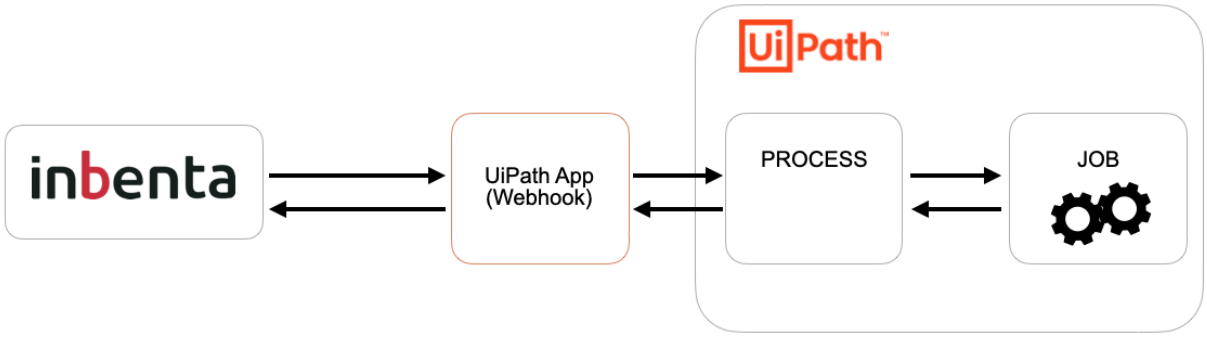
Introduction

Inbenta can be integrated with UiPath (Robotic Process Automation software) to automate repetitive front office tasks, complex business solutions (such as enterprise resource management), or manufacturing with robotic processes in action. UiPath streamlines processes, uncovers efficiencies and provides insights, making the path to digital transformation fast and cost-effective. It leverages existing systems to minimize disruption.

How to integrate with UiPath

Uipath integration with Inbenta is done through webhooks. Setup a webhook in the Inbenta chatbot instance. This webhook will call the respective Process (in Uipath Orchestrator) by passing the user inputs. Process starts the Job in UiPath studio or UiPath assistant. Once a Job is done the response from the Uipath is sent back to the chatbot.



Here is an example of how you could call a uiPath process from Inbenta.

Let's say you want to invoke an *Insurance Claim* process set up in UiPath.

As a Bot Master you can set up an action (with webhook) in Inbenta and have a form to collect all the information from the user to submit for an Insurance claim. Once the user inputs the values, they will get stored in variables and then Inbenta chatbot will pass the values to the webhook that in turn will invoke the relevant UiPath process. The UiPath process will get executed and the response (in this example a ClaimID) will be sent back to the chatbot.

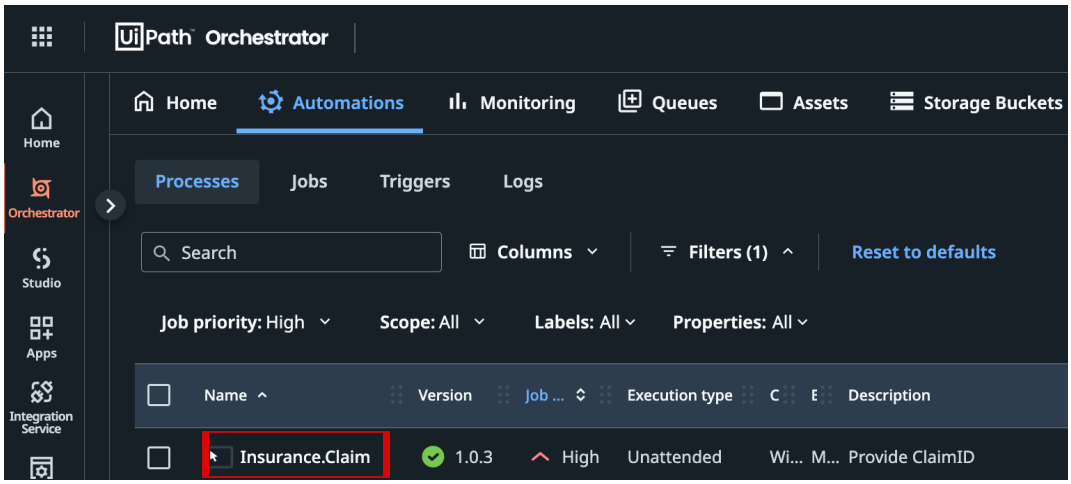
Detailed Walkthrough of the steps with an insurance claim example:

1. Create the UiPath Studio project with required ***Input*** and ***Output*** arguments

Name	Direction	Argument type	Default value
strOutputParam	Out	String	*Default value not supported*
strInsuranceType	In	String	"insurance_type"
strAccidentDescr	In	String	"Accident description"
strAccidentTime	In	String	"accident time"
strCarDamage	In	String	"car damage"
strOtherCars	In	String	"other cars"
strInjuries	In	String	"injuries"
strPolice	In	String	"police"
strCarType	In	String	"car type"
strInsuranceAccNr	In	String	"insurance account number"
processStatus	In	String	"pending"

Variables | Arguments | Imports

2. Upload the project to the UiPath Orchestrator. Verify that the process (***Insurance.Claim***) was loaded correctly.



3. Create a WEBHOOK file (*You can find the example code in the folder **example**)*

   3.1. Make a note of the Orchestrator API authorization parameters (**api_client_id**, **api_user_key**, **tenant_name**, **org_name**, **org_unit_id**) and your Process identification (**process_name**, **process_version, url_token**). These parameters will be used for authorization

```
$api_client_id = 'api_client_id';
$api_user_key = 'api_user_key';
$tenant_name = 'tenant_name';
$org_name = 'org_name';
$org_unit_id = 'org_unit_id';
$process_name = '\'process_name\'';
$process_version = 'process_version';
$url_token = 'https://account.uipath.com/oauth/token';
```

3.2. Define webhook ***Input*** parameters in i**nsurance_example_webhook.*php*** file. Make sure that variables have identical names to the ones described in your UiPath project ().

```
$input_parameters = [
   "strInsuranceAccNr" => $_POST['UIPATH_INSURANCE_ACC_NR'],
   "strInsuranceType" => $_POST['UIPATH_INSURANCE_TYPE'],
   "strName" => $_POST['FIRST_NAME'],
   "strEmail" => $_POST['EMAIL_ADDRESS'],
   "strAccidentDescr" => $_POST['UIPATH_ACCIDENT_DESCRIPTION'],
   "strAccidentTime" => $_POST['UIPATH_ACCIDENT_TIME'],
   "strCarType" => $_POST['UIPATH_CAR_TYPE'],
   "strCarDamage" => $_POST['UIPATH_CAR_DAMAGE'],
   "strOtherCars" => $_POST['UIPATH_OTHER_CARS'],
   "strInjuries" => $_POST['UIPATH_INJURIES'],
   "strPolice" => $_POST['UIPATH_POLICE'],
   "strDate" => date('Y-m-d'),
   "strTransactionID" => rand(),
   "processStatus" => 'Pending'
];
```

3.3. In this example we are taking the "***strOutputParam***" output argument as insurance ***claim ID*** (**insurance_example_functions.php**) and sending it back to the chatbot.

```
       $job_status_output_obj =
json_decode($job_status['value'][0]['OutputArguments']);
       $ClaimID = $job_status_output_obj -> strOutputParam;
       $message_reply="Thanks, your Claim ID is -> $ClaimID";
```

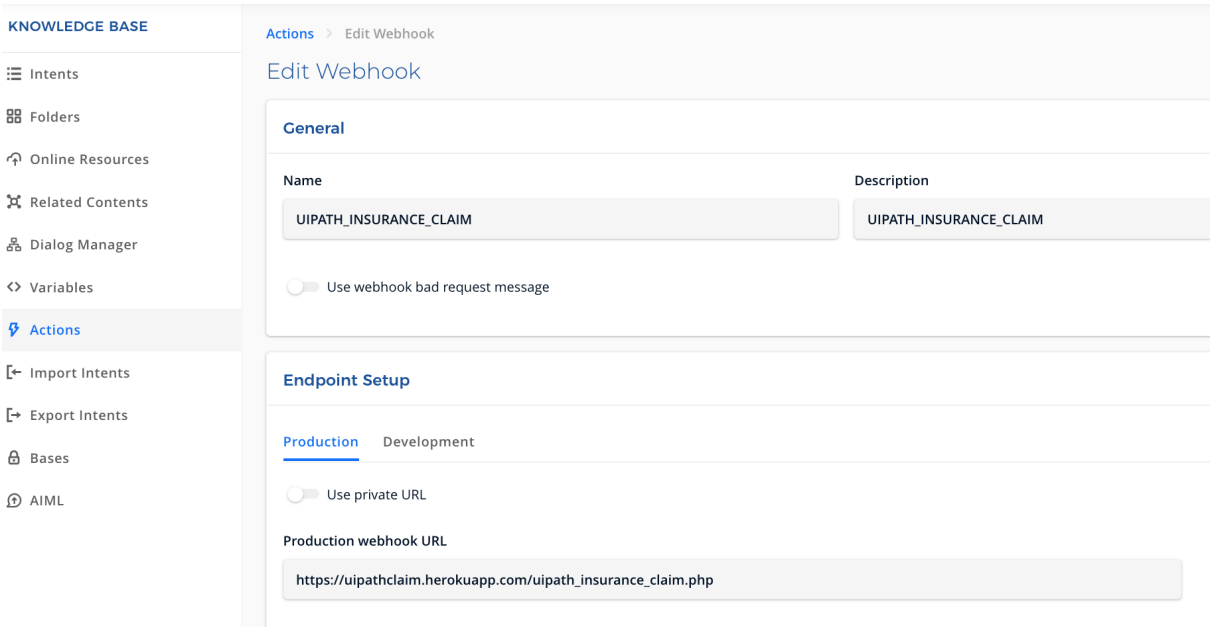
3.4. Copy the webhook URL and keep it handy. You will need it in the next step

4. Log in to the Inbenta workspace, navigate to the variables tab and create required variables. User inputs will be collected in these variables and passed to the webhook

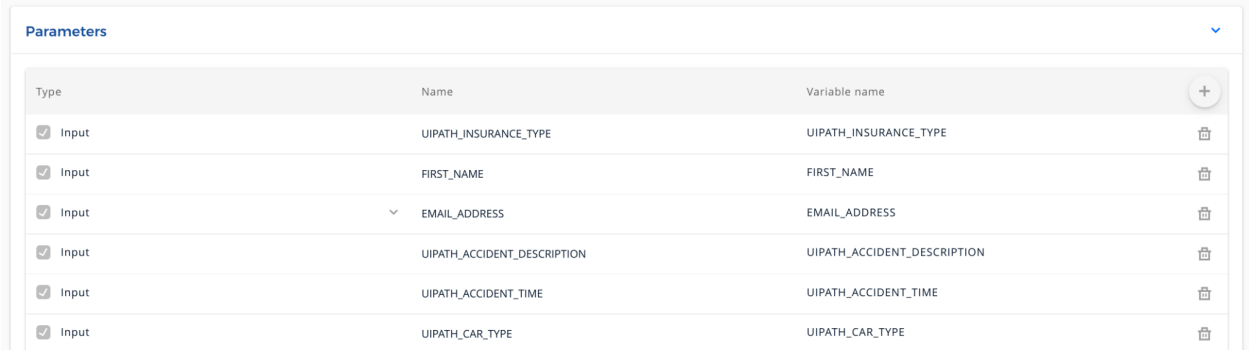
   Here is a detailed guide on how to create a variable: https://help.inbenta.com/en/chatbot/manage-knowledge-base/working-with-variables/creating-and-editing-variables/

5. Navigate to the actions tab and create a new webhook. Here is a detailed step by step guide that explains how to create a webhook: https://help.inbenta.com//en/creating-a-new-webhook/

   5.1. Enter the copied webhook URL from the previous step in the field **Production Webhook URL**

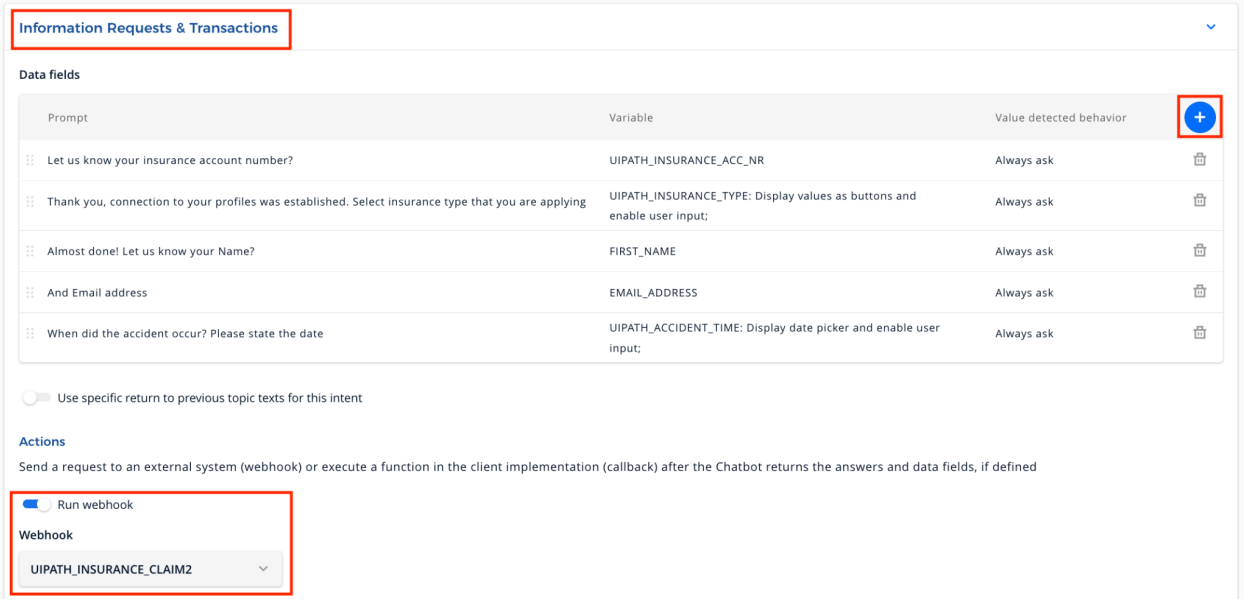


   5.2. Add all the variables you created as input parameters (Example: ‘FIRST_NAME’, ‘UIPATH_CAR_TYPE’ etc’) and click on **Create**



6. Navigate to the intents tab and create new Standard intent. How to create an intent -> https://help.inbenta.com/en/chatbot/manage-knowledge-base/managing-intents/creating-and-editing-intents-in-chatbot/

   6.1. Give your Intent a **Title** and an **Answer**
   6.2. Navigate to the “Information Requests & Transactions” tab. Add action and variables created in ***Steps 3 and 4*** to your new intent and save the changes



7. Test the connection. Verify that Uipath can receive parameters from Inbenta Chatbot

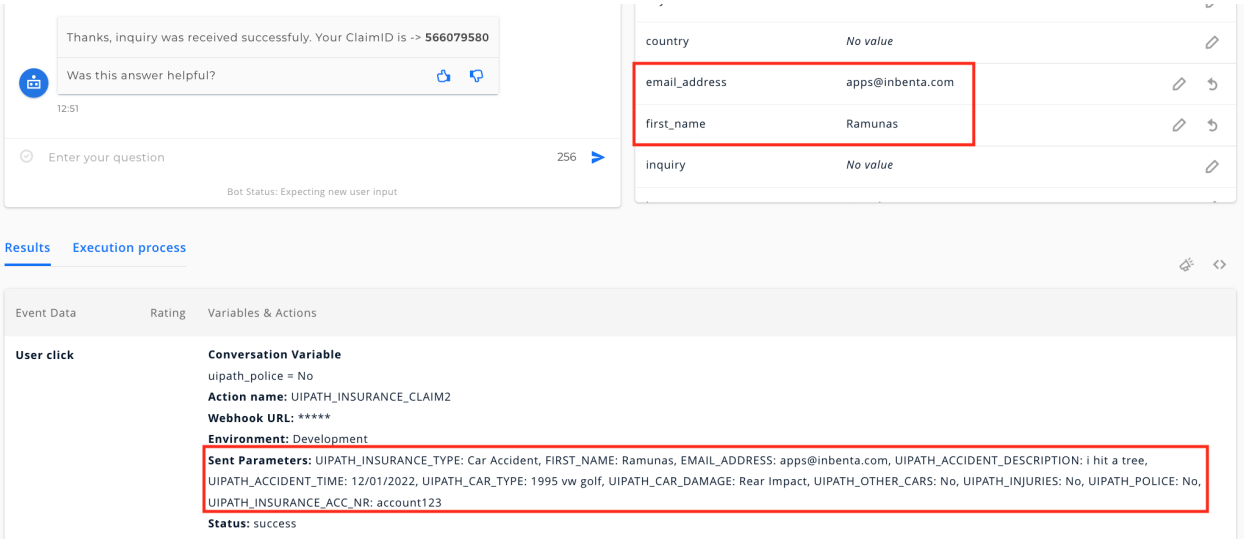
   7.1. Navigate to the **Test & Linguistic tuning** tab and select **Chatbot Debug**
   7.2. Ask your question (in this example, you could talk about filing an insurance claim).
   7.3. Bot will ask you to enter the values for all the parameters that we want to collect from the user.

7.4. As the user enters you can see the variable values getting populated on the right side of the debug window.

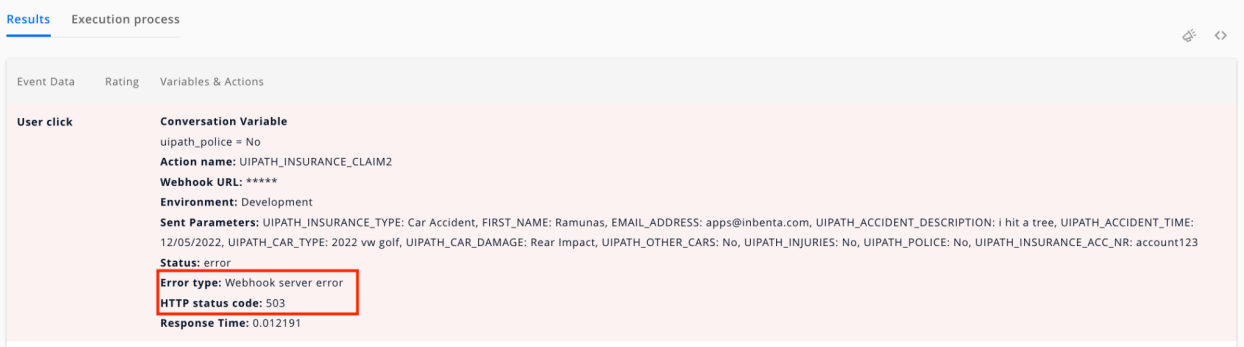
7.5. Once all the related user inputs are collected, the webhook will be called. You can check if the values passed to the webhook are correct.

7.6. If the webhook is executed successfully, it would have invoked the UiPath process.

7.7. The user will get the response from the bot with the message we have set up in the webhook JSON Response.

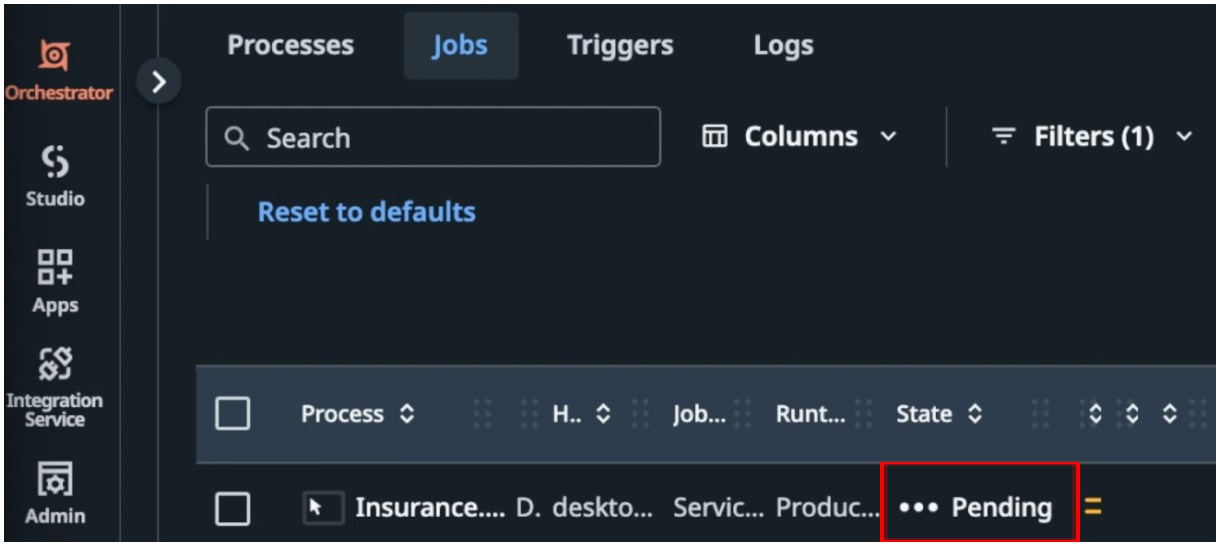


Note: If there were any issues and the webhook was not successfully executed, you would be able to see the error and additional details in the debug window.



8. Login to the UiPath Orchestrator, navigate to the Jobs tab and verify that Job was successfully executed

    8.1. It takes a few seconds for the Orchestrator to connect to the UiPath. Uipath to execute the task and provide the results back to the Orchestrator. During that time the Job status is "***Pending***".



    8.2. If the job status is "***Success***", it means the job got executed successfully on the UiPath side.