Using Functions and Calling APIs in a Flow

Table of Contents

1 Objective

In the previous exercise you created a flow that interacted with the user via the browser to collect information. This exercise will take this flow to the next level by adding function calls to test the data entered and call an external API to provide a robot check during the collection of the user data.

Tip

All the number instruction steps in each section must be followed to successfully complete this exercise. Any bullet points or images are for example only unless called out in a numbered step.

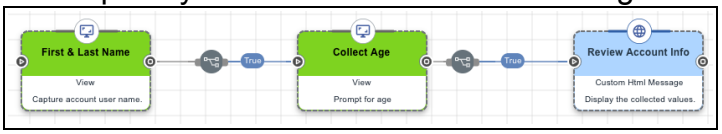
2 Do the following task items

In order to complete the flow started in the previous exercise the age must be checked to make sure the registering user is over the legal age. Additional information needs to be collected if they are old enough, which will happen after the age check. And finally a robot check needs to be done to make sure that somebody is not trying to run a script to create an account in the application.

2.1 Check the age entered

Now it is time to check the age that was entered is over the legal limit and if it is not stop the process.

At this point your flow looks like the following screenshot:

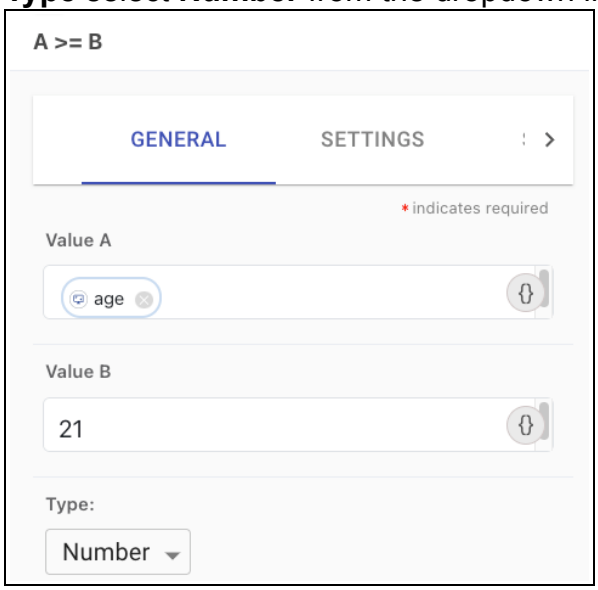


1 You need to insert a node to check age between **Collect Age** and **Review Account Info** therefore delete the **Action Decision** node between them.
2 Drag the **Review Account Info** node to the right to give you some space on the canvas.

3 Click right  connection icon of your **Collect Age** node holding the left mouse button and drag it to the right, then release the mouse button.
4 In the **filter/search** type **fun** then click **Functions** connector to add it to the canvas.
5 Click on the **Functions** node you added to the canvas.
   - This connection provides a list of standard functions that you can use in your flow.
   - It also allows the use of custom JavaScript (more on that later) to provide some scripting options in a flow.

   5.1 Click on the **gear** icon, you will see that there is nothing to configure for this connection.
   5.2 Click **Close** button to close the dialog.

6 Click **A >= B** capability and configure it as follows:
   6.1 **Value A** is a variable reference to the age from the **Collect Age** node.
   6.2 **Value B** is the number **21** which is the legal age.
   6.3 **Type** select **Number** from the dropdown list



   6.4 Click **Apply** button to apply your changes.
   6.5 Click **Settings** tab.
   6.6 **Node Title** is
      - **Age Verification**
   6.7 **Node Description** is
      - **Age >= 21**
   6.8 Click **Apply** button to apply your changes.
   6.9 Click **Close** to close the dialog.
7 Connect this node to the **Review Account Info** node.
8 Continue with the next section.

2.1.1 Test the age check

You will test the age with a legal age and one that is below the legal age.

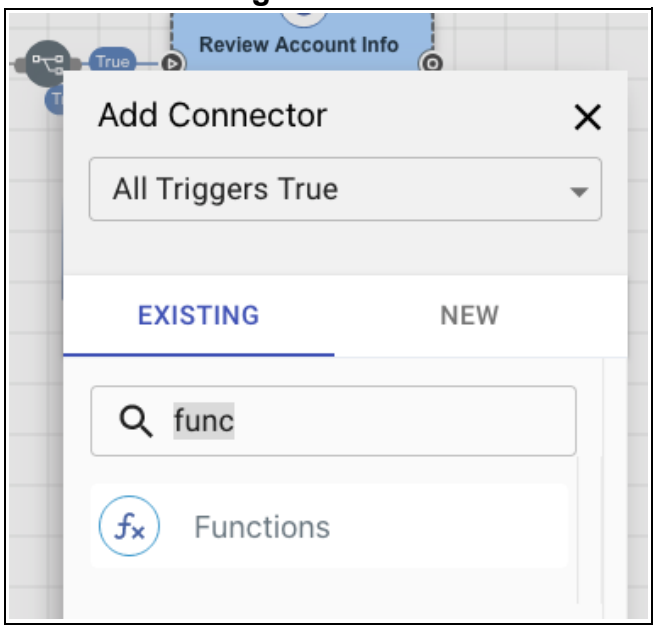
1 Click **Save** button at the top.
2 Click **Deploy** button at the top.
3 Click **Try Flow** button at the top.
4 Enter a first and last name then click **Next** button to proceed to the next node in the flow.
5 Enter an age value of **21** or greater then click **Next** button.
6 Do another test but this time enter a value of **18**.
   - Your flow just hangs and does not continue, this is because you do not have a false path only a true path from the function.
7 You can close the browser tabs you used to test.
8 Continue with the next section.

2.1.2 Add a false path to your flow for the age check

Now add a false path and display a message to the user that they are not of legal age.

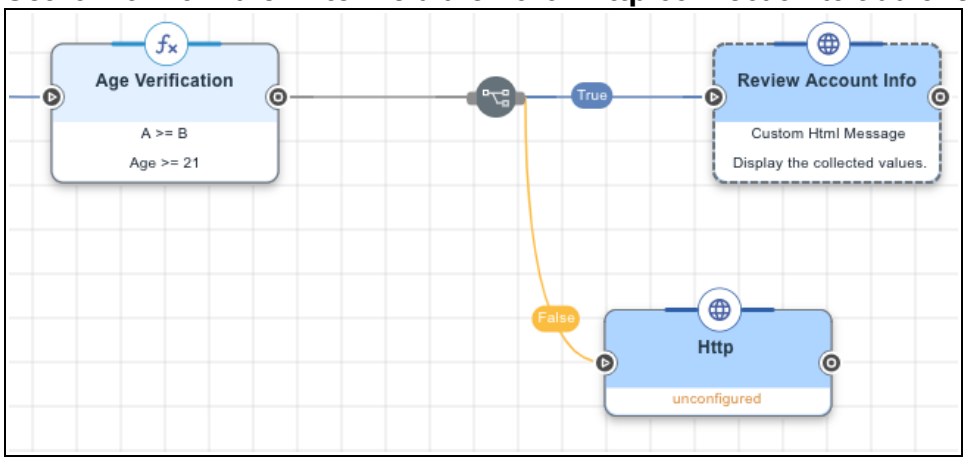
1 The little node that looks like a flow that is connecting two nodes has a dot at the right or left end that can be used to drag lines from/to it. See the following screenshot:

2 In the one after **Age Verification** click on the right dot and drag a line down and release it.



3 Select **Any Trigger False** from the dropdown list.

4 Search for **ht** in the **Filter** field then click **Http** connection to add one to the canvas on that path.



5 Click on the new **Http** node you just added to the canvas.

5.1 Click **Custom HTML Message** capability.

5.2 **Message Title** is

- **Sorry!**

5.3 **Message** is

- **You must be 21 to create a new account.**

5.4 Click **Apply** button to apply your changes.

5.5 Click **Settings** tab.

5.6 **Node Title** is

- **Not Legal Age**

5.7 **Node Description** is

- **Less than 21**

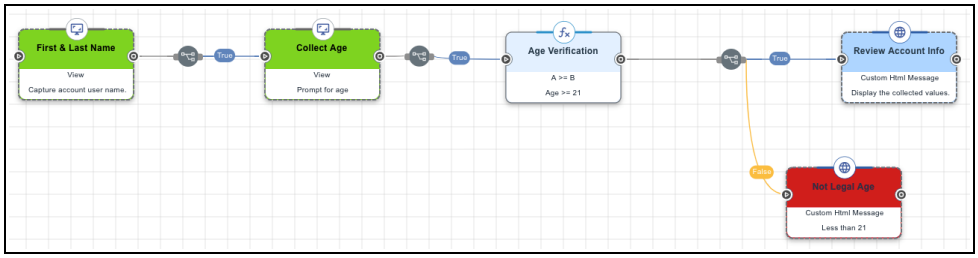
5.8 Click the color under **Node Background Color** to select a node color and select the **red** in the first row.

5.9 Click **Apply** button to apply your changes.

5.10 Click **Close** to close the dialog.

Tip

You can drag the nodes around to make your flow look less cluttered if nodes are to close together.

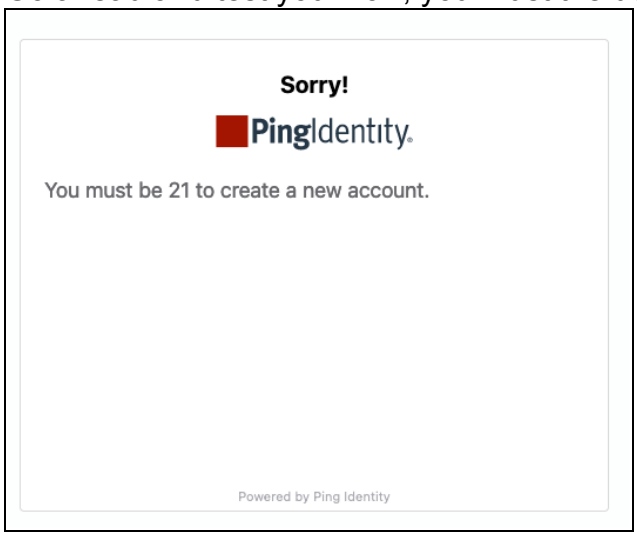


6 If needed click **Save** then click **Deploy** button to save your current flow.

Tip

You want to save often as you make changes to your flow. In future sections this step will be assumed and not called out. So don't forget to save.

7 Go ahead and test your flow, you must the deploy the current saved version to try it out.

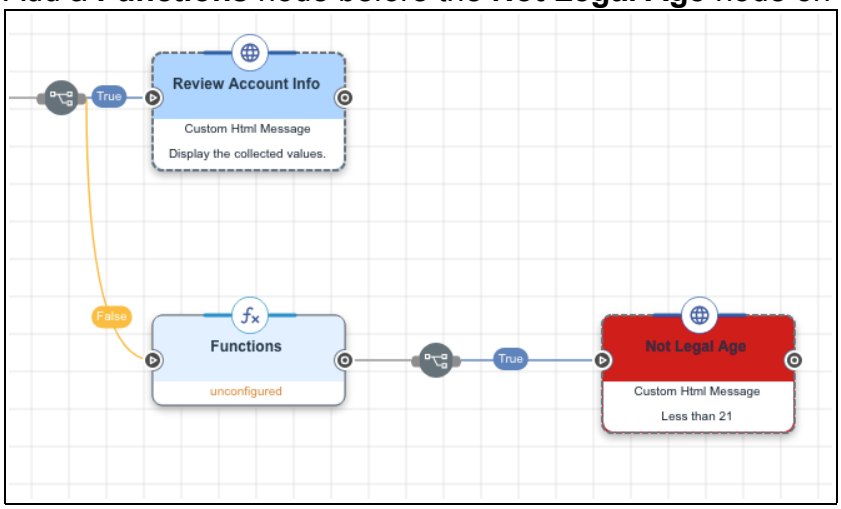


8 Continue with the next section.

2.2 Calculate age difference

The online gaming service is anxious to sign up new users, let's calculate how many years it will be until they can register.

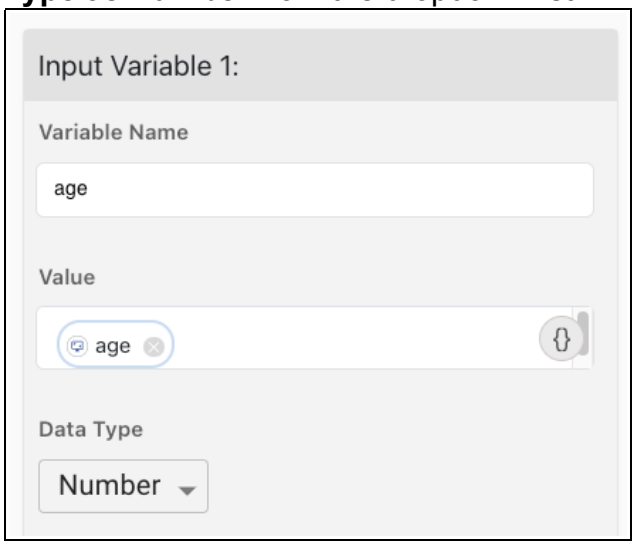
1 Add a **Functions** node before the **Not Legal Age** node on the false path.



- This will be a custom function that will use JavaScript to calculate the age difference.
- You can delete the false line to add the new node and then reconnect it, instead of deleting the entire Action Decision node.

2 Click on the new **Functions** node you just added to the canvas.
3 Select **Custom Function** as the capability
4 For our function to work, you'll need to pass in the age value. You can do this in the variables input list. Remember it is of type number.
   4.1 Add an a variable to **Variables Input List** section.
   4.2 **Variable Name** as
      - **age**
   4.3 **Value** is a variable reference to the **age** from the **Collect Age** node.

4.4 **Type** as **Number** from the dropdown list



5 Select **Code** field from the list, you will need to scroll down a bit.

- The field will provide some sample code, which you will replace with the code in the following steps.

Tip

You can use the  icon above text boxes to expand them to the entire browser window and then use the  icon to compress it back down to the original window.

5.1 In the **Code** text area delete the code that is listed by default. This sample provides a base example of how your function would accept parameters from DaVinci and how it would return a result. The base function below follows this pattern and calculates the age for you.

5.2 **Copy and paste** the following code in the **Code** text area.

```
module.exports = a = async ({params}) => {
    let agedelta = 21 - params.age;
    return {'agedelta': agedelta};
}
```

Tip

After you copy and paste you may need to adjust the code formatting so that it reads as shown; this makes for easier reading of the code.

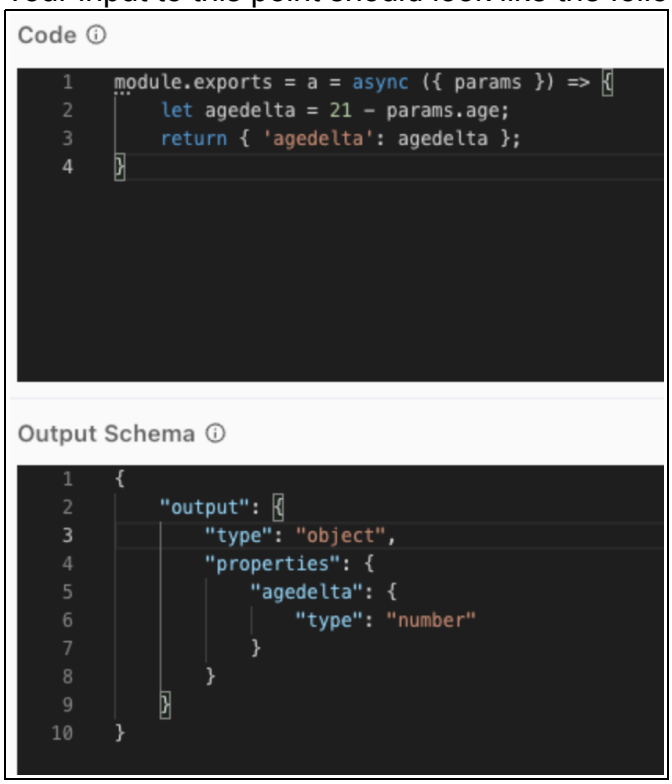
5.3 In the **Output Schema** text area delete the code that is listed by default.

5.4 **Copy and paste** the following code in the **Output Schema** text area.

```
{
  "output": {
      "type": "object",
      "properties": {
          "agedelta": {
              "type": "number"
          }
      }
  }
}
```

- This outlines the data that will be returned by your function and can be mapped into later nodes in the flow.
- You will see that `agedelta` is actually referenced in the code you entered in the previous steps.
- You can use the right-mouse button menu and the Format menu option to reformat the code to make it easier to read.

5.5 Your input to this point should look like the following:



6 In the **Settings** section enter the following:

6.1 **Node Title** as

- **Calculate Years**

6.2 **Node Description** as

- **Years until 21**

7 Click **Apply** button to apply your changes.

8 Click **Close** to close the dialog.

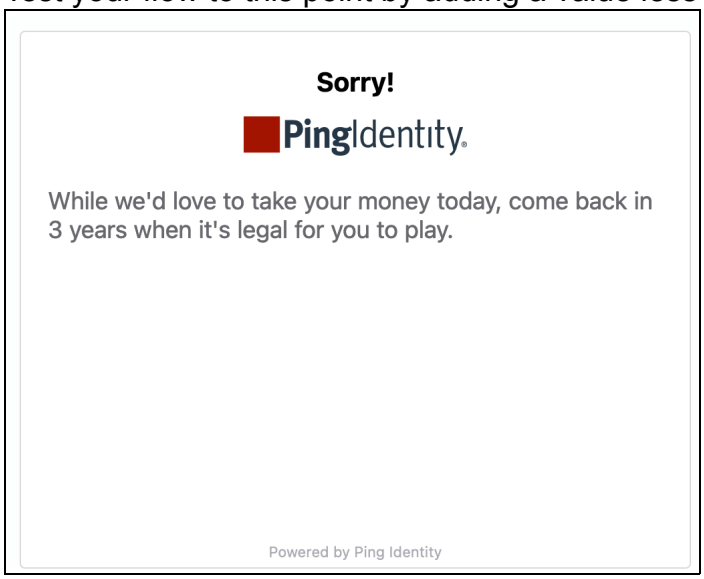
9 Continue with the next section.

2.2.1 Change the error message for under 21

Now it is time to change the message presented to the user when they are under 21. This will tell them know when they can come back to register.

1 Click **Save** button at the top if needed to save your changes to this point.
2 Click on the **Not Legal Age** node.
3 Replace the message text with the following:
   - **While we'd love to take your money today, come back in AGEDELTA years when it's legal for you to play.**
4 Replace the **AGEDELTA** in the message with a variable of **agedelta** from the **Calculate Years** functions node.

5 Test your flow to this point by adding a value less than 21 for the age.



6 Continue with the next section.

2.3 Collect email and password

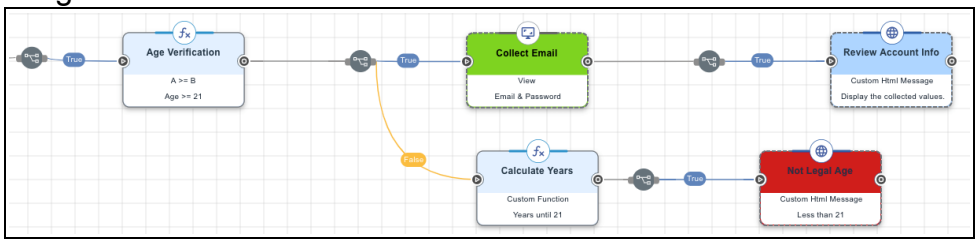
Now that you know the user is old enough to register, the application needs to collect the email and a password for the user. Although technically you're not saving this information (that will come later) for this exercise you want to have the flow be as complete as possible.

1 You will be adding the new Screen node before the **Review Account Info** node, so drag it over and create some space for the new node.

2 Right-mouse click on the line between the **Action Decision**  node and **Review Account Info** and **Delete** the true path.
3 Drag a new line from the **Action Decision** node and add a new **Screen** node.
   - This should be an *All Triggers True* path.

4 Select **View** for the new node's capability.
5 In the **General** tab configure it as follows:
  5.1 In the **Navigation Title** field type (or copy and paste) the following text.
    - **Email & Password**
  5.2 In the **Add a UI Component** dropdown select **Label** to add a field to the form.
    5.2.1 **Expand** the field you just added to configure it.
    5.2.2 In the **Property Name** field type **pageTitle**
    5.2.3 In the **Value** field type (or copy and paste) the following text.
      - **Almost done! Enter your email address and set a password.**
    5.2.4 The **Data Type** field is **String**
    5.2.5 Change **Alignment** field to **Center**
    5.2.6 The **Label Variant** field is **Title**
    5.2.7 Leave the default **Label Text Color**
    5.2.8 **Collapse** the field configuration.
  5.3 In the **Add a UI Component** dropdown select **Text Field** to add a field to the form.
    5.3.1 **Expand** the field you just added to configure it.
    5.3.2 For **Property Name** type **email**
    5.3.3 For **Display Name** type **Email**
    5.3.4 For **Data Type** select **String** from the dropdown list.
    5.3.5 Accept the defaults for the remaining configuration.
    5.3.6 **Collapse** the field configuration.
  5.4 In the **Add a UI Component** dropdown select **Text Field** to add a field to the form.
    5.4.1 **Expand** the field you just added to configure it.
    5.4.2 For **Property Name** type **password**
    5.4.3 For **Display Name** type **Password**
    5.4.4 For **Data Type** select **String** from the dropdown list.
    5.4.5 For **Secure** toggle it on, will be blue.
    5.4.6 Accept the defaults for the remaining configuration.
    5.4.7 **Collapse** the field configuration.
  5.5 Select the **Settings** tab.
  5.6 In the **Node Title** field type (or copy and paste) the following text.
    - **Collect Email**
  5.7 In the **Node Description** field type (or copy and paste) the following text.
    - **Email & Password**
  5.8 Click the color under **Node Background Color** to select a node color then select the **lite green** in the first row.
  5.9 Click **Apply** button to apply your changes, you should see the changes to the node in the canvas.

  5.10 Click **Close** to close the dialog.
6 Drag a line from the **Collect Email** node to the **Review Account Info** node to connect them.

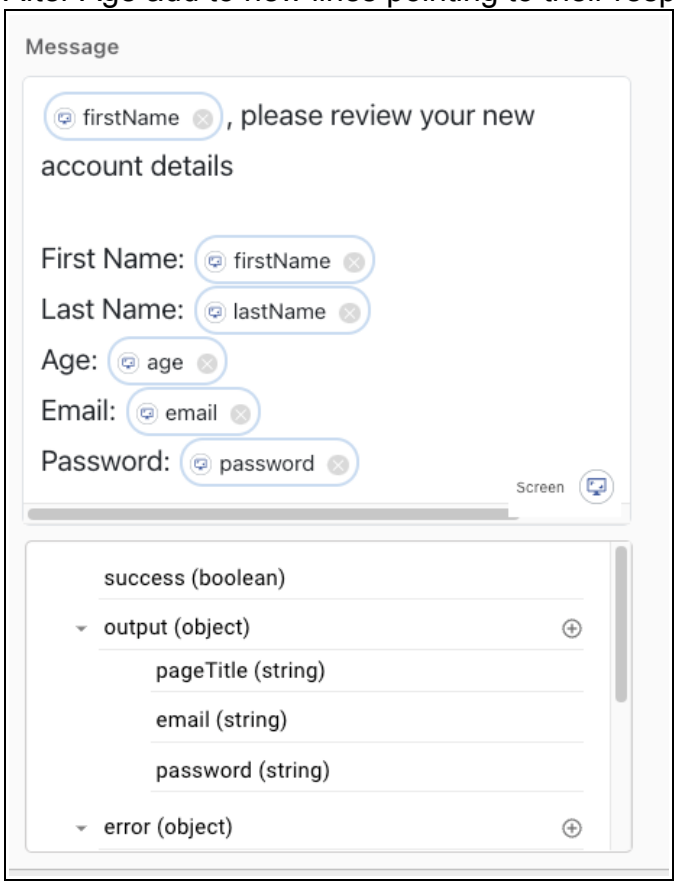


7 Don't forget to save.
8 Continue with the next section.

2.3.1 Update the review account info display

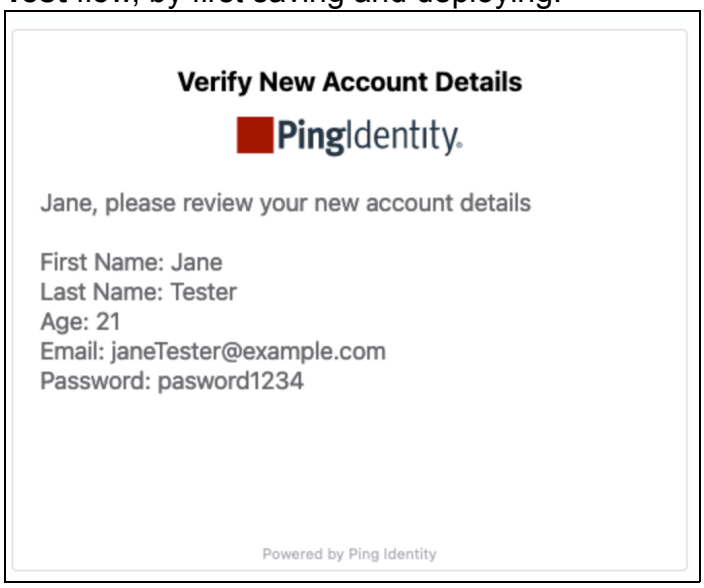
The additional data that is collected in the previous form needs to be displayed to the user. Note that you will be displaying the password but this is for testing only. In practice you should never do this, although you may allow an option to view it but never display as the default. Optional viewing of the password will be added when you start dealing with using CSS and having flows called by external applications.

1 Select the **Review Account Info** node to edit it.
2 After Age add to new lines pointing to their respective variables from the **Collect Email** node.



3 Apply and close the dialog.

4 **Test** flow, by first saving and deploying.



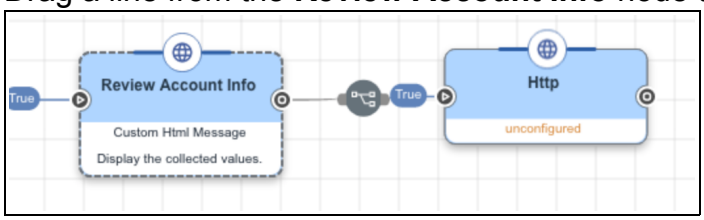
5 Continue with the next section.

2.4 Making an API call

Before you complete the registration process let's build a captcha like test to verify this is a real user.

Oftentimes in your flows, you'll need to make API calls to retrieve information that is needed in your flow. For today, you're going to be reaching out to the Deck of Cards API service to retrieve a random card that you can use as a simple test to verify this is a real user request. You'll retrieve the card's image URL, face value, and suit then have the user identify the card.

1 Drag a line from the **Review Account Info** node to add a new Http node.



2 Select the new node you just added to edit it.
3 Select **Make REST API Call** for the new node's capability.
   - You will take a look at the API call first before fully configuring the node.
4 In the **Settings** tab configure it as follows:
   4.1 **Node Title** as
      - **Deck of Cards**
   4.2 **Node Description** as
      - **API call.**
   4.3 Apply your changes to save them while you explore the API.
5 Continue with the next section.

2.4.1 Review the API options

Lets take a quick look at the API to test its input requirements and the output it produces.

You can find out about the API methods at https://deckofcardsapi.com/, which is what you will test in the following steps.

1 Open a new browser tab.
2 In the address bar enter https://deckofcardsapi.com/api/deck/new/draw/
   - This will return a JSON response and depending on your browser and extensions you will see something like the following your actual result will vary.


```
{
  "success": true,
  "deck_id": "so67g6wcd35a",
  "cards": [
      {
          "code": "5C",
          "image": "https://deckofcardsapi.com/static/img/5C.png",
          "images": {
              "svg": "https://deckofcardsapi.com/static/img/5C.svg",
              "png": "https://deckofcardsapi.com/static/img/5C.png"
          },
          "value": "5",
          "suit": "CLUBS"
      }
  ],
  "remaining": 51
}
```

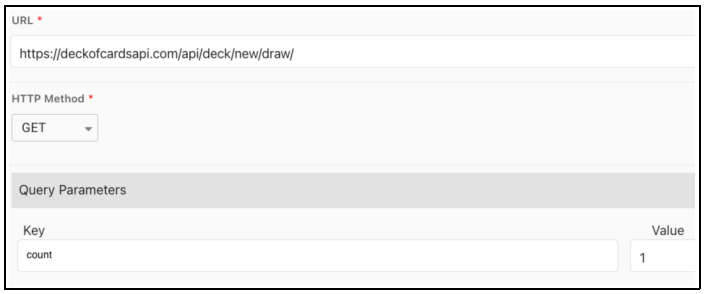

3 You want only one card so you can pass the parameter **count** with a value of 1, as follows: https://deckofcardsapi.com/api/deck/new/draw/?count=1
   - This is the API call that will be used in the Http node in your flow.
4 Continue with the next section.

2.4.2 Configure the node to call the API

Now that you know what the API returns and what method is to be used; time to configure the node in your flow.

1 Back on the browser tab with your DaVinci flow.
2 On the **General** tab enter the following:
   2.1 **URL** as
      - **https://deckofcardsapi.com/api/deck/new/draw/**
   2.2 **HTTP Method** is **GET** from the dropdown list.

2.3 In **Query Parameters** click the **+** to add an entry:
  2.3.1 **Key** is
    - **count**
  2.3.2 **Value** is
    - **1**

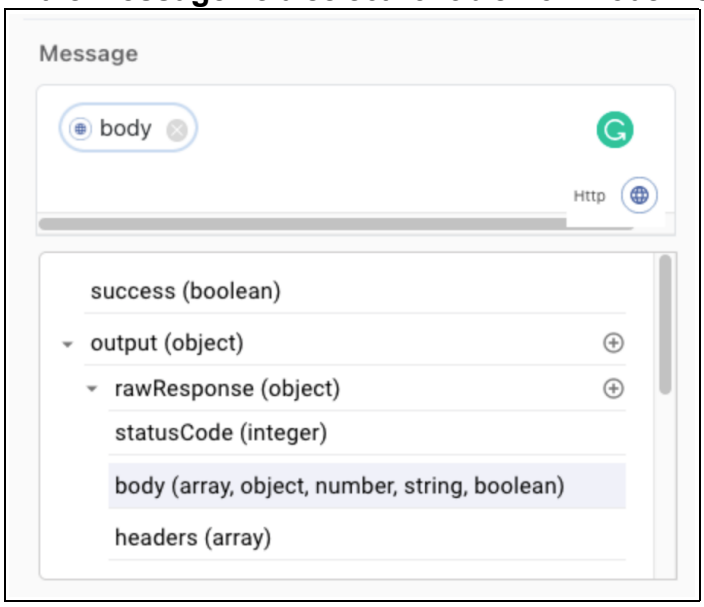


3 Apply your changes then close the dialog.
4 Continue with the next section.

2.4.3 Setup to display API output

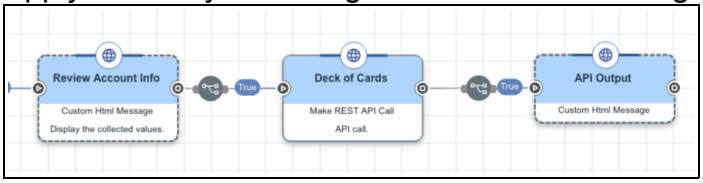
Often when you make an API call you'll want to see what's returned. So let's add another HTTP connector and display the API result.

1 Drag a line from the **Deck of Cards** node to add a new Http node.
2 Select the new node you just added to edit it.
3 Select **Custom Html Message** as the capability.
4 In the **Message** field select variable from node **Deck of Cards** called **body** as shown below:



5 For node title use **API Output**

6 Apply to save your changes and close the dialog.

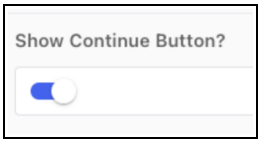


7 Continue with the next section.

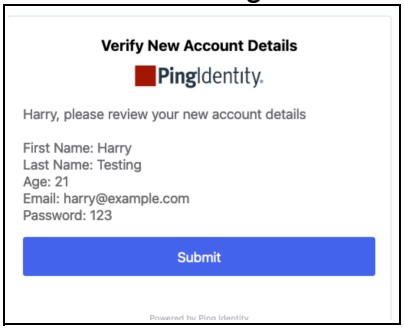
2.4.3.1 Test flow with API call

Now it is time to test the flow and see the results of the API call. But first the Review Account Info node needs a continue button.

1 Click the **Review Account Info** node in your flow.
2 Scroll to the bottom of the **General** tab.
3 Select the **Show Continue Button** toggle.



4 Apply your changes.
5 Now time to test your flow.
   - Your review dialog now has a submit button.



   - And you should see the results of the API call.

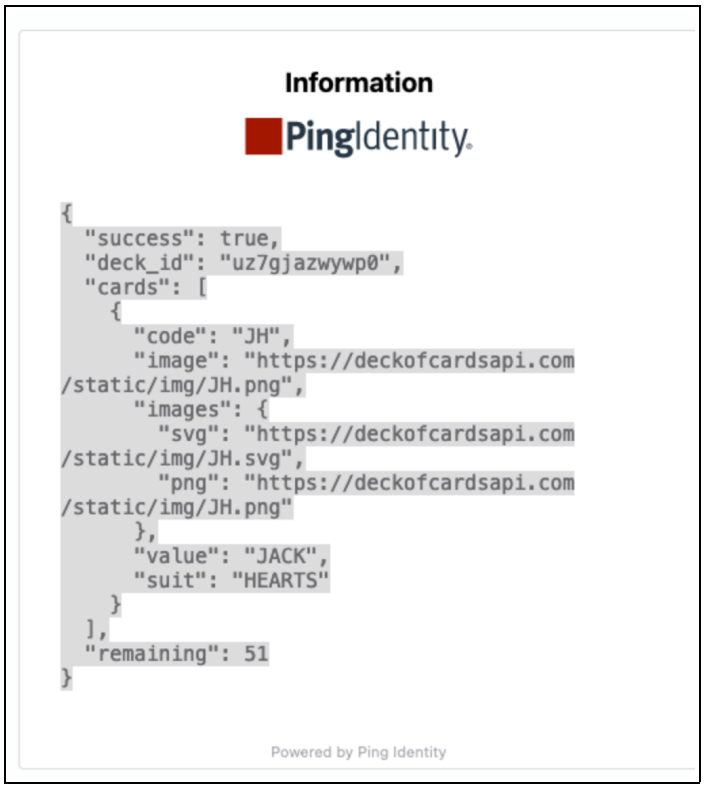
6 Continue with the next section.

2.4.4 Parse the API response

You could use the returned values by using JSON dot notation to fetch the image PNG URL and card values, however DaVinci allows you to supply a success response schema that makes retrieving and using results so much simpler.

In order to accomplish this you need to generate a JSON schema and you can use the generator at the following link: JSON Schema Creator. This will take you to the Ping Library where you can find many helpful resources to build your flows. You may want to save this for future reference.

1 **Highlight and copy** the JSON output on the final page from your test run in the previous section.



2 Open the URL https://library.pingidentity.com/share/fe782908-191c-4a3d-8771-0062171bc665 in a new browser tab.
3 Click the **Clear** button to the right of **JSON Object** text area to clear it.

4 **Paste** the JSON you copied into the **JSON Object** text area on the site.

5 Click **Create Schema** button, under the text boxes.

- You will see the generated schema in the JSON Schema text area.

6 Click the **Copy** button to the right of **JSON Schema** text area.

7 In your DaVinci tab open the **Deck of Cards** node.

8 Scroll down until you see the **Success Response Schema** block.

9 **Paste** the schema into this block.
10 Apply and close the dialog box.
11 Continue with the next section.

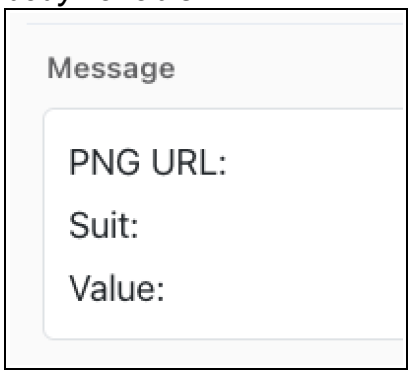
2.4.4.1 Use result from schema in API Output

Now it is time to change the API Output node to use the schema to display the values you are going to use later in this exercise. Then you will test your flow.

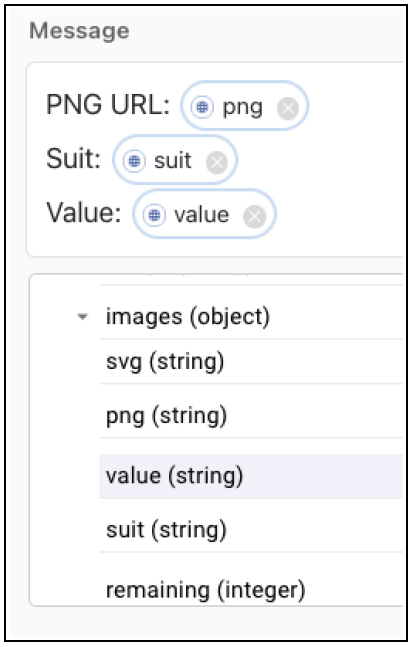
1 Click the **API Output** node in your flow.
2 You are going to edit the message to pull specific data from the previous node using the provided schema, using the following steps:

2.1 In the top right of the dialog there is a icon that will open the dialog as full width of your browser **click** it.
- This can make it easier to work with this dialog for some use cases. Use it whenever you find that you need more space. You can always toggle it back to the original view.

2.2 Edit the **Message** block as shown in following screenshot, removing the existing entry of *body* variable.



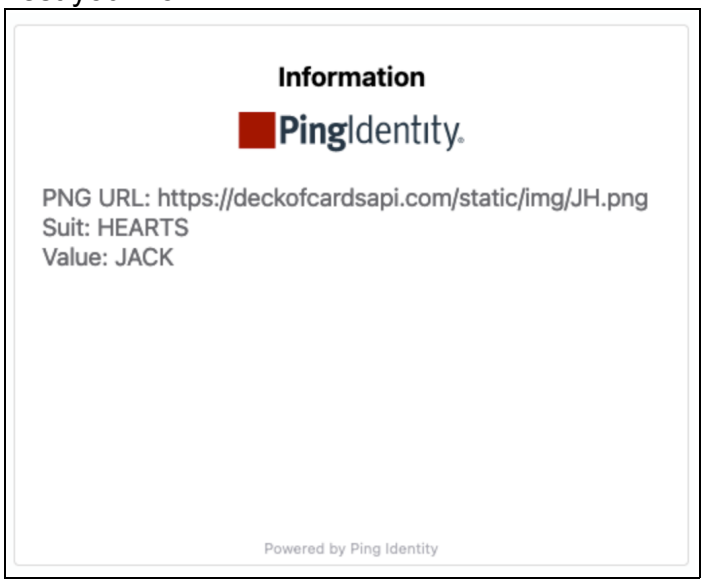
2.3 From the **Deck Of Cards** node select the variables as shown in following screenshot.



- The variables you are looking for will be under **output rawResponse body cards (array)** as you navigate through the variables.

3 Apply your changes and close the dialog.

4 Test your flow



- Your actual return values will vary from what is shown above.

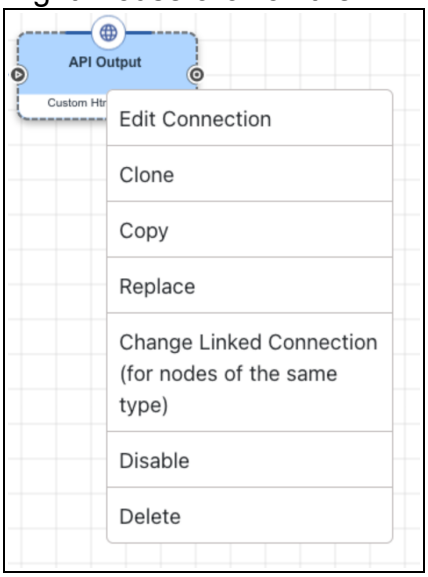
5 Continue with the next section.

2.5 Check for a robot process

In this section you will deal with providing the results to the user from the API call and having them select what card was shown to them. If they do this successfully then you assume they are not a robot and will then complete the process of creating their account.

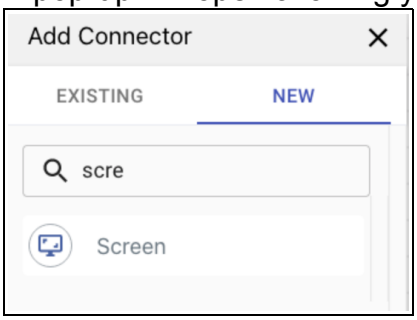
The first step is to display the API result to the user and ask them to identify which card is presented to them.

1 Right-mouse click on the **API Output** node in your flow.

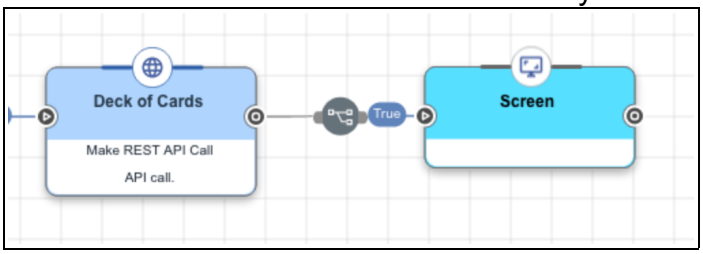


2 Select **Replace** from the dropdown list.
   - This will allow you to replace the existing node in the flow with a new one keeping it in the path of the existing flow.
   - When you replace a flow its internal node ID will remain the same so references variables in other nodes may still work, but you should verify them anyway.

3 A pop-up will open allowing you to select a connector, type **scre** to start the search.



4 Select the **Screen** connector to add it to your flow, replacing the existing Http node.

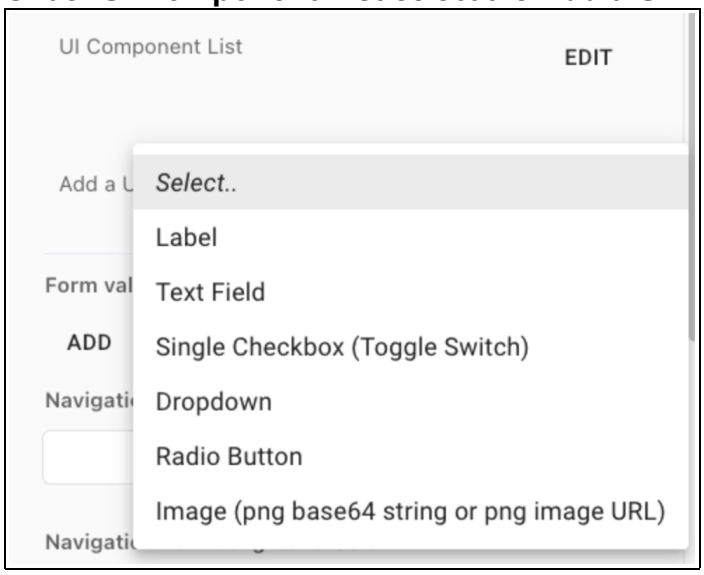


5 Click on the **Screen** node on your canvas.
6 Select the **View** capability.
7 Continue with the next section.

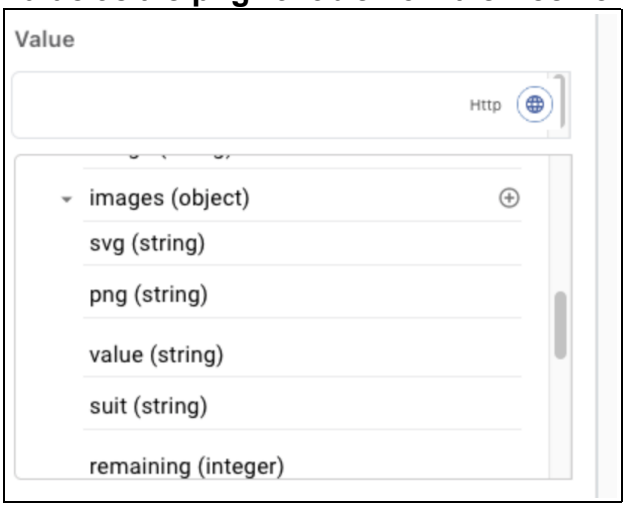
2.5.1 Configure the UI for the user

In this section you will configure the UI components that will present the card image to the user and request they identify what suit and value for the card presented to them.

1 Under **UI Component List** select the **Add a UI Component** dropdown.

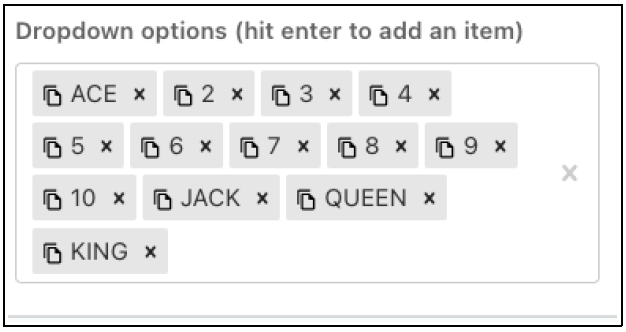


2 You will be doing this three times in the following steps:
- 2.1 First one is select **Image** from the list, configure it as follows:
  - 2.1.1 **Property Name** as
    - **cardimage**
  - 2.1.2 **Value** as the **png** variable from the **Deck of Cards** node.



  - 2.1.3 **Image Type** as **URL** from the dropdown list.
  - 2.1.4 **Image Content Mode** as **Center** from the dropdown list.
  - 2.1.5 **Image Alignment** as **Center** from the dropdown list.
  - 2.1.6 Remaining entries take the defaults.
- 2.2 Second one is select **Dropdown** from the list, configure it as follows:
  - 2.2.1 **Property Name** as **cardvalue**
  - 2.2.2 The value will not be provided, you want the user to select it.
  - 2.2.3 **Display Name** as **Card Value**
  - 2.2.4 **Dropdown Options** will be the following in order, hit enter after each to add it to the list. Case is important.
    - 2.2.4.1 **ACE**
    - 2.2.4.2 **2**
    - 2.2.4.3 **3**
    - 2.2.4.4 **4**
    - 2.2.4.5 **5**
    - 2.2.4.6 **6**

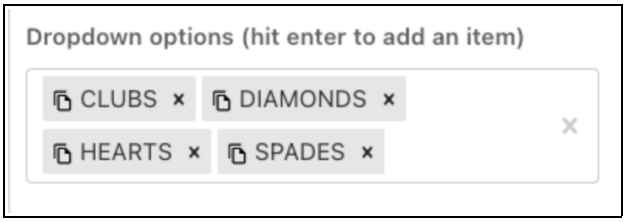
2.2.4.7 **7**
2.2.4.8 **8**
2.2.4.9 **9**
2.2.4.10 **10**
2.2.4.11 **JACK**
2.2.4.12 **QUEEN**
2.2.4.13 **KING**



- Your result should look like the above screenshot.

2.2.5 Remaining entries take the defaults.

2.3 Third one is select **Dropdown** from the list, configure it as follows:
2.3.1 **Property Name** as **cardsuit**
2.3.2 The value will not be provided, you want the user to select it.
2.3.3 **Display Name** as **Card Suit**
2.3.4 **Dropdown Options** will be the following in order, hit enter after each to add it to the list. Case is important.
2.3.4.1 **CLUBS**
2.3.4.2 **DIAMONDS**
2.3.4.3 **HEARTS**
2.3.4.4 **SPADES**



- Your result should look like the above screenshot.

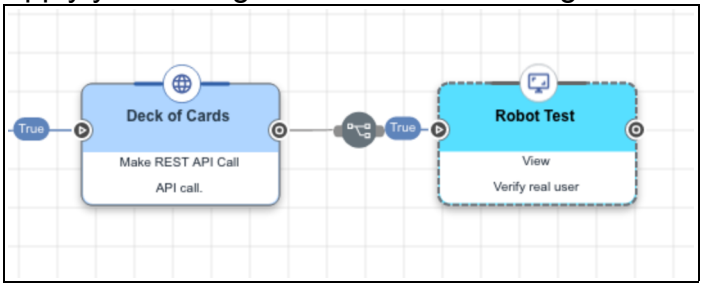
2.3.5 Remaining entries take the defaults.

3 Set the **Navigation Title** to
- **Please identify the card**

4 Under **Settings** set:
4.1 **Node Title** to
- **Robot Test**

4.2 **Node Description** to
- **Verify real user**

5 Apply your changes and close the dialog.

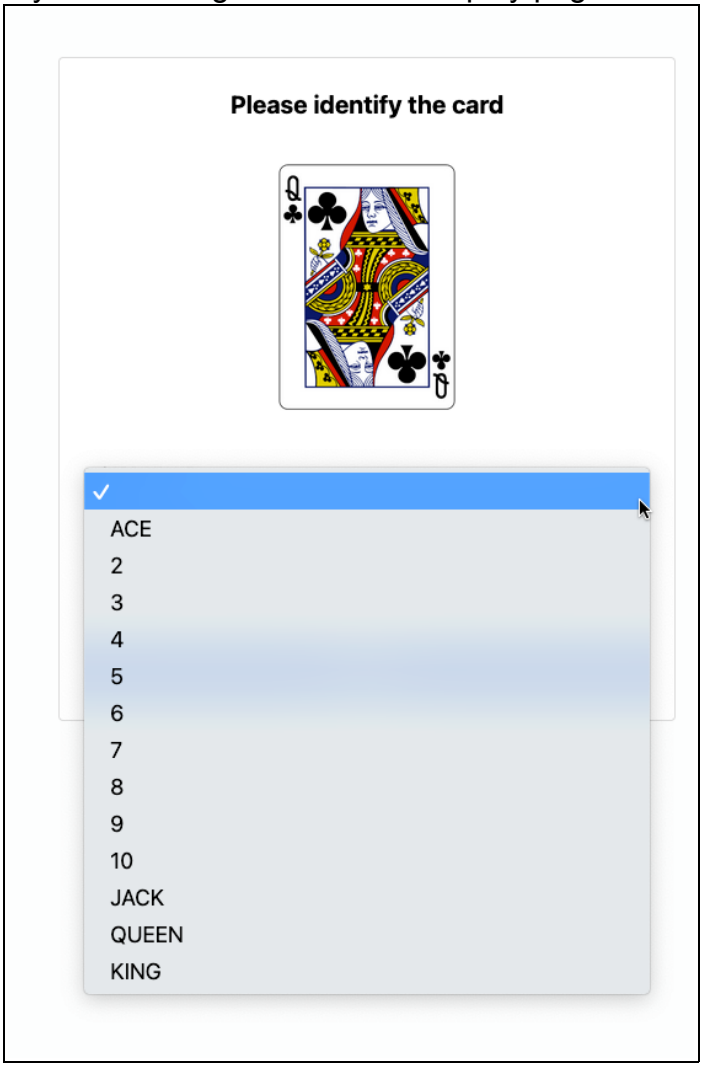


6 Continue with the next section.

2.5.2 Test the API call and user prompt

Now it is time to test your flow to the point of displaying the results of the API call to the user and requesting their input.

1 Save and deploy your canvas changes to this point.
2 Try the flow to get to the card display page.



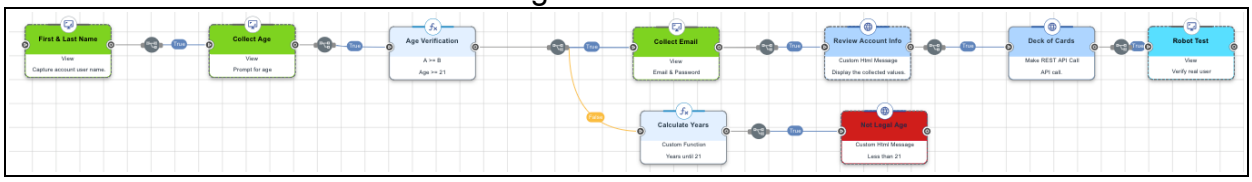
- Your test will most likely be different. Verify that the inputs are correct for the two dropdowns.

3 Continue with the next section.

2.6 Complete the account creation in the flow

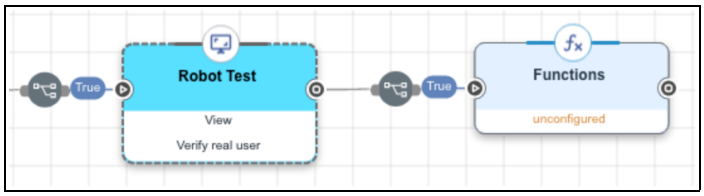
Now it is time to complete the flow to verify the input from the robot test and complete creation of the account. Of course at this time you will not be actually creating an account, just simulating the end of the process.

1 Your flow should look like the following at this time:



2 From the **Robot Test** node drag a line to right and release mouse button.

3 Search for **func** then select the **Functions** connector.

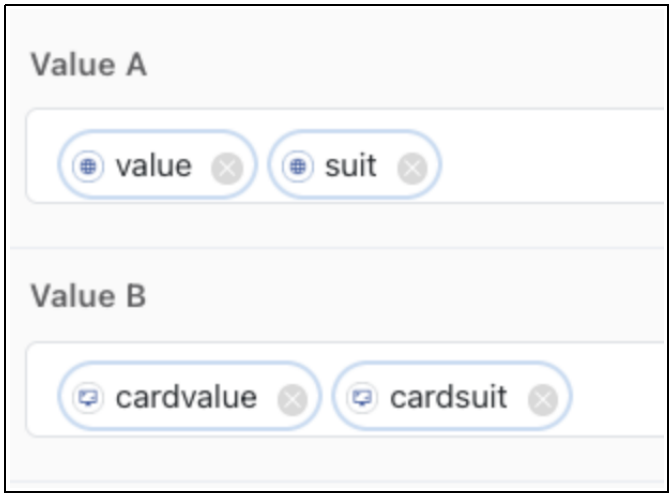


4 On the node you just added select the **A == B** capability and configure it as follows:

> **Tip**
>
> You can use the  icon in the top right of the dialog to expand the dialog to the entire canvas and make it easier to edit its configuration.

4.1 For **Value A** select the **value** and **suit** variables from the **Deck of Cards** node.

4.2 For **Value B** select the **cardvalue** and **cardsuit** variables from the **Robot Test** node.



- The order of the variables being compared is important for each of the values, so you are comparing the same type of data.

4.3 Under **Settings**

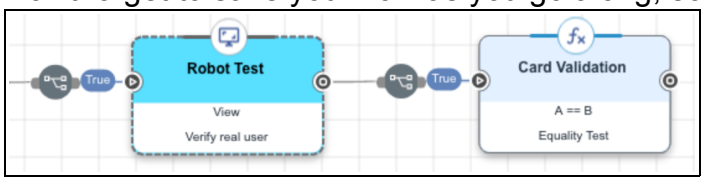
4.3.1 **Node Title** as

- **Card Validation**

4.3.2 **Node Description** as

- **Equality Test**

5 Apply your changes and close the dialog.
6 Don't forget to save your flow as you go along, so you don't lose your changes.

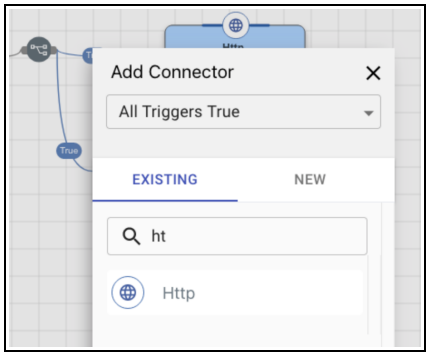


7 Continue with the next section.

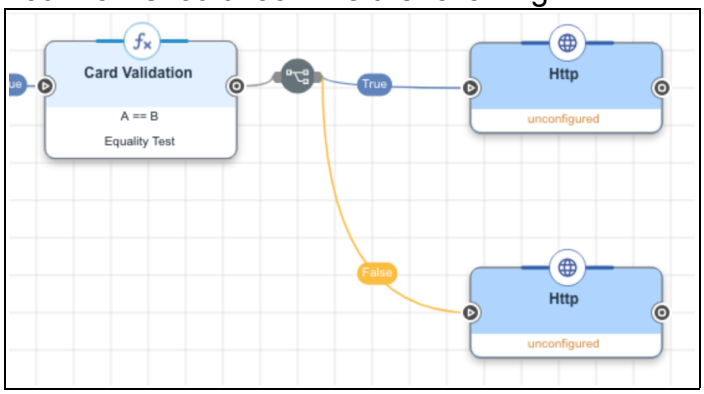
2.6.1 Define two paths for card validation

After the user identifies what card is selected there are two potential paths. They can get it right and be successful or they can fail (or you could have a robot on your hands) so now time to define and handle those two results. In this section you will first define these two paths and following sections configure the nodes with response to the user/robot.

1 From the **Card Validation** node drag a line to left and release mouse button.
2 At the top of the dialog the **All Triggers True** must be the option.
3 Search for **ht** then select the **Http** connector to add the node.
4 From the **Action Decision** node drag a new path and release the mouse button.



5 At the top of the dialog select **Any Trigger False** from the dropdown.
6 Select the **Http** connector to add the node.
7 Your flow should look like the following:



8 Continue with the next section.

2.6.1.1 Configure the true path

Now it is time to configure the user path to create an account as the user passed the robot test.

1 Click on the **Http** node on the **true** path after **Card Validation** node.
2 Select **Custom Html Message** capability.
3 Configured it as follows:
   3.1 **Message Title** as
      - **Account Created!**
   3.2 **Message** as

```
Your Beginner's Luck Online Casino & Gaming account has been successfully created!
♣♣Happy Gaming♣♣
```

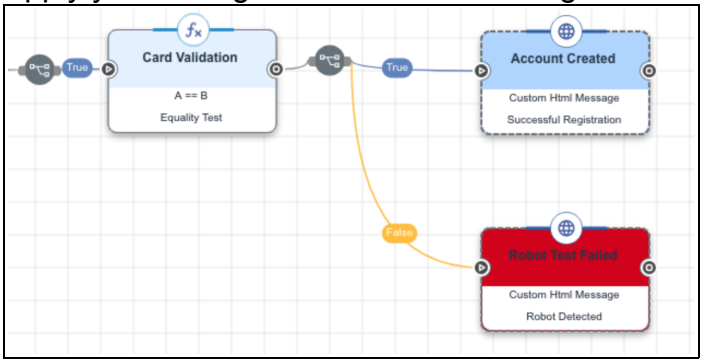
      - You may need to add the line feed to get it on two lines if your copied and pasted.
   3.3 **Node Title** as
      - **Account Created**
   3.4 **Node Description** as
      - **Successful Registration**
4 Apply your changes and close the dialog.
5 Continue with the next section.

2.6.1.2 Configure the false path

Now it is time to configure the robot path to reject the creation of the account.

1 Click on the **Http** node on the **false** path after **Card Validation** node.
2 Select **Custom Html Message** capability.
3 Configured it as follows:
   3.1 **Message Title** as
      - **Robot Detected**
   3.2 **Message** as
      - **Does not compute!**
   3.3 **Node Title** as
      - **Robot Test Failed**
   3.4 **Node Description** as
      - **Robot Detected**
   3.5 **Node Background Color** as **red**.

4 Apply your changes and close the dialog.

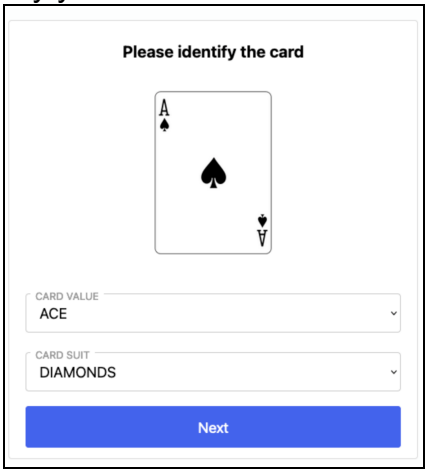


5 Continue with the next section.

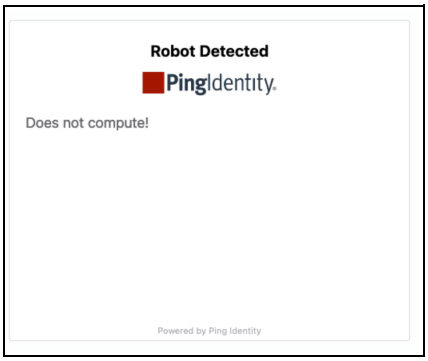
2.6.2 Test your flow with both paths

You want to test both paths therefore for one of the runs you will not enter the correct card values.

1 Save and deploy your flow to this point.
2 Try your flow and at the robot test select the wrong suit or value.

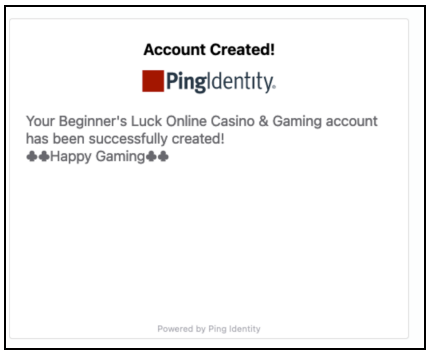


3 The result will be:



4 Try your flow again and at the robot test select the correct suit and value.

5 The result will be:

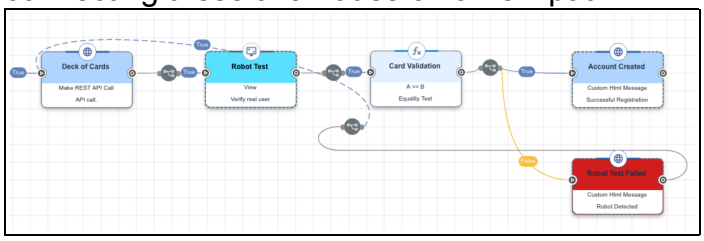


6 Continue with the next section.

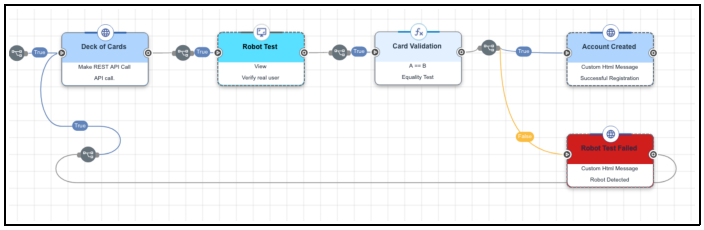
2.6.3 Add a retry on the robot test

The user may get it wrong the first time therefore lets let them try again. You will not put a limit on the retries but you could by setting some sort of counter and using a function.

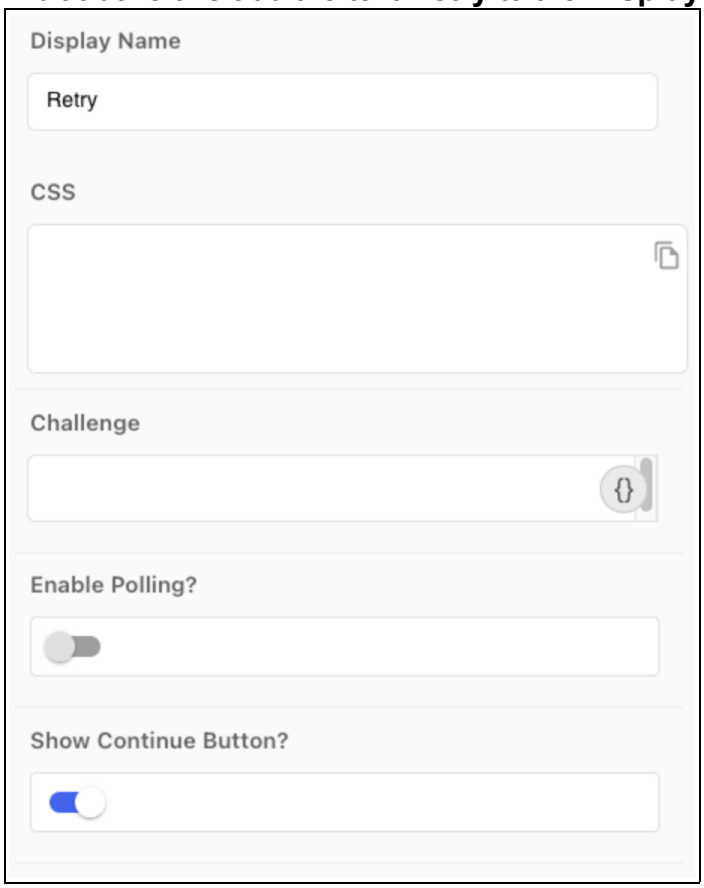
1 From the end of the **Robot Test Failed** node to the start of the **Deck of Cards** node draw a line connecting these two nodes on a new path.



2 You can drag around the nodes to better position the lines, for neatness but it will not effect how the flow runs.



3 Click the **Robot Test Failed** and scroll to the bottom to toggle **Show Continue Button** to be on.
4 A bit above this add the text **Retry** to the **Display Name** below Button.

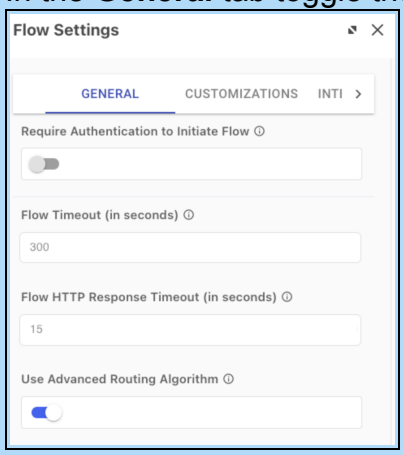


5 Once you have applied and close the dialog, click  icon after the **Try** button.
6 Click **Flow Settings** from the menu.

Note

The following is no longer required as this option is on by default now. Left here for reference in the event you run into in older documentation or comments in flow templates.

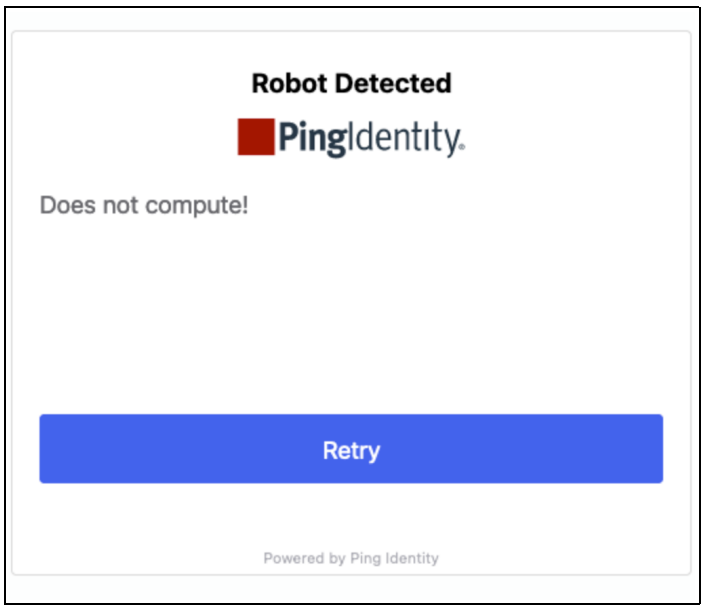
6.1 In the **General** tab toggle the **Use Advanced Routing Algorithm** toggle on.



- Since you are looping back within our flow, you will need to enable Advanced Routing from the settings menu.

6.2 Click **Save** button.

7 Save and deploy your changes then test it using the false path by entering incorrect suit or value.



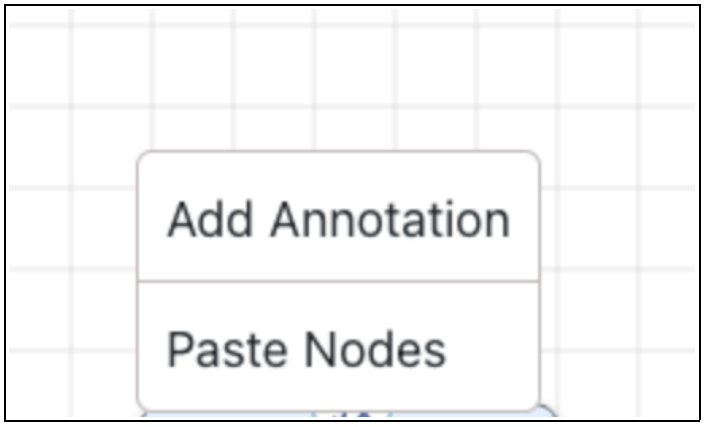
8 After you click **Retry** you will get a new card, this time select the proper suit and value.
9 Continue with the next section.

2.7 Documenting your flow

It is a good practice to document your flow so that others reviewing can understand what it is doing. You have been doing this by providing meaningful node title and description. You can also caption sections of your flow by using annotations or to add more detail explanations for the flow.

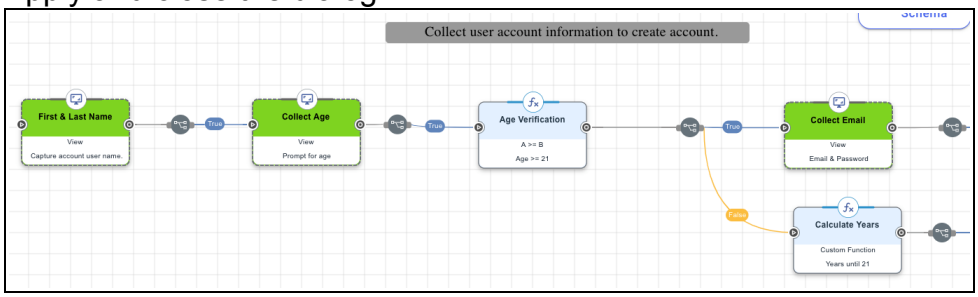
Even though the annotations are connectors, you simply add one by right-clicking on the flow canvas.

1 Scroll to the left on your canvas.
2 In the general area above the first few nodes that collect use input, **right-mouse click** in a area of the canvas:



3 Click **Add Annotation** to add it to the flow.
4 Click on the node you just added to edit it.
5 For the **Annotation Text** add the text **Collect user account information to create account.**
6 For the **Background Color** enter **999999** or select any other color that you like.

7 Apply and close the dialog.



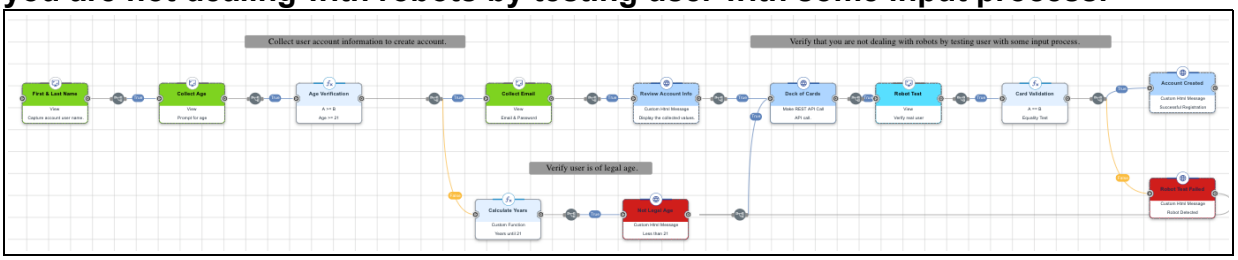
- You can drag the node around to position it where you want it.

8 **Right-mouse click** on the annotation you just added, and click **Clone** to copy it.

9 Drag the clone to be above the **Calculate Years** handling flow.

10 Change the text to **Verify user is of legal age.**

11 Clone the node again and drag over the robot test section and change the text to **Verify that you are not dealing with robots by testing user with some input process.**



12 You can add additional comments to your code if you want. Annotations do not impact the runtime when your flow is executed. They are there for documentation.

13 Save your changes and then close the flow.

14 You have completed this exercise.