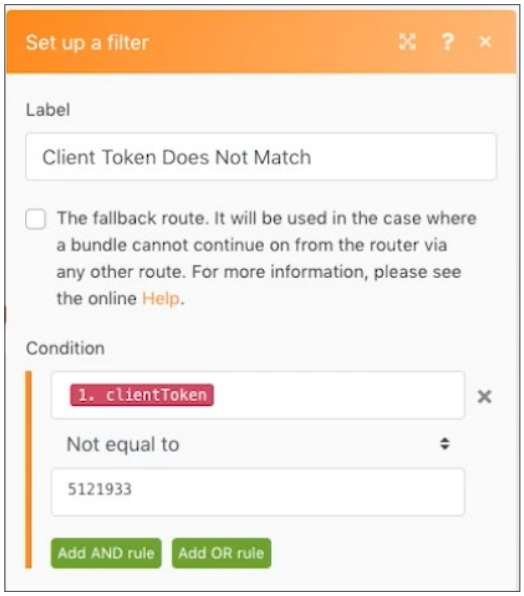
18. Set the status to 401.

19. Set the Body to {"error": "Failed to authenticate request. Please check your clientToken"}.



20. Create a filter between the router and the Webhook response module. Name it "Client token does not match."

21. For the Condition, use the clientToken field from the trigger module and do a numeric "Not equal to" comparison to the number 5121933.

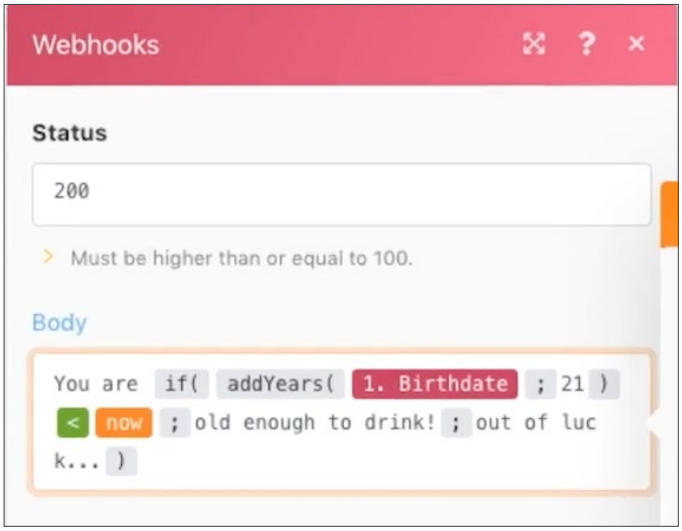


Webhooks (cont.)

22. In the bottom path, add another Webhook response module. This will be our path for when the client token does match.

23. Set the status to 200.

24. In setting up the Body, use the mapping panel functions to test to see if the person is 21 or older. If they are, return "You are old enough to drink!", otherwise return "You are out of luck..."



25. Create a filter between the router and the Webhook response module on the lower path. Name it "Client token does match."

26. For the Condition, use the clientToken field from the trigger module and do a numeric "Equal to" comparison to the number 5121933.

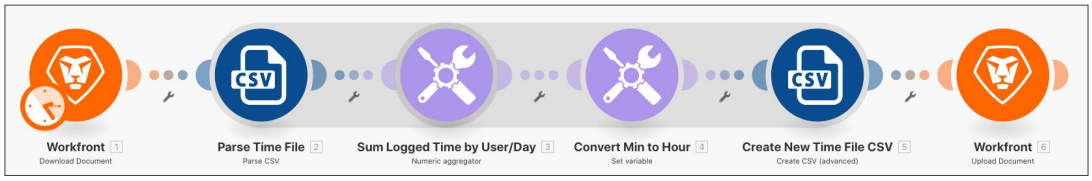
27. Click on the Scheduling button under Run once to activate your scenario so that any time there is a new post it will be received, go down either path, and generate a response.

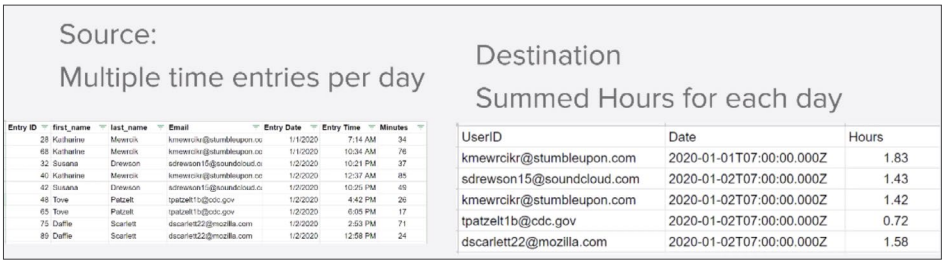
Data structures

Transform data from a source file into a destination file.

Exercise overview

Open a CSV file that contains a list of time entries. These time entries are for minutes logged throughout certain days by multiple users. The goal is to take this information and produce a new CSV that shows the total time, in hours, logged by each user, each day.





In this scenario, you will open a file that contains a list of time entries for minutes worked, including the date and time, how many minutes were entered, and the email address of who made the entry. There are 100 time entries, some made by the same individuals and some of those were made on the same day as others.

To produce a file that shows the total time, in hours, worked each day by each individual, you'll follow these steps:

1. In the trigger module, get a file from the Workfront folder. Download the file.

2. In the first CSV module, parse the time entry data to output one bundle for each time entry. This is an iterator.

3. The first Tools module is a numeric aggregator. This will SUM all the minutes and group the rows by email address, then by date. The result is the total minutes worked each day by email address.

4. The second Tools module is a Set Variable module. Use this to format the minutes to divide by 60 and round to 2 decimals.

5. In the second CSV module, set up the output file.

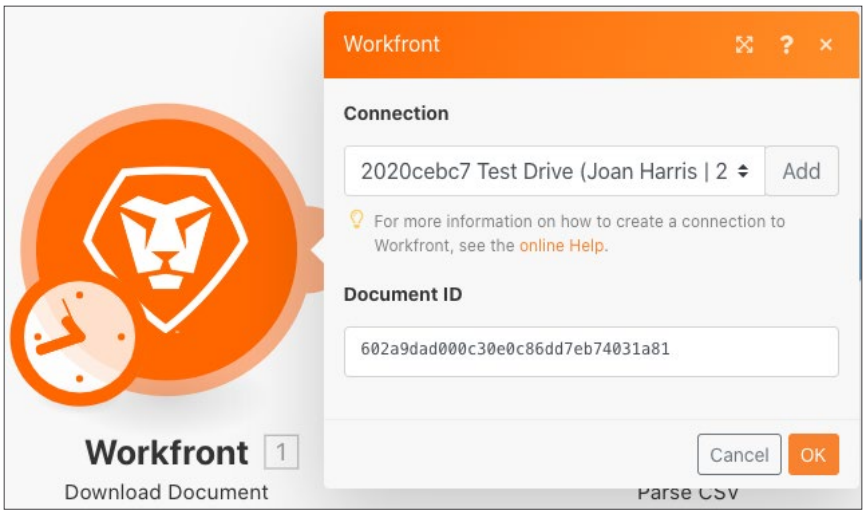
6. In the final module, upload the CSV file into Workfront.

Data structures (cont.)

Steps to follow

Download the file from Workfront.

1. In the Workfront "Fusion Exercise Files" folder, select "_Fusion2.0JanTime.csv" and click Document Details.
2. Copy the first ID number from the URL address.
3. Create a new scenario. Name it "Creating & using data structures."
4. Start with the Download Document module from the Workfront app.
5. Set up your Workfront connection and include the Document ID you copied from the Workfront URL.

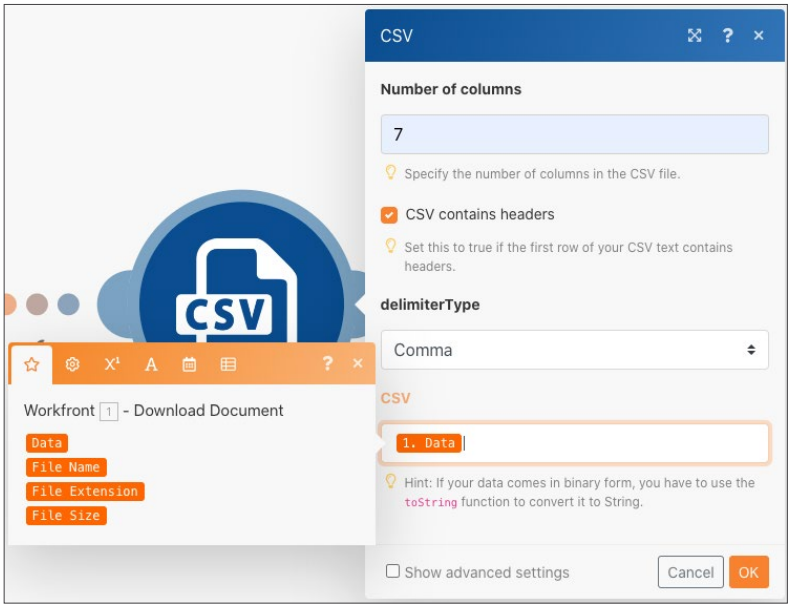


Parse the time entry data.

6. Add another module, selecting Parse CSV.

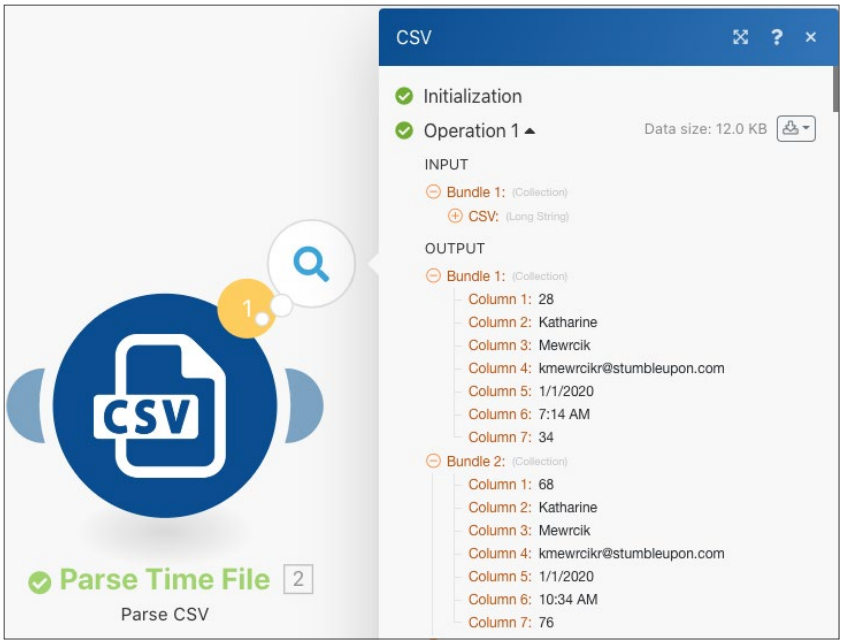
Data structures (cont.)

7. Set up Parse CSV for 7 columns. Check the CSV contains headers box. Choose the Comma delimiterType and put Data in the CSV field.



8. Click Run once to view the output.

9. Open the execution inspector to see the inputs and the outputs of the Parse CSV module. There is one bundle (a CSV file) as input and several bundles as outputs (one bundle for each row in the CSV file). It should look something like this:

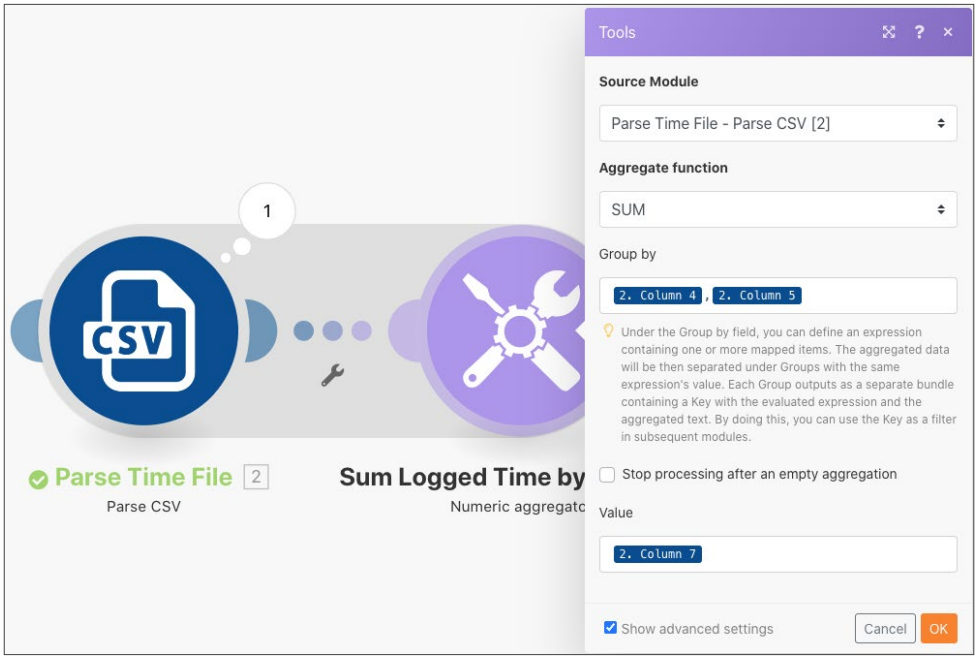


Data structures (cont.)

Next, transform the data into the desired output form, with aggregated time totals expressed in hours instead of minutes.

10. Add a Numeric Aggregator tool module.
11. Select the source module, which is the Parse CSV module.
12. Select SUM for the aggregate function.
13. The Value field is column 7 from CSV file. This is the minutes logged by each user.
14. To sum the fields by group, click Advanced Settings and set Group by to email (column 4), date (column 5).
    - This will sum for every combination of the email and the date. Be sure to put a comma between column 4 and column 5. This will be used as a delimiter later on.

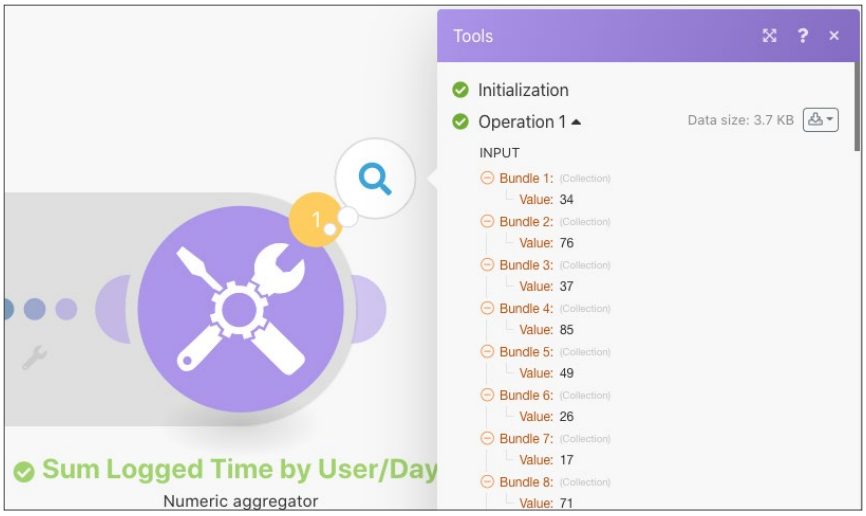
Your mapping panel should look like this:



15. Click Run once to check the aggregation output.

Data structures (cont.)

The output bundles should look like this:



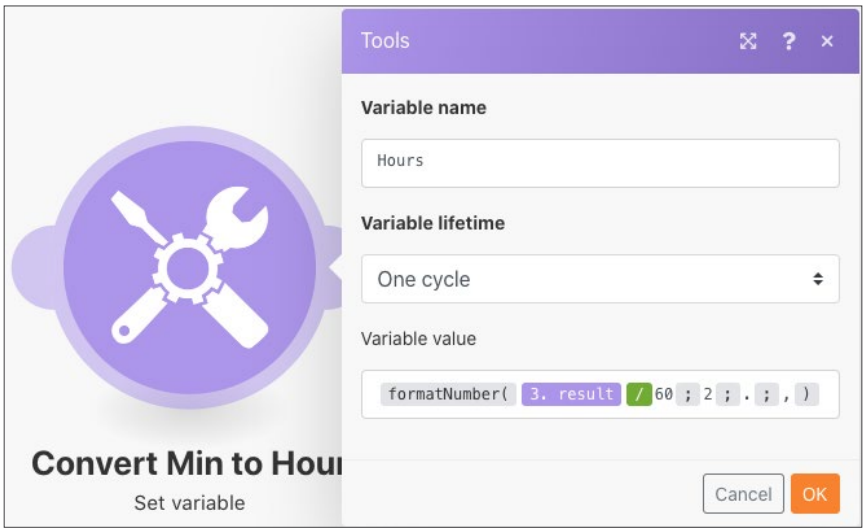
Now convert the aggregated minutes into hours.

16. Add another tools module, selecting Set Variable.

17. Name the variable "Hours."

18. Set the variable value to formatNumber(result/60;2;.;,)

Your mapping panel should look like this:



Data structures (cont.)

Next, get the values set up for the output file. You want the userID and the date value used for the groupings. You also want the hours that were calculated.

19. Add another module— CSV module using the aggregator Create CSV (advanced).

20. The source module is the Tools - Numeric aggregator.

21. Click Add by the Data structure field and name our data structure "Time Logged Daily Sum."

22. Click Add Item to create the first item.

23. Name the item "UserID" and set the type to Text. Click Add.

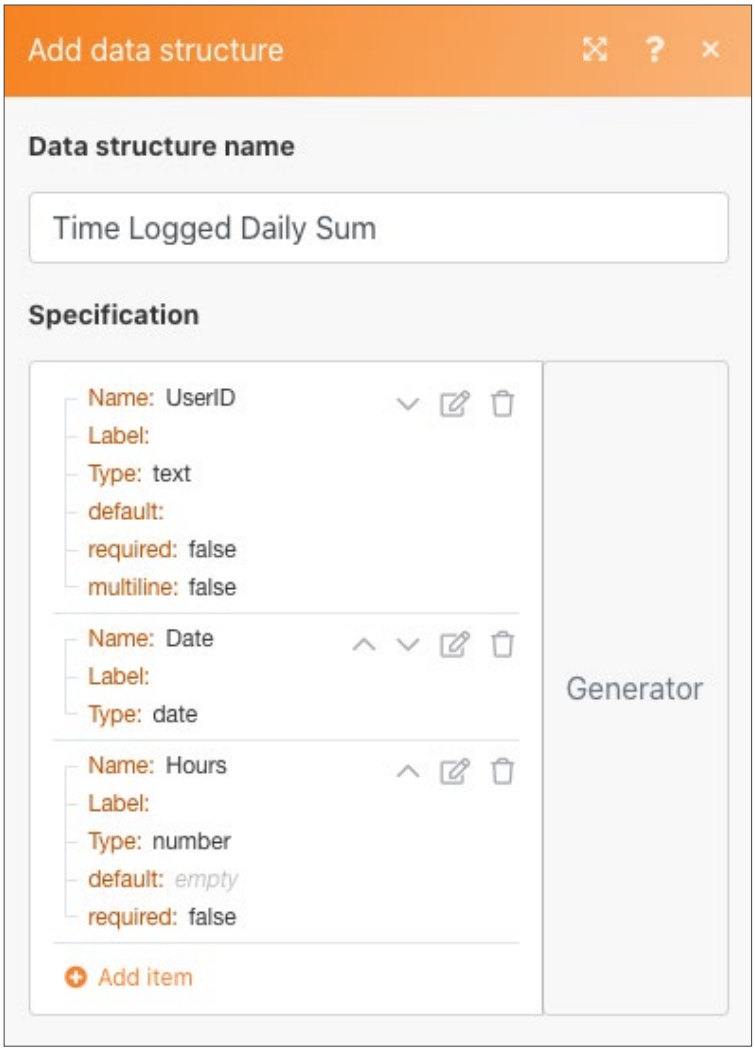
24. Click Add Item again to create the second item.

25. Name the item "Date," set the type to Date, and click Add.

26. Click on Add Item one more time.

27. Name the item "Hours," set the type to Number, and click Add.

Your data structure should look like the image at right:



28. Click Save to finish the Time Logged Daily Sum data structure.

Now you supply the values for the three fields you just created. You should see those three fields in the CSV mapping panel.

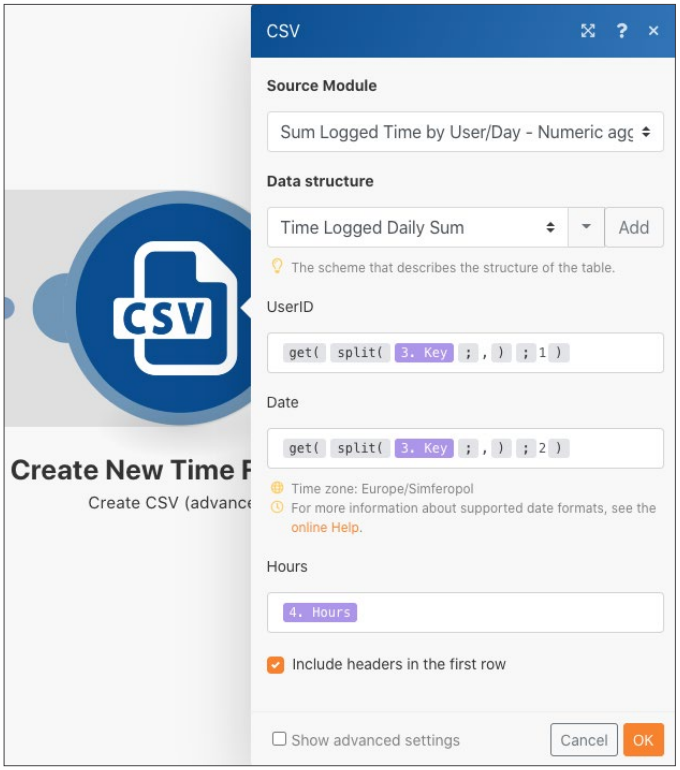
29. Click in the UserID field and choose GET from the general functions tab. In the first parameter, put SPLIT from the text and binary functions tab. The first parameter for the SPLIT function is the Key field. Add a comma as the delimiter and 1 as the index. This indicates you want the GET to retrieve the first field in the Key array.

Data structures (cont.)

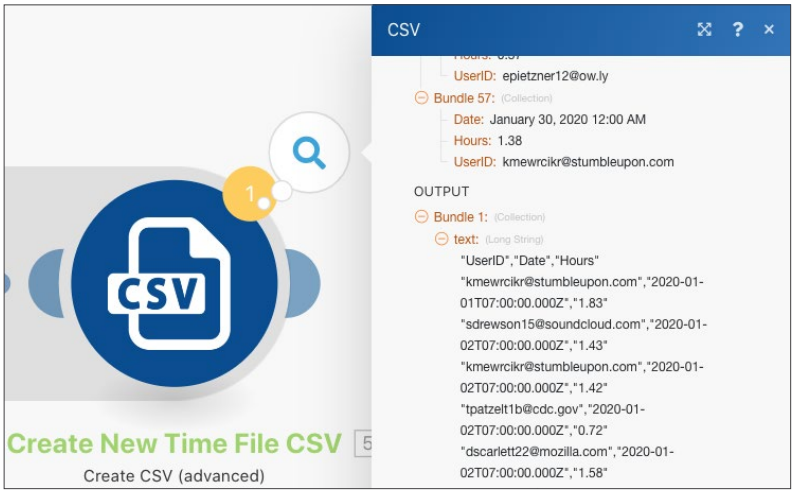
30. Copy this expression into the Date field. Change the index from 1 to 2 to GET the second value in the array.

31. For the Hours field, add the Hours field from the Set Variable tool.

Your CSV mapping panel should look like this:



If you run the scenario now, you should see this output:



Data structures (cont.)

Now, add a module to take this output and upload it as a document to an existing project in Workfront.

32. Open the project in Workfront and copy the project ID from the URL.

33. Go back to the scenario in Fusion and add another module—the Upload Document module from the Workfront app.

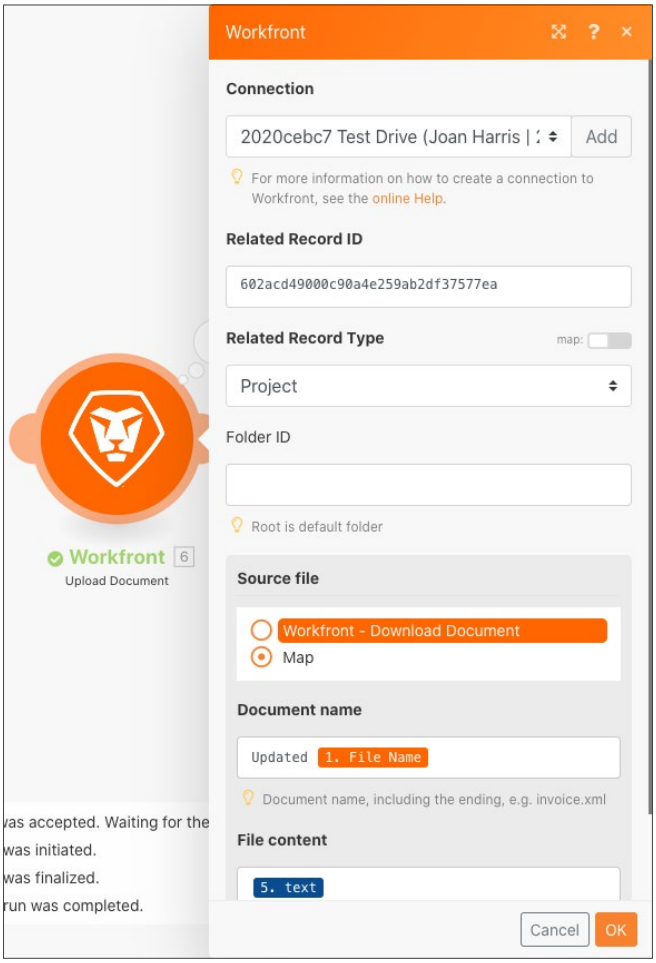
34. Paste the project ID into the Related Record ID field.

35. Choose Project for the Related Record Type.

36. Choose the Map option for the Source file.

37. For the Document Name, use the file name you downloaded, adding "Updated " in front of it.

38. For the File content, use the Text output from the Create CSV module.



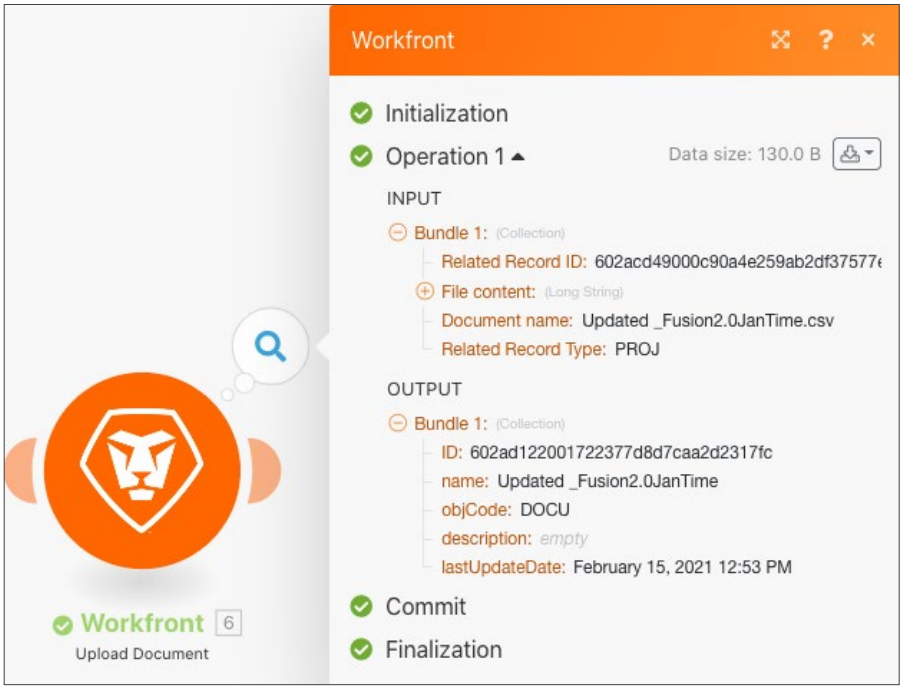
Data structures (cont.)

Your mapping panel should look like this:

39. Click OK and Save the scenario.

40. Click Run once to run the scenario.

Check the execution inspector in the Upload Document module to confirm the document was uploaded.

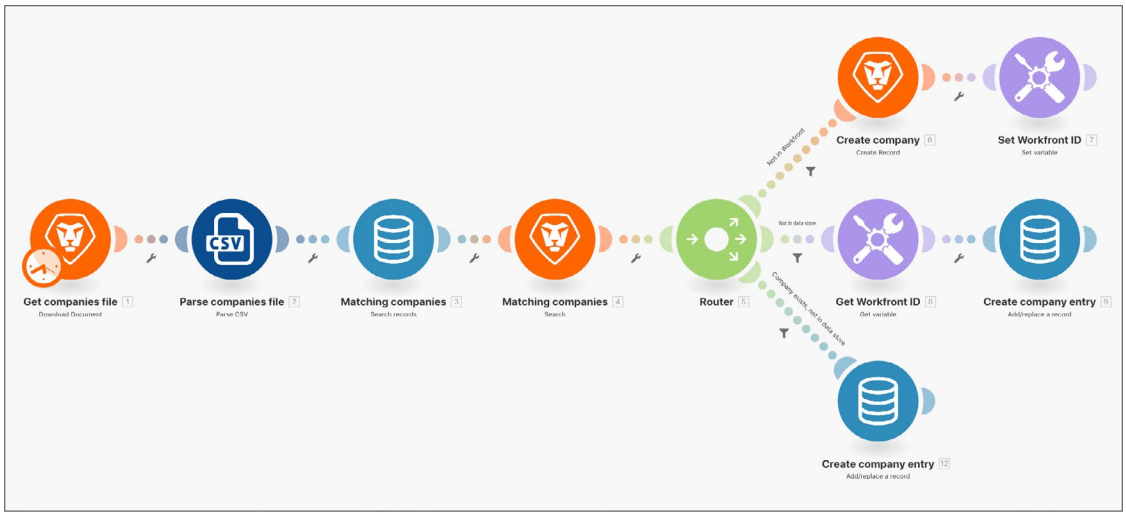


Data stores

Learn how to synchronize company names between two systems.

Exercise overview

This is the first part of a one-directional synchronization of companies in Workfront and another system. For now, it only syncs between a Fusion data store and Workfront. A table in a data store keeps track of the Workfront ID (WFID) and the company ID in the CSV file (CID) for each company. This allows for a bi-directional synchronization at some point in the future.



Steps to follow

Download the file from Workfront.

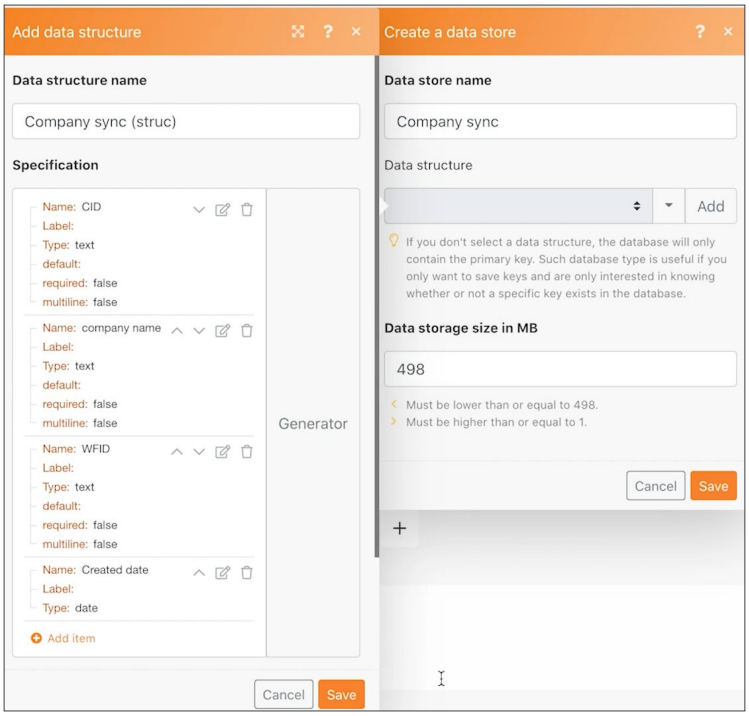
1. In the Workfront "Fusion Exercise Files" folder, select "_Companies.csv" and click Document Details.
2. Copy the first ID number from the URL address.
3. In Fusion, create a new scenario named "Using data stores to sync data."
4. For the trigger module, select the Workfront Download Document module.
5. Set up your Workfront connection and include the document ID copied from the Workfront URL.
6. Name this module "Get companies file."

Data stores (cont.)

7. Now add a Parse CSV module.
8. For the Number of columns field, type 2.
9. Map Data from the Download document module in the CSV field.
10. Name this module “Parse companies file.”
11. Save your scenario and click Run once.

Create a data store and a data structure.

12. Add a data store Search records module.
13. Create a new data store named “Company sync.”
14. Within the data store, create a data structure named “Company sync (struc).”
15. Create four fields.
    - CID -— The company ID in the CSV file
    - Company name
    - WFID — The Workfront company ID
    - Created date —Make sure that the data type is date

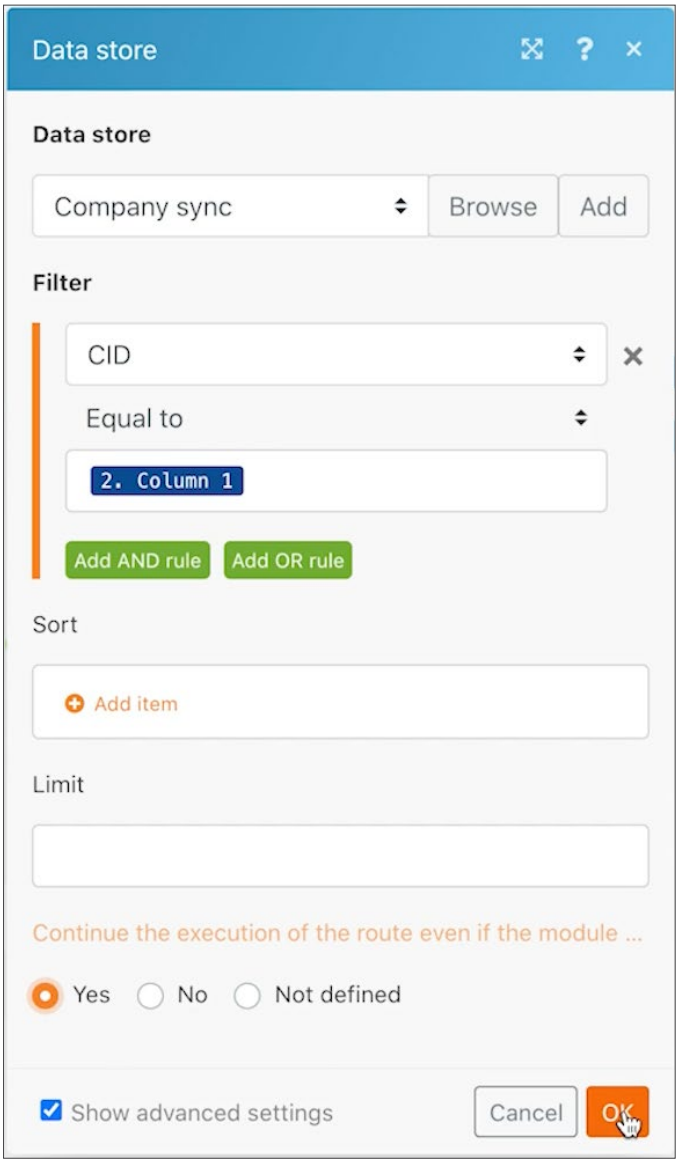


Data stores (cont.)

16. Click Save on the data structure, then set the data storage size to 1 and save the data store.

17. Continuing in the Data store module, set up a Filter where the CID is equal to the ID of the company from the Parse CSV module (Column 1).

18. Click Show advanced settings and select the option to "continue the execution of the scenario or the route, even if this module returns with no results."



19. Rename this module "Matching companies."

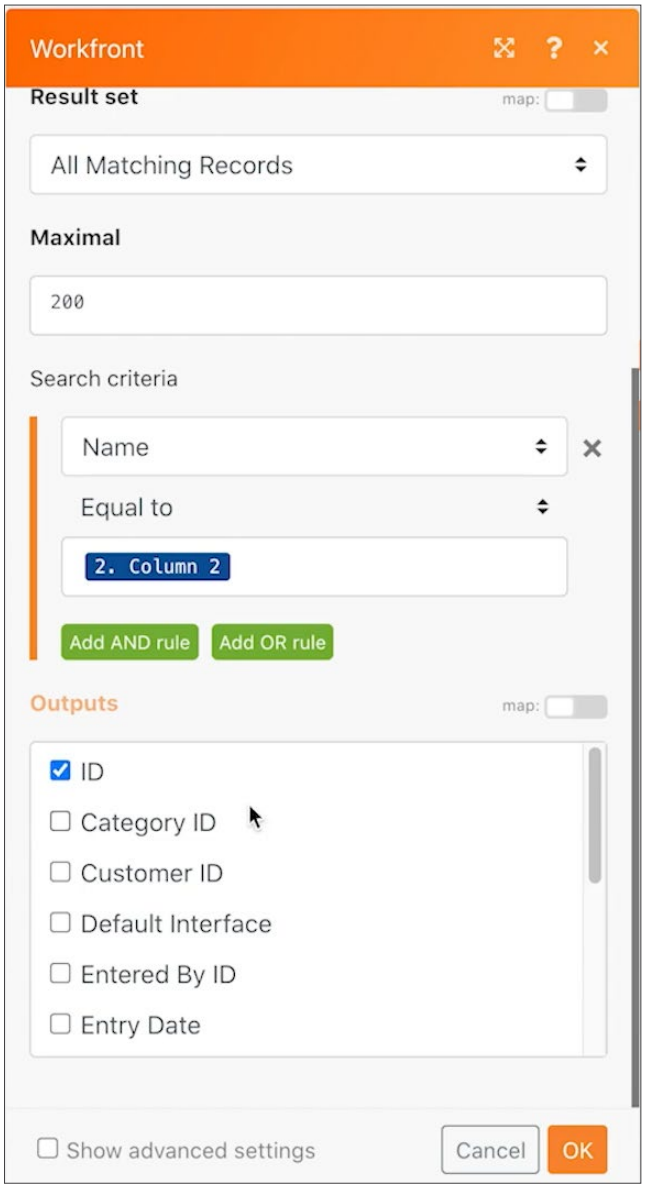
20. Add a Workfront Search records module.

Data stores (cont.)

21. Choose Company as the record type.

22. Search criteria is the company name within Workfront is equal to the company name in the CSV file.

23. For outputs, select the company name and the ID.



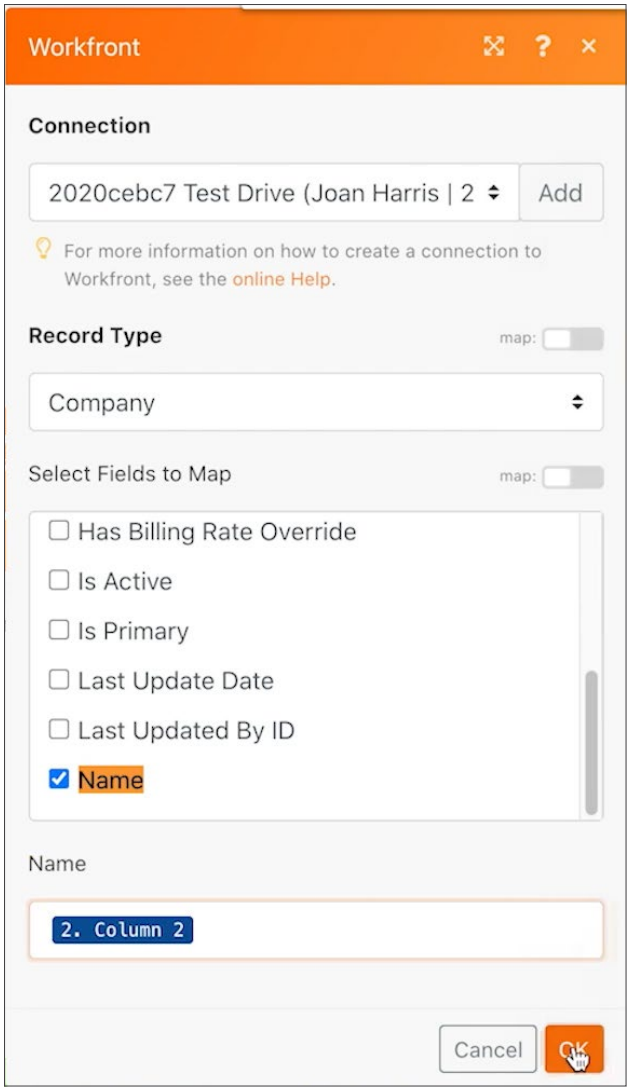
24. Click OK and rename this module "Matching companies."

Data stores (cont.)

Create different paths based on whether the company exists within Workfront or the data store.

Routing path 1—Create a company.

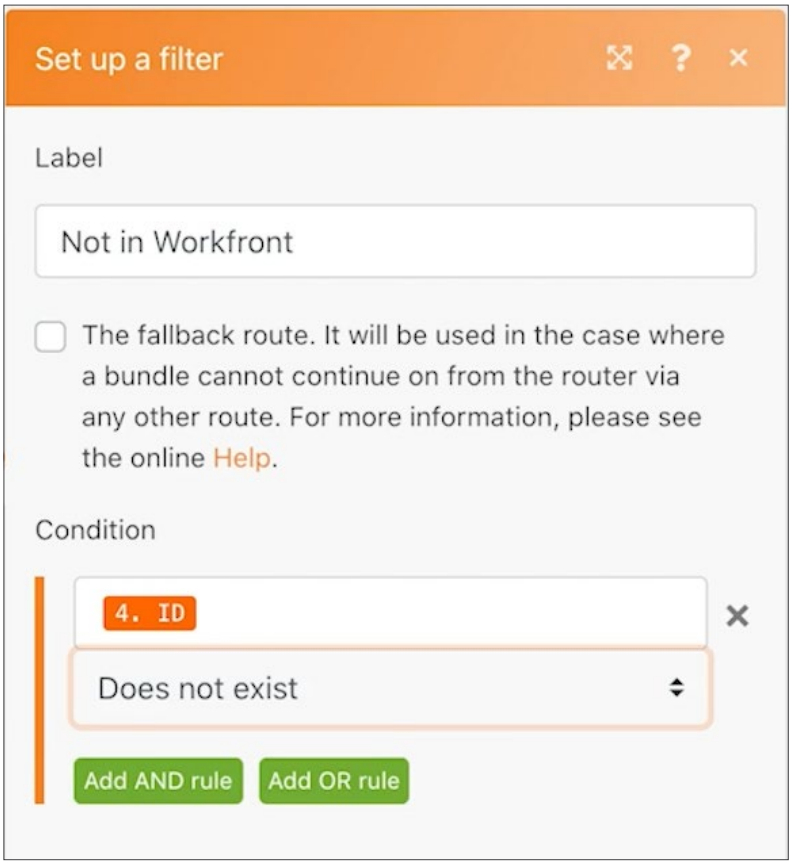
25. Add a router module to the right of the Workfront Search records module.
26. Add a Workfront Create Record module to the top path.
27. Set the record type to Company.
28. Select Name from Fields to Map. Map the name field to the output from the Parse CSV module (Column 2).
29. Rename this module "Create company."



Data stores (cont.)

30. Add a filter after the router to only create a company if it's not already in Workfront. Name it "Not in Workfront."

31. Set the Condition to the ID from the Workfront Search module and does not exist.



Prepare to update the data store in the next path.

32. Add a Set variable module to the end of the top path.

33. Set the Variable name to "Workfront ID."

34. Set the Variable value to the ID from the Create company module.

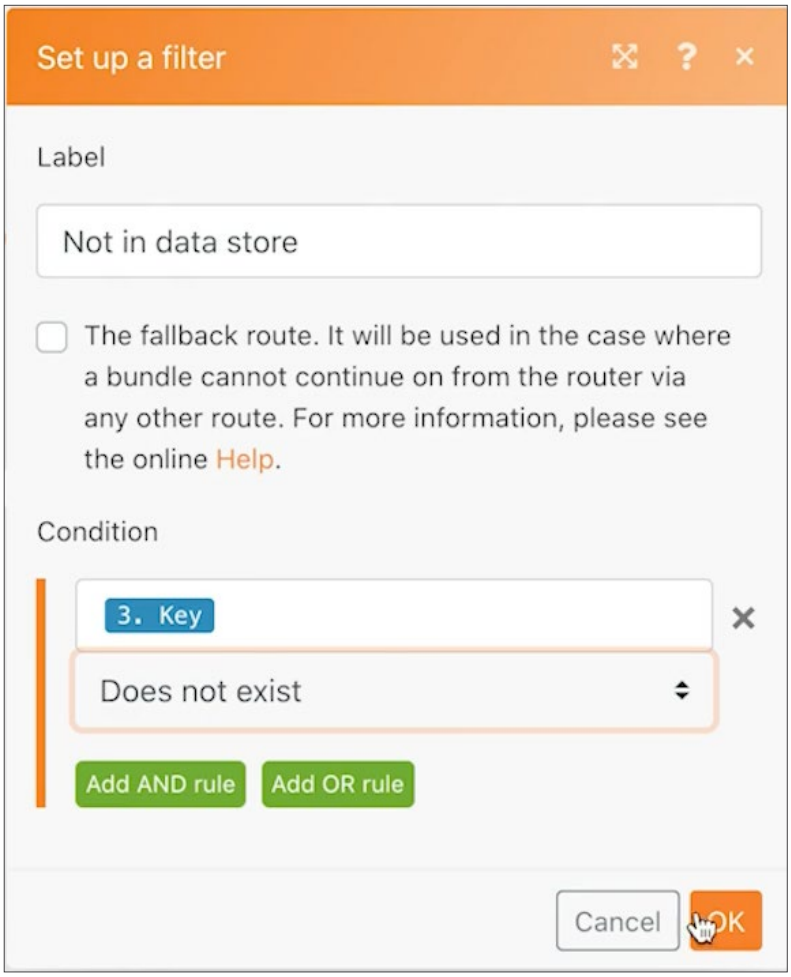
35. Rename this module "Set Workfront ID."

Data stores (cont.)

Routing path 2—Update the data store.

36. Create a filter on routing path 2. Name it "Not in data store."

37. Set the Condition to the Key from the Data store module and does not exist.



38. The first module in this path is the Get variable module.

39. Set the Variable name to "Workfront ID."

40. Rename this module "Get Workfront ID."

41. Add another module from the Data store app, Add/replace a record.

Data stores (cont.)

42. In the Data store field, choose Company sync. This is the data store you created earlier.

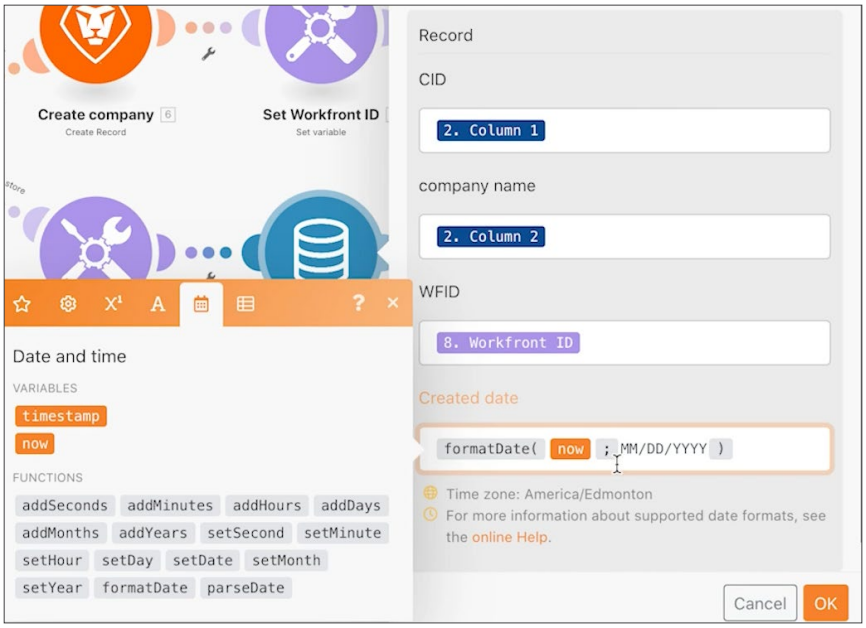
43. Leave the Key field blank.

44. Map the CID field from Column 1 in the Parse CSV module.

45. Map the company name field from Column 2 in the Parse CSV module.

46. Map the WFID field from the Get Workfront ID module.

47. For the Created date field, use the formatDate function from the Date and time tab to format the current date as MM/DD/YYYY.



48. Click OK and rename this module "Create company entry."

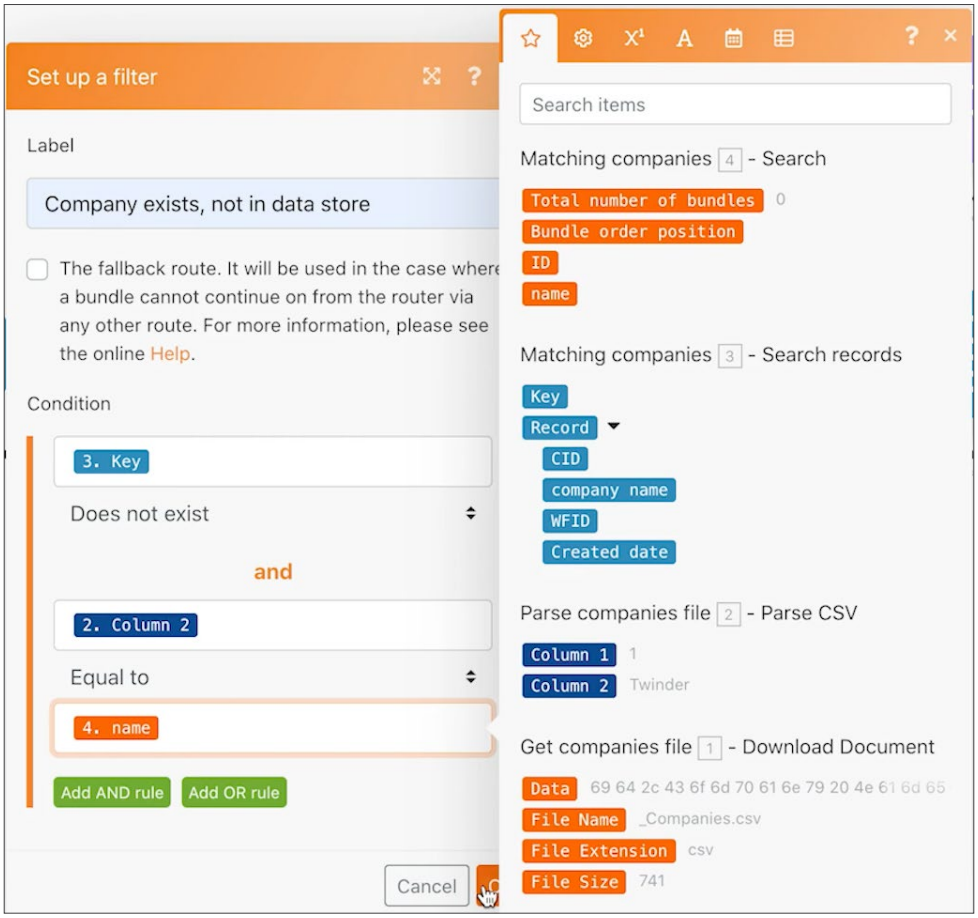
Data stores (cont.)

Routing path 3—Sync the data store between systems.

49. Start by creating a filter on routing path 3. Name it "Company exists, not in data store."

50. Set the Condition to the Key from the Data store Search records module and does not exist.

51. Click the Add AND rule button and designate that the company name from the CSV file (Column 2) is equal to the name of the company found in the Workfront Search module.

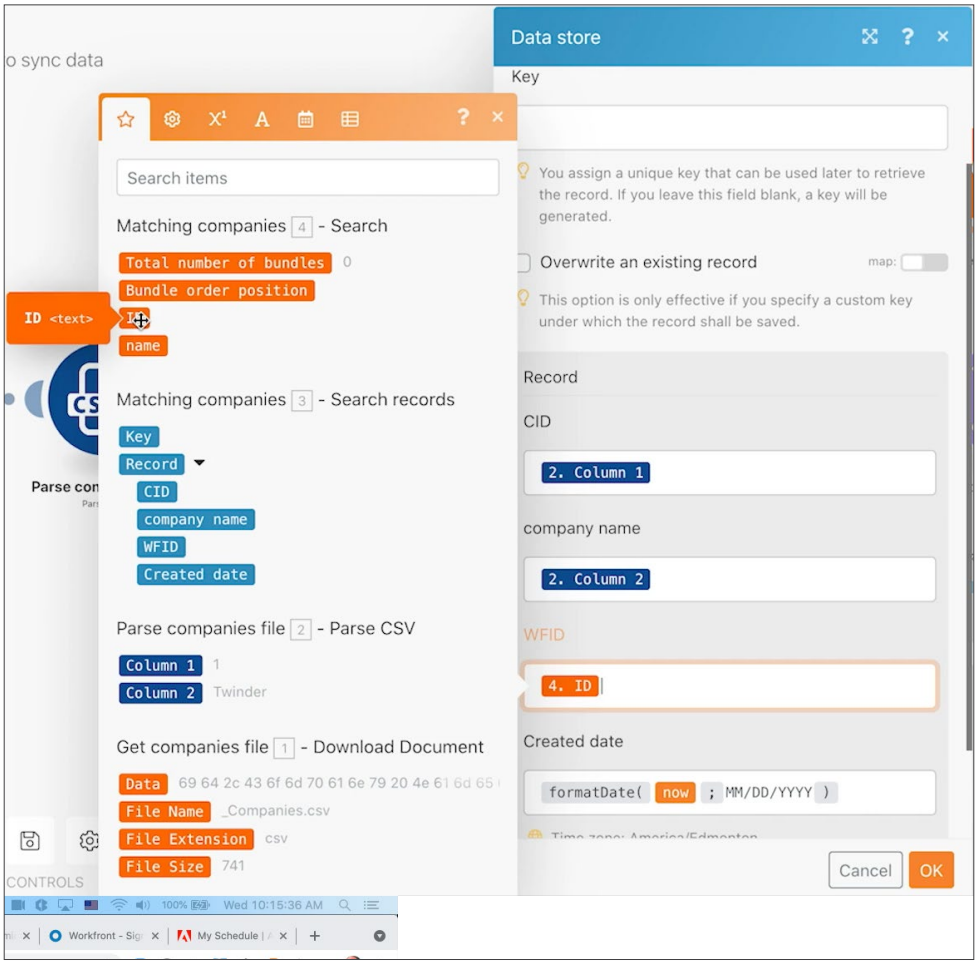


Data stores (cont.)

52. Now add another Add/replace a record module by cloning the one at the end of routing path 2.

53. Drag the cloned module into place at the end of routing path 3. Delete the empty module that was there.

54. Click on the cloned module. All fields should stay the same except the WFID field. Map it from the Matching companies Search module.



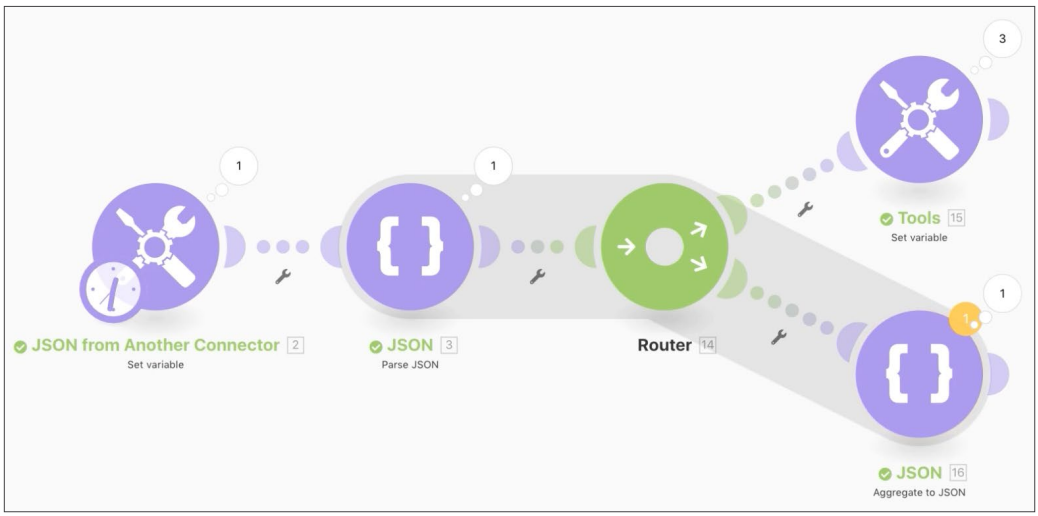
55. Click OK and rename this module "Create company entry."

Working with JSON

Learn how to create and parse JSON within a scenario to support your design needs.

Exercise overview

The purpose of this exercise is to conceptually show how to utilize information sent into a scenario in a JSON format, parsing it into fields and items that you can map throughout your scenario. Then you can either grab information from those mapped arrays or aggregate the information into JSON to then be sent to another system that expects JSON as a receiving input.



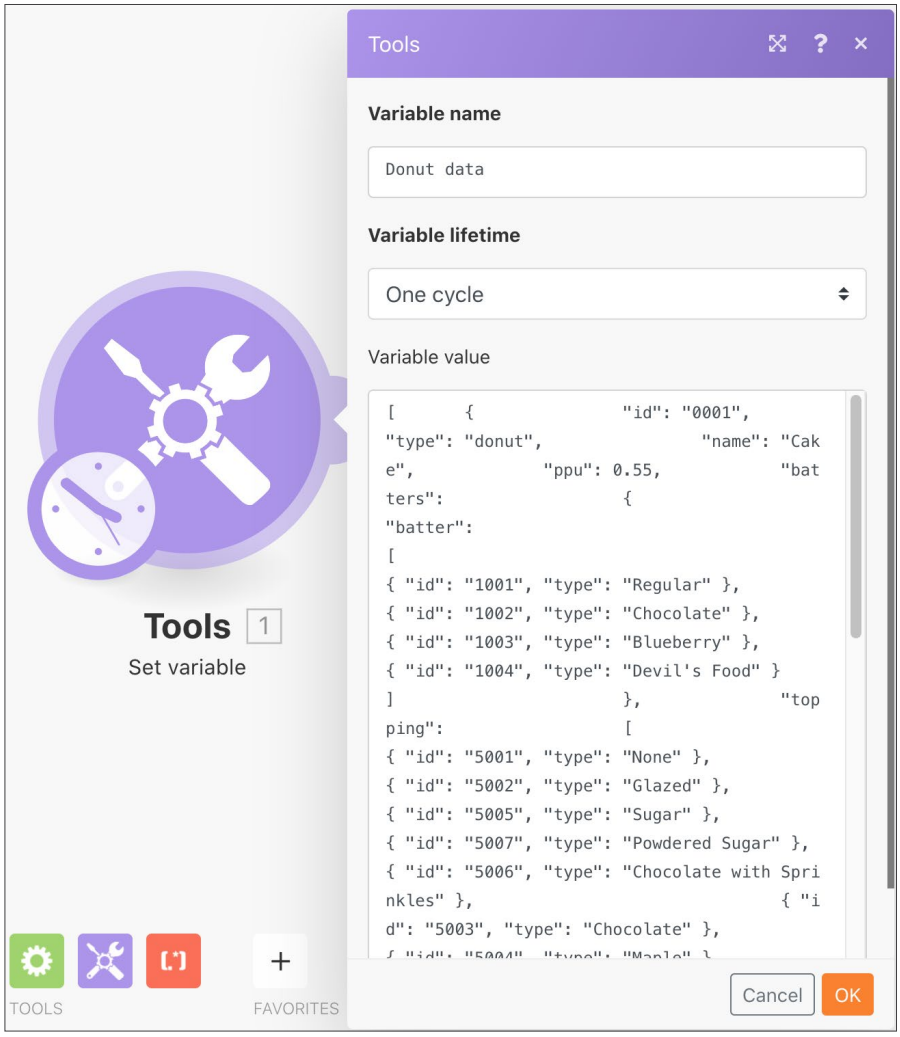
Steps to follow

Create a data structure and parsing JSON.

1. Create a new scenario and name it “Working with JSON donut data.”
2. For the trigger module, use the Set variable module.
3. For the Variable name, type in “Donut data.”

Working with JSON (cont.)

4. For the Variable value, copy and paste the contents of the "_Donut Data - Sample JSON.rtf" document found in the Fusion Exercise Files folder in your test drive.



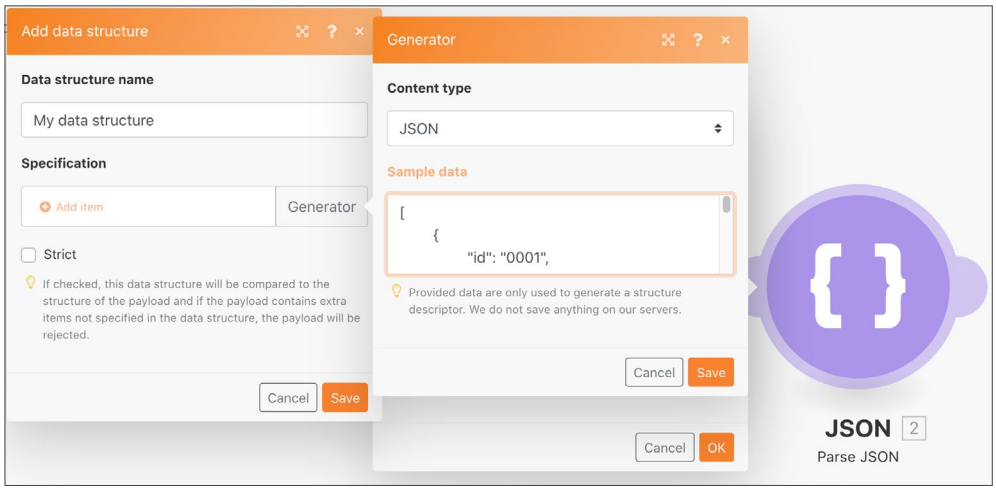
5. Rename this module "JSON from another connector."

6. Add a Parse JSON module.

7. Click Add for the Data structure field.

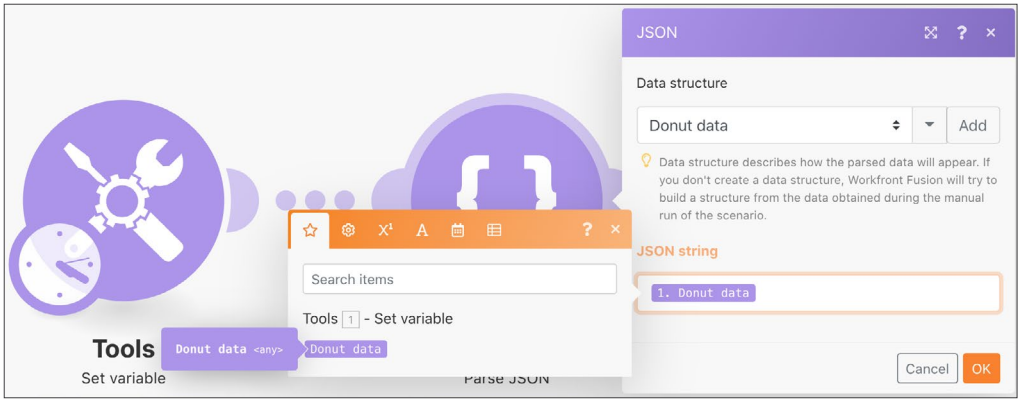
Working with JSON (cont.)

8. Select the Generator and paste the Donut Data - Sample JSON data that you copied into the Sample data field.



9. Click Save, naming the data structure "Donut data." Then click Save.

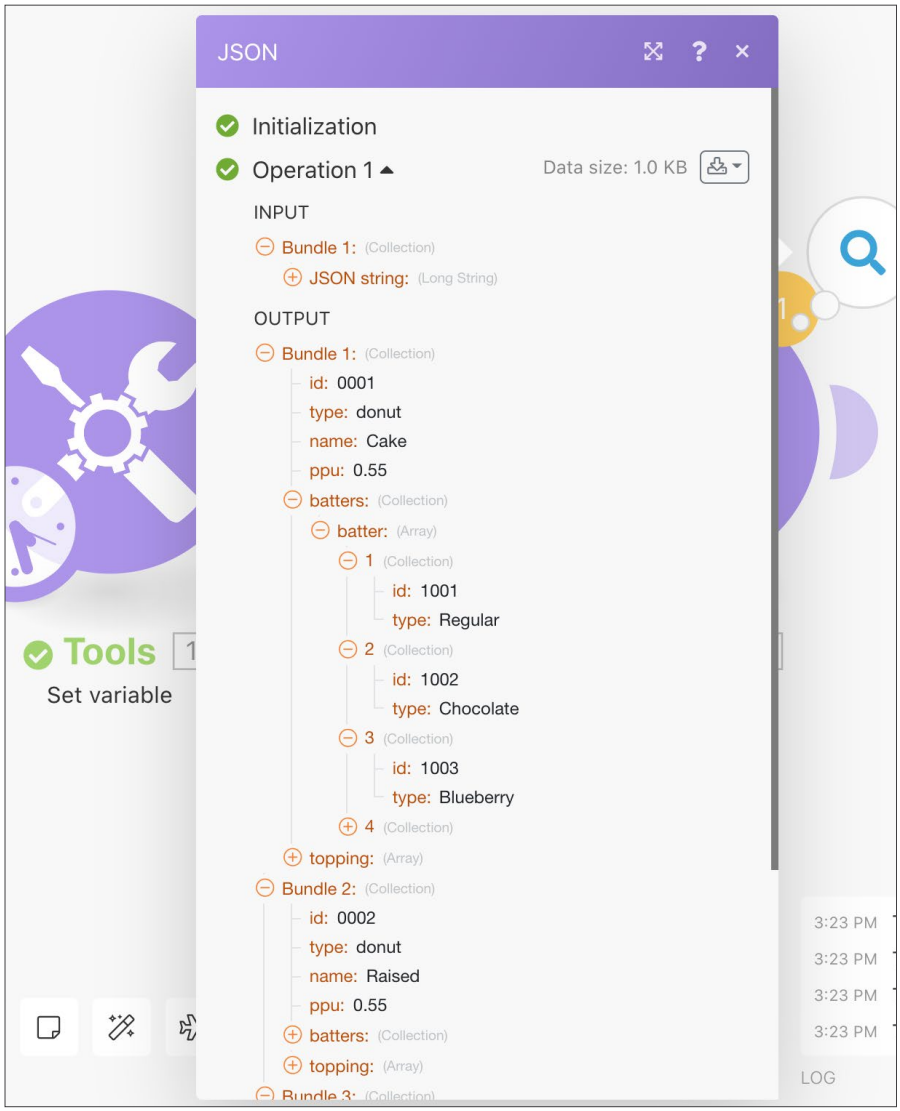
10. Map the Donut data from the Set variable module to the JSON string field.



11. Save your scenario, then click Run once to see the output.

Working with JSON (cont.)

The output of the Parse JSON module should look like this:



Working with JSON (cont.)

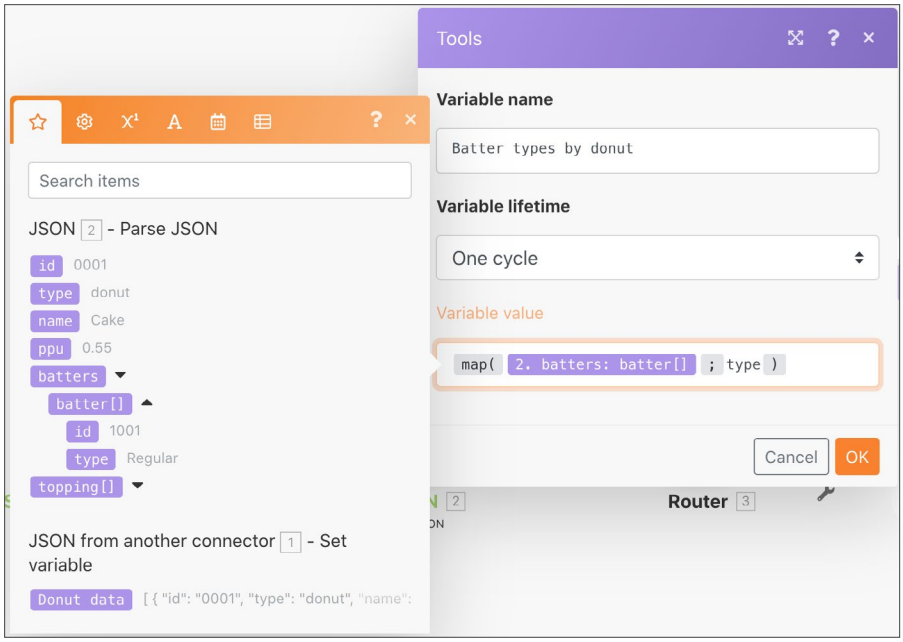
Map to specific array variables.

12. Add a router after the Parse JSON module.

13. In the top path, add a Set variable module.

14. For the Variable name, type “Batter types by donut.”

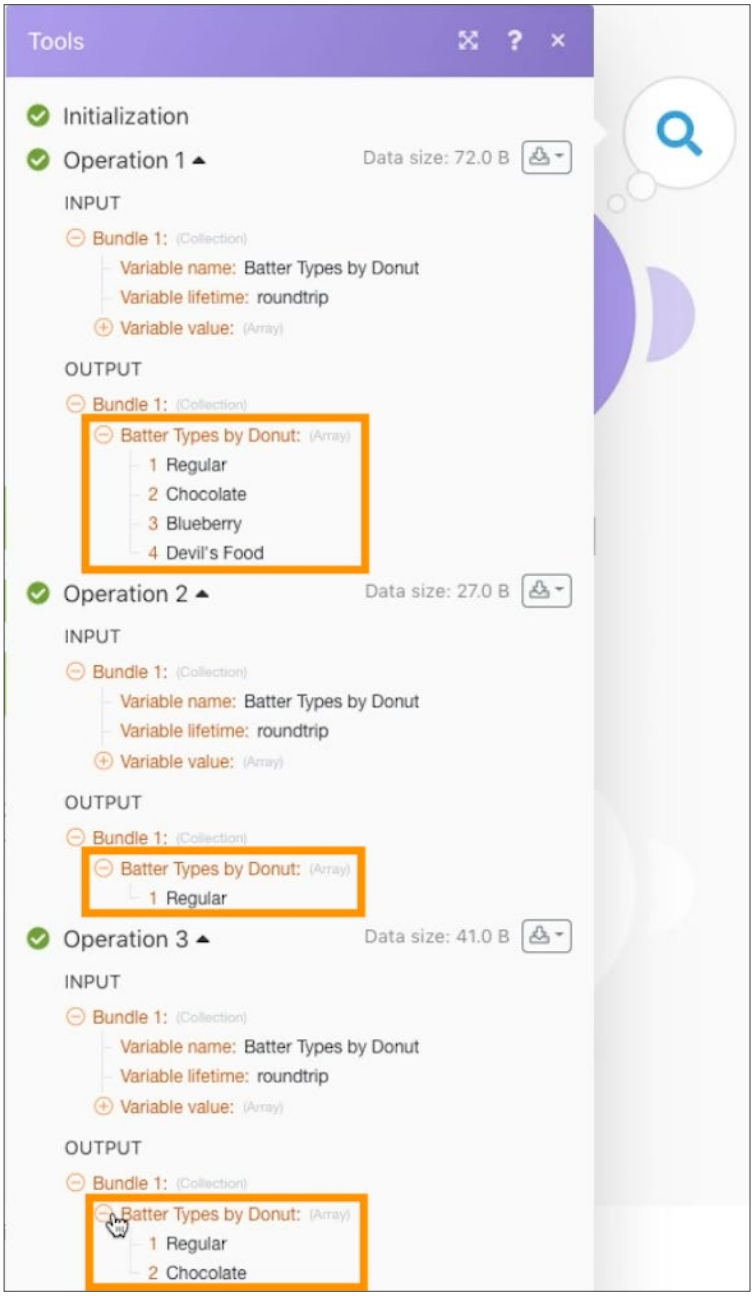
15. For the Variable value, use the map function to get the batter types from the batters array.



16. Click OK, then Run once.

Working with JSON (cont.)

17. Open the execution inspector to see the output bundle for each of the three operations, showing the batter types for each.



Aggregate scenario data to JSON.

18. On the lower routing path, add an Aggregate to JSON module.

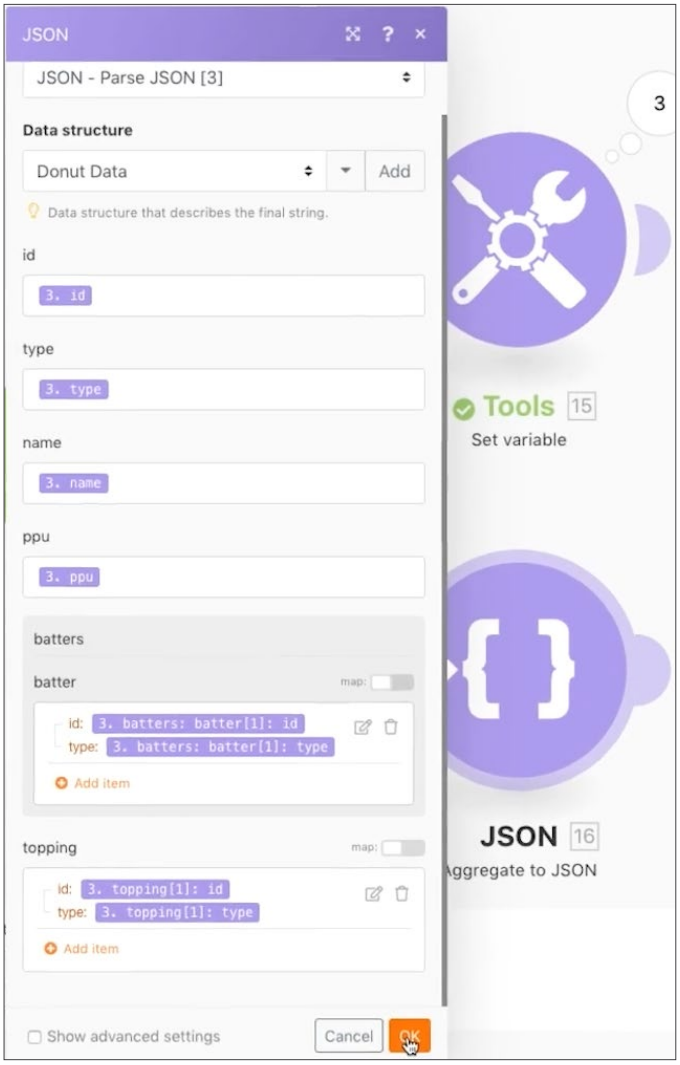
Working with JSON (cont.)

19. For the Source Module, choose the iterator—the Parse JSON module.

20. For the Data structure, create or choose any data structure. For this example, use Donut data.

21. Go ahead and map the fields over directly for this example, as shown below.

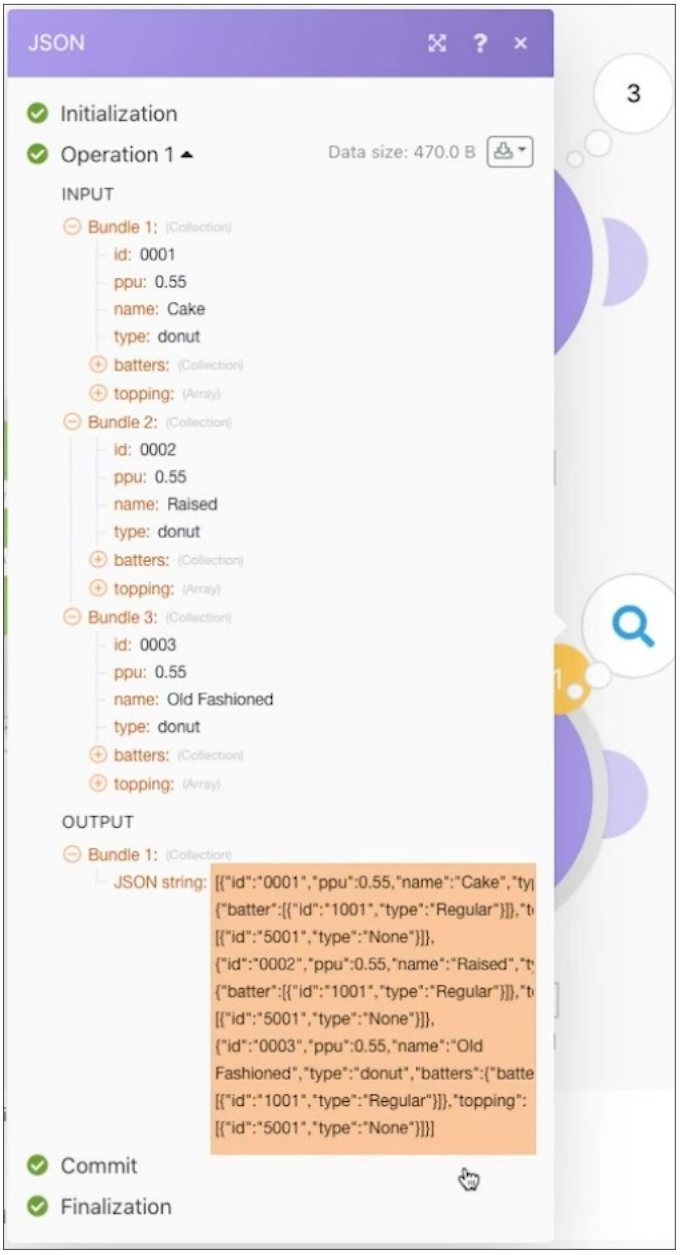
22. When you get to batter and topping, notice these are arrays, so you need to click Add item to map them.



23. Save the scenario and click Run once.

Working with JSON (cont.)

Look at the execution inspector for the Aggregate to JSON module and notice how you were able to aggregate three bundles into a single JSON string. You can then send this string to other systems that expect JSON.



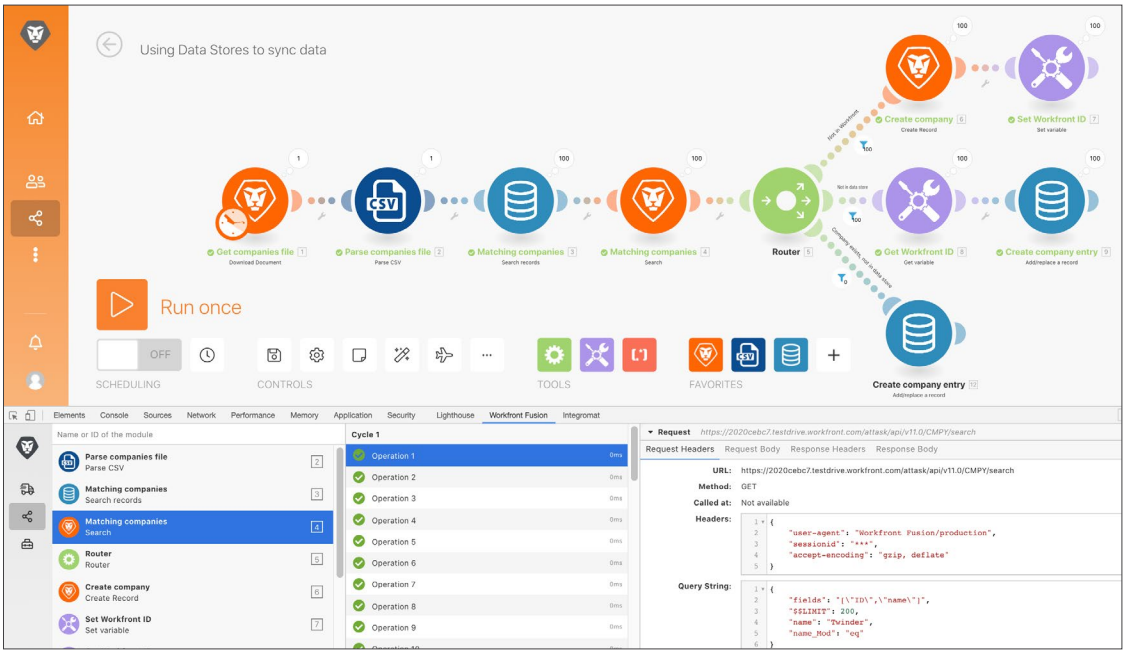
DevTool

Enhance your abilities to troubleshoot a scenario and ease complex configurations using the DevTool.

Exercise overview

Install and use the different areas in the Workfront DevTool to take a deeper dive into requests/responses made and advanced scenario design tricks.

Note: The Workfront Fusion DevTool is only available in the Chrome browser when using the **Chrome developer tool**.



Steps to follow

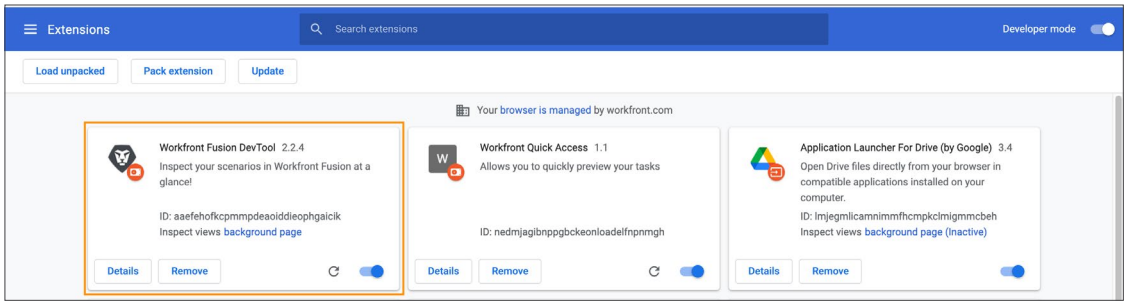
Install the DevTool.

1. Download the “workfront-fusion-devtool.zip” document found in the Fusion Exercise Files folder in the test drive.
2. Extract the Zip files to a folder.
3. Open a tab in Chrome and enter **chrome://extensions**.

DevTool (cont.)

4. Toggle on Developer mode using the switch at the top right, then click the "Load unpacked" button that appears at the top left. Select the folder containing the DevTool (this is where you unzipped it).

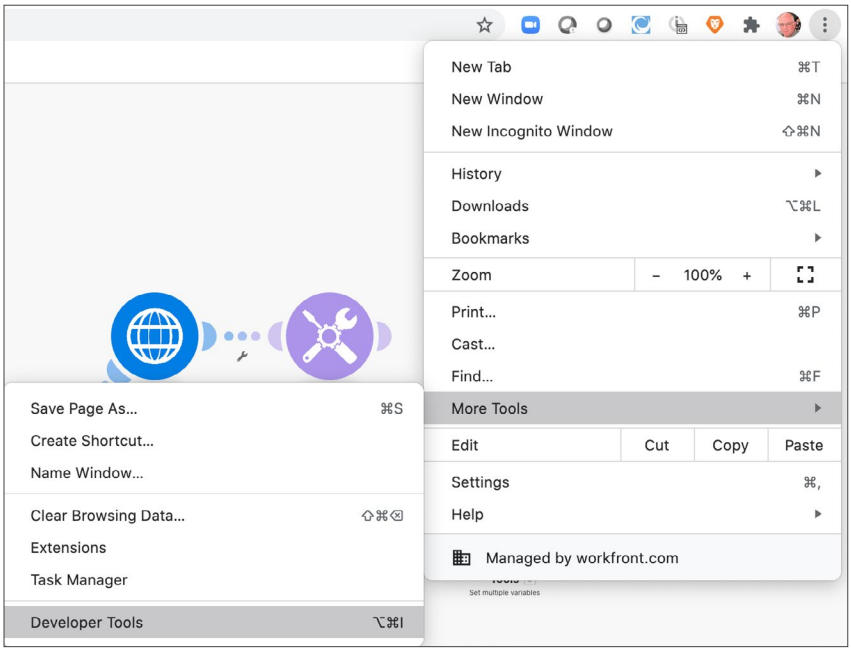
5. Once unpacked, the DevTool appears among your other extensions.



Use the Live Stream.

6. Start by opening the "Using Data Stores to sync data" scenario.

7. Open the DevTool by typing F12 or function F12. Or you can click the three-dot menu in the Chrome address bar and navigate to Developer Tools.

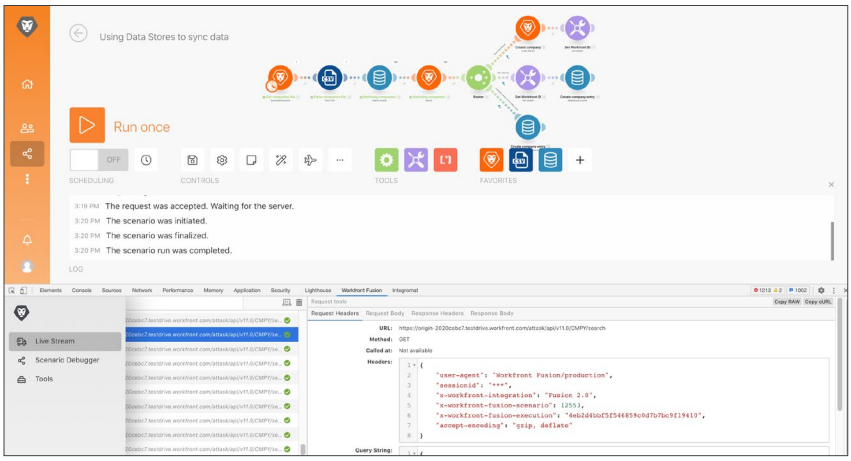


DevTool (cont.)

8. Click the Workfront Fusion tab, then select Live Stream from the list on the left.

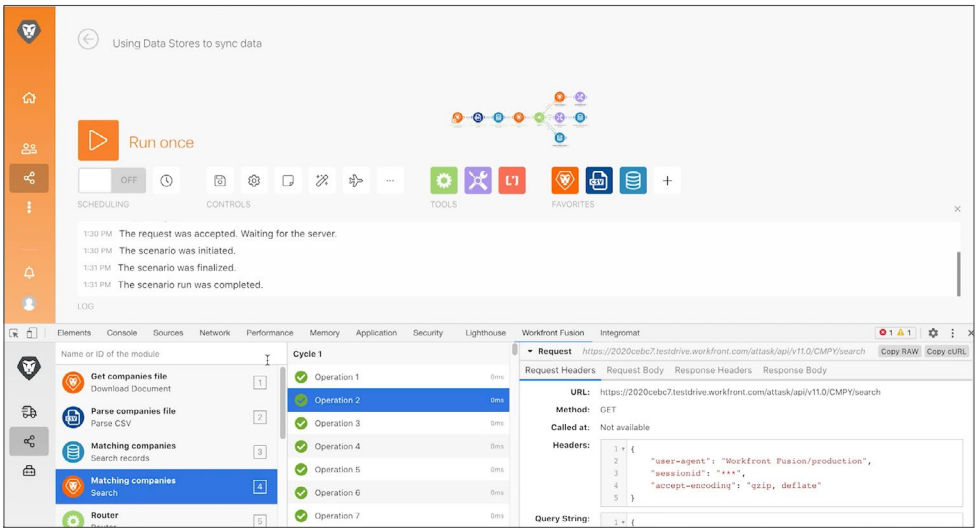
9. Click Run once to see events as they occur.

10. Click on an event to see tabs on the right for Request Headers, Request Body, Response Headers, and Response Body.



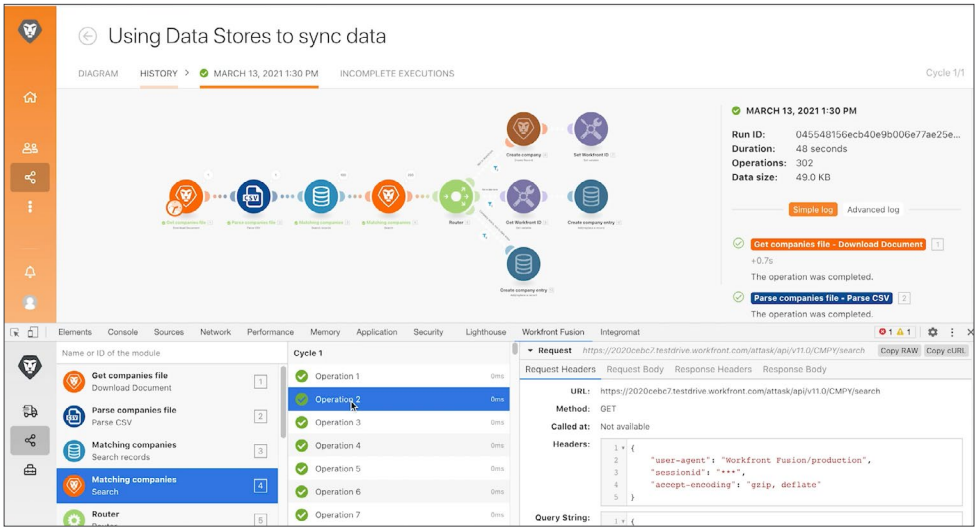
Use the Scenario Debugger.

11. Select Scenario Debugger and click a module to see information about the operations of that module.



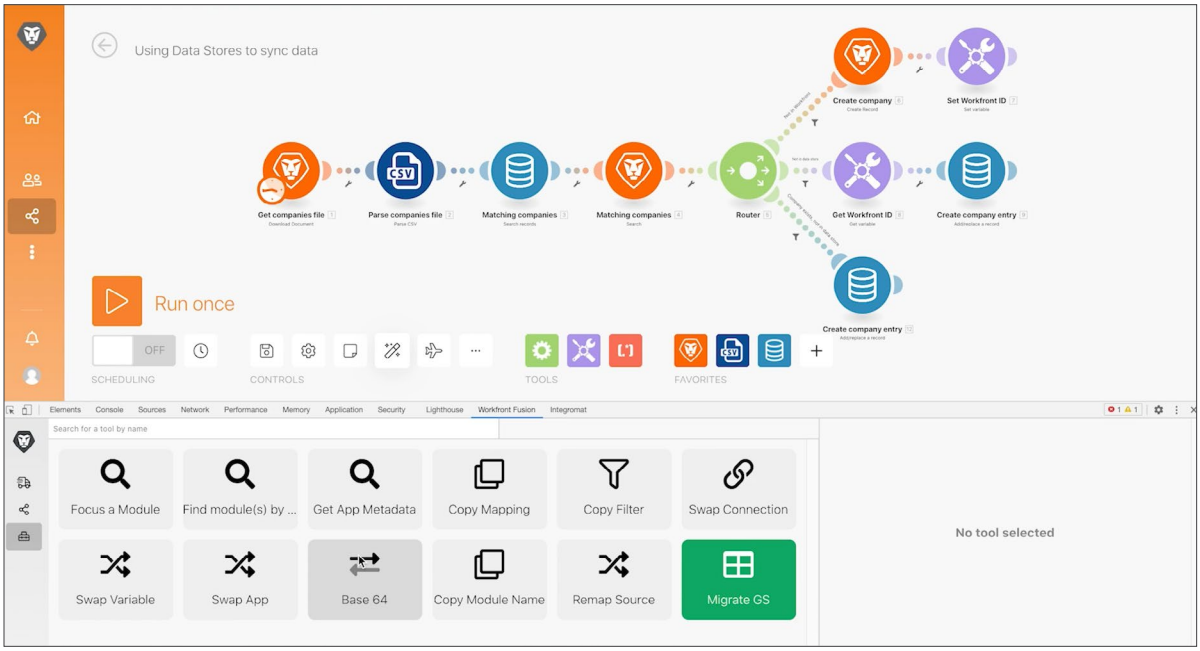
DevTool (cont.)

12. Navigate to the History tab. Click Details on an execution to examine module operation details for a specific execution.

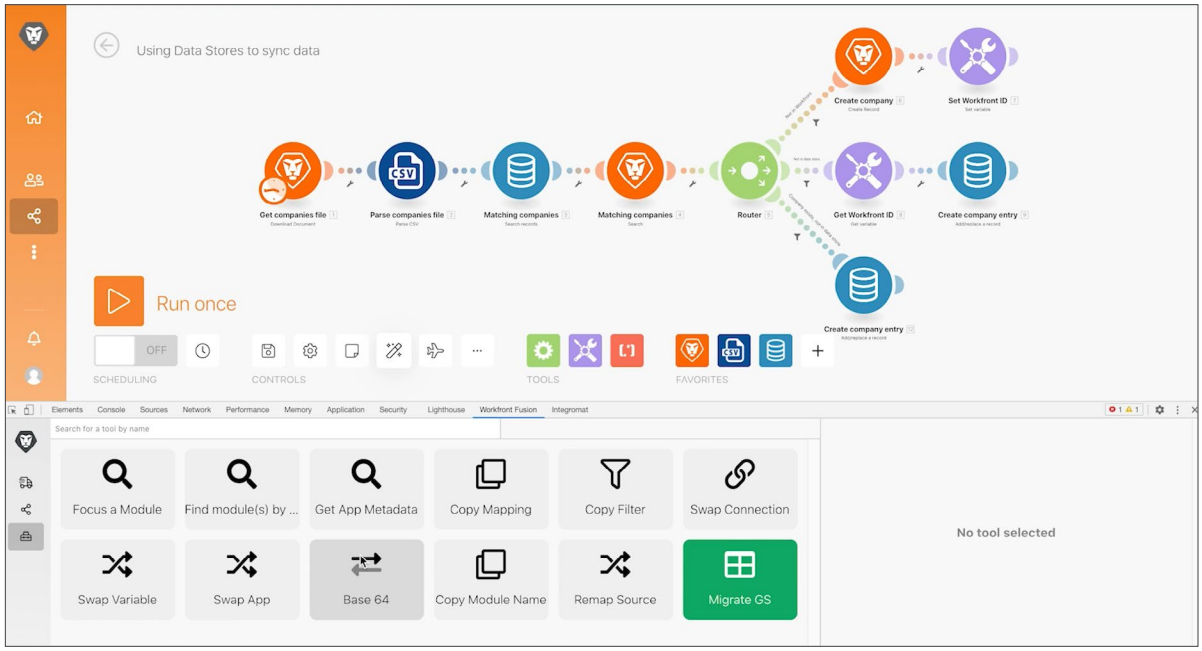


Use the Tools.

13. Go back to the scenario designer and select Tools in the DevTool. This displays the tools available.

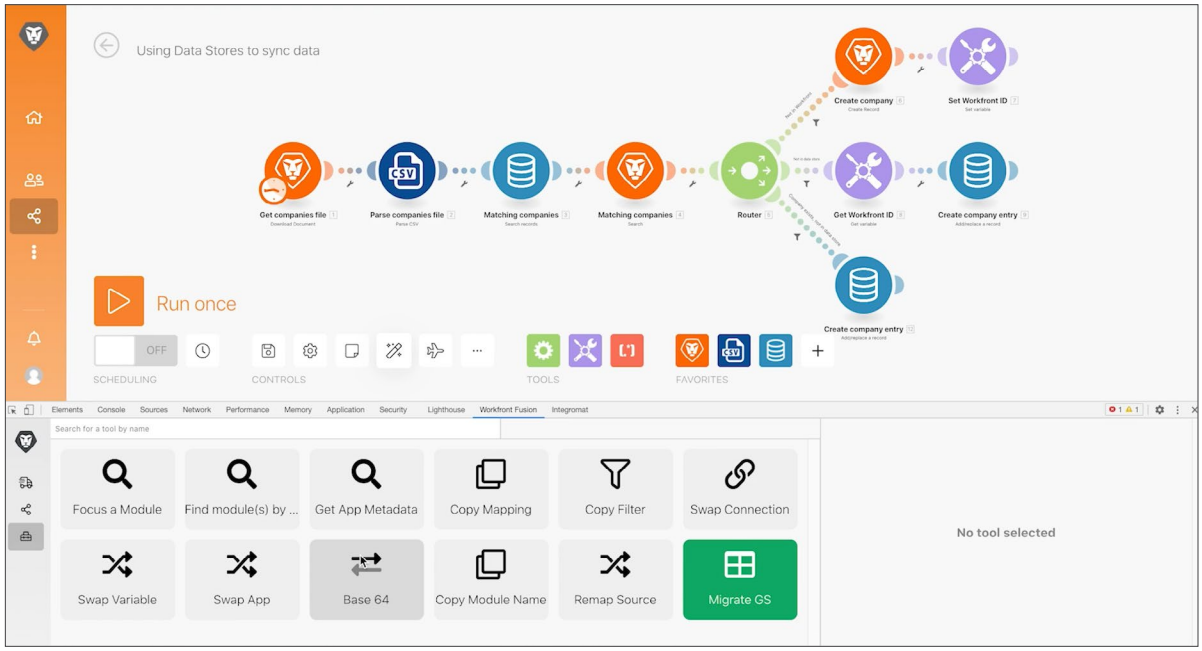


DevTool (cont.)



- Focus a Module—Find and open a module quickly by using the module ID.
- Find Module(s) by Mapping—Search a scenario using a keyword to find mapped values and/or keys in modules.
- Get App Metadata—See the metadata for the selected app a scenario.
- Copy Mapping—Copies mapping from one module to another. You also can clone the module in the designer.
- Copy Filter—Copies a filter. The filter is always assigned to the module on it's right.
- Swap Connection—The tool takes the connection from the selected module and sets the same connection to all modules of the same app in the scenario. This is helpful if you have to change the connection throughout a completed scenario. Avoid losing all mapping and save time by using this tool.
- Swap Variable—Finds all occurrences of the given variable across the whole scenario, or in one module, and replaces them with the new one. Wildcards are not supported. If you've accidentally mapped a value throughout the entire scenario, this can help you easily swap for the correct value.
- Swap App—Swaps the given app for another one.

DevTool (cont.)



- Base 64—Encode the entered data to Base64 or decode Base64. Useful when you want to search for particular data in the encoded request.
- Copy Module Name—Copies the selected module name to the clipboard.
- Remap Source—Change the mapping source from one module to another. You need to first add the module to use as a source module to the route in a scenario.
- Migrate OS—Made specifically to upgrade Google Sheets (legacy) modules to the latest Google Sheets version. It adds a new version of the module just after the legacy version of the module in the scenario route.