



# REST integration – using ODM with IBM RPA

## Hands-on Lab

*Version 1.2 for  
public use*

### *Authors:*

*Paul Pacholski – [pacholsk@ca.ibm.com](mailto:pacholsk@ca.ibm.com)*

*Zach Silverstein – [zachary.silverstein@ibm.com](mailto:zachary.silverstein@ibm.com)*

*Gabriel Sanchez – [gabriel.sanchez@wdgautomation.com](mailto:gabriel.sanchez@wdgautomation.com)*

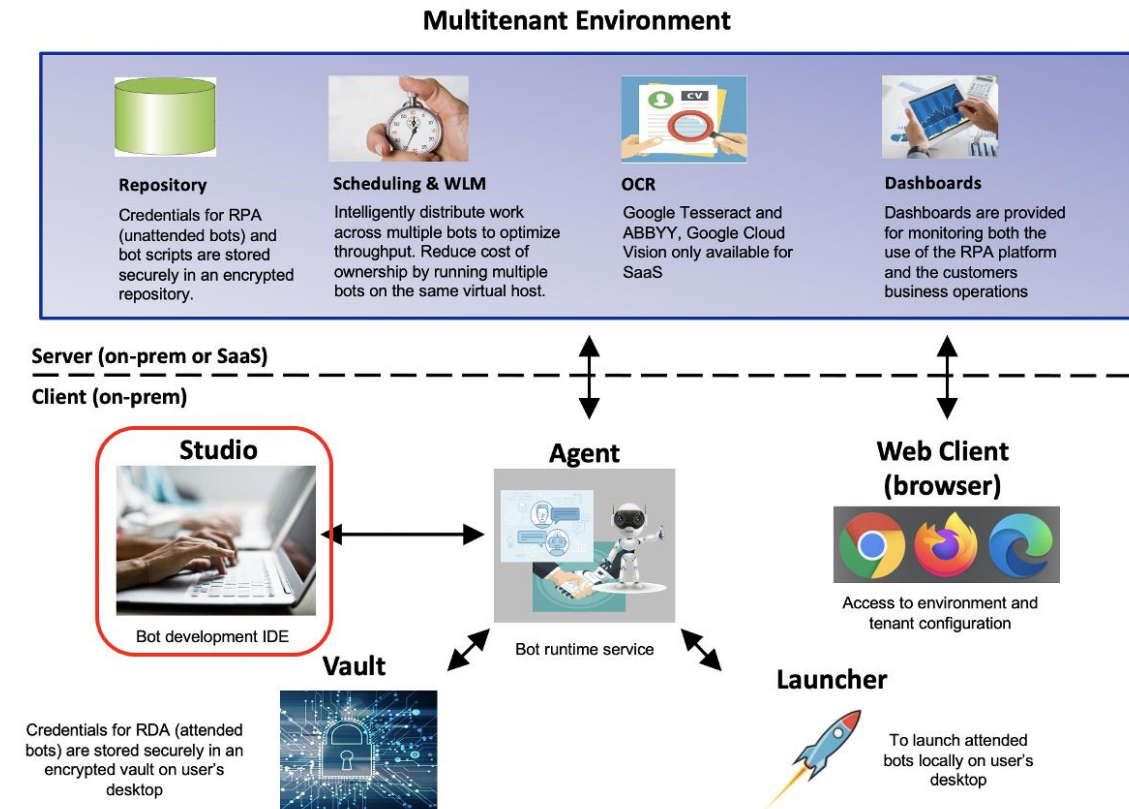
*Jukka Juselius – [jukka.juselius@fi.ibm.com](mailto:jukka.juselius@fi.ibm.com) (modifications for BP virtual workshop)*

## Table of Contents

|  |    |
|--|----|
| 1. Introduction .....                    | 3  |
| Use case .....                           | 4  |
| Prerequisites .....                      | 4  |
| 2. Lab Instructions.....                 | 5  |
| Open completed lab.....                  | 5  |
| Examine the Decision Service .....       | 5  |
| Call the Decision Service .....          | 10 |
| Test your automation .....               | 16 |
| 3. Work with a Completed Automation..... | 18 |

# 1. Introduction

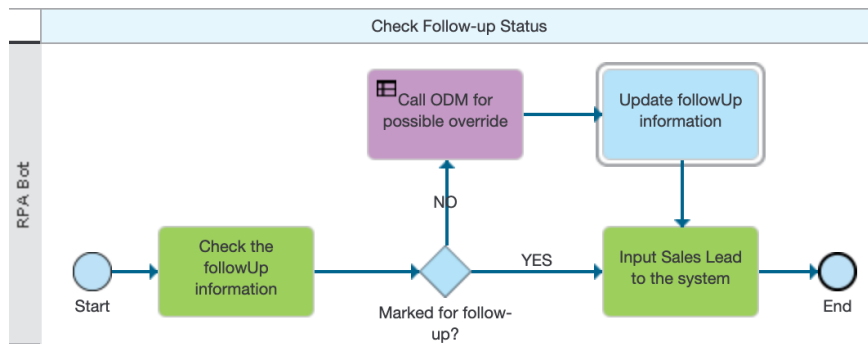
This hands-on lab you will use IBM RPA Studio to make outbound calls via assorted functionality. You will utilize the sales leads file from previous labs and query assorted services.



## Use case

In Lab 1 you learned how to automate processing of sales leads that arrive in a CSV/Excel format. The bot you created in Lab 1 automatically entered the sales leads into the online opportunity system of record (JK Automation Sales Leads).

You will now modify your automation to make an outbound call to IBM ODM rule engine using its REST API to know whether or not to override the follow up requested flag when it's set to "No". The flow diagram below shows the logic that we're going to add to our automation script.



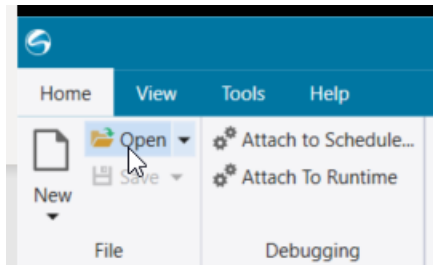
## Prerequisites

IBM ODM up and running.

## 2. Lab Instructions

### Open completed lab

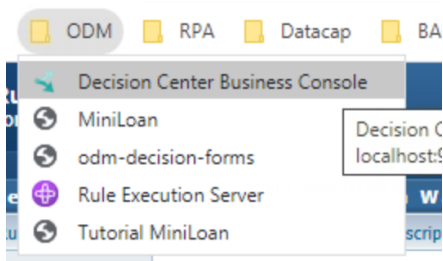
\_1. In IBM RPA Studio, Select Open .



\_2. Navigate to **sales-lead-automation-REST-START.wal** in *Desktop* → *IBM RPA Lab Resources* → *Lab 3* – *REST API Usage* and click Open .

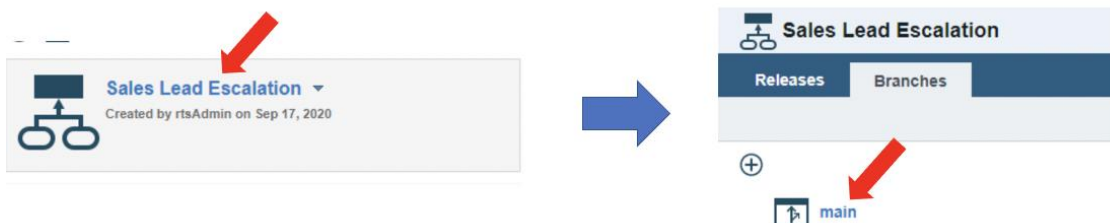
### Examine the Decision Service

\_1. Open up ODM Decision Center. Open **Chrome** (🌐 icon in the task bar) web browser and from the bookmarks bar select **ODM** → **Decision Center Business Console**.

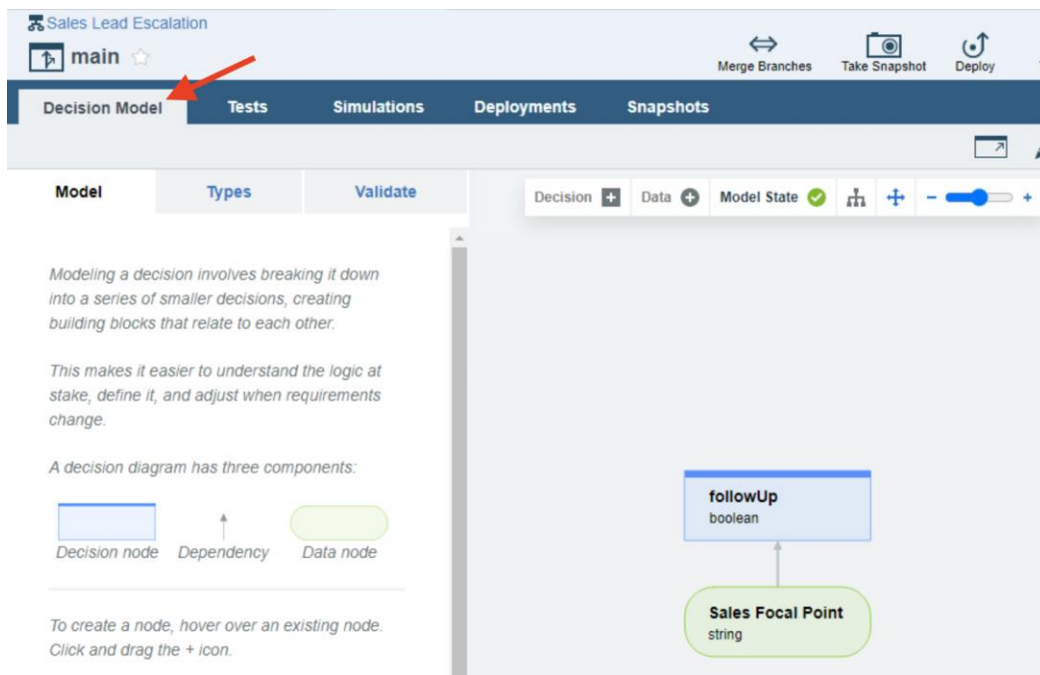


\_2. Login as **rtsAdmin** with password **rtsAdmin**.

\_3. Move to **LIBRARY** tab (if not already) to see all the Decision Services managed in the ODM environment. Click the name **Sales Lead Escalation** and then **main** branch.

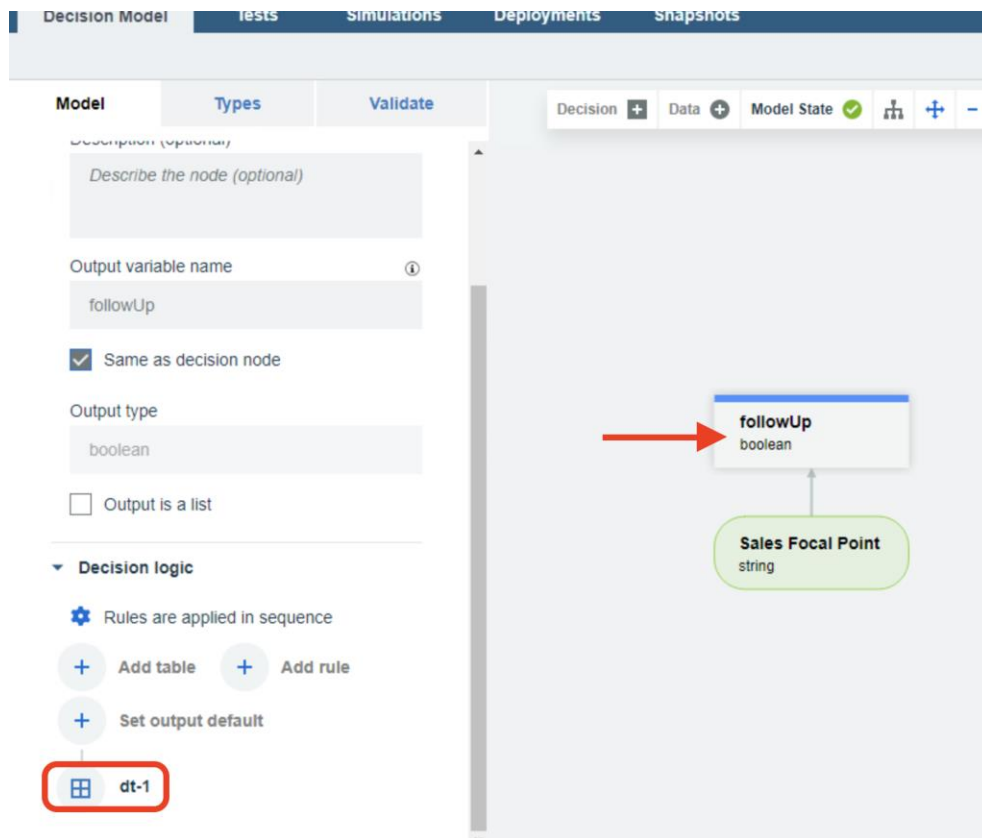


\_4. Move to **Decision Model** tab (if not opened by default) to view the decision service we're going to use with our automation.



This simple decision service is modeled using the integrated DMN (Decision Model and Notation) based modeler. It takes one input (as modelled with green input data node), Sales Focal Point as a string, and has one decision node for the decision logic that provides a single Boolean value as the out from the service.

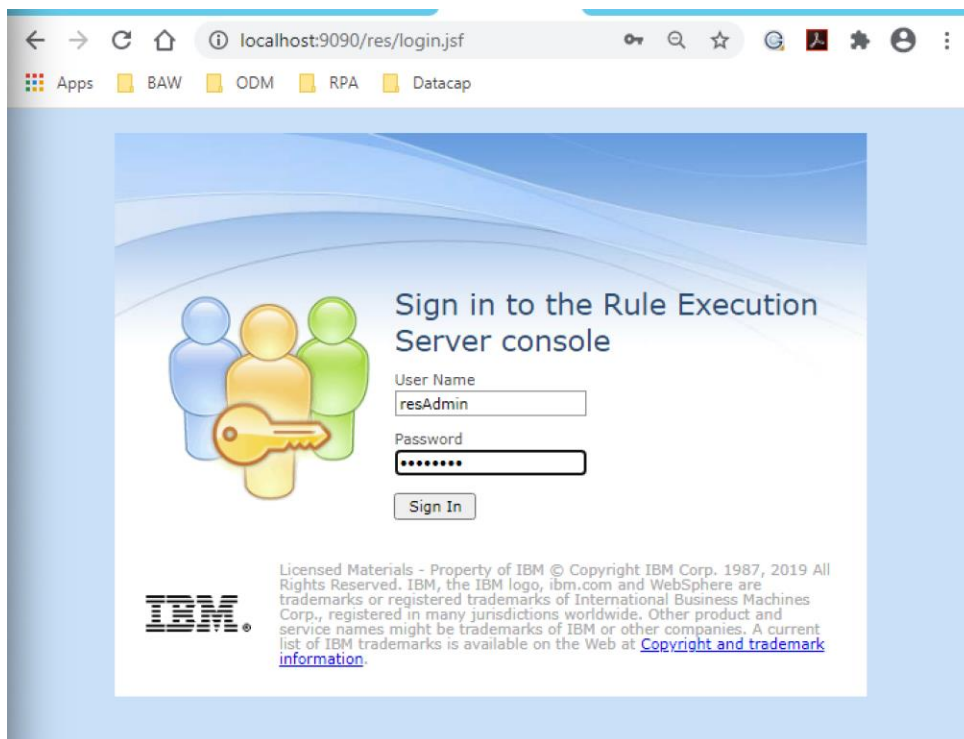
\_5. **Click the blue decision node** to select it and then scroll down the left-hand side model panel to see the Decision logic components inside the node. It just has one decision table named **dt-1**. **Click** it to see how it has been implemented.



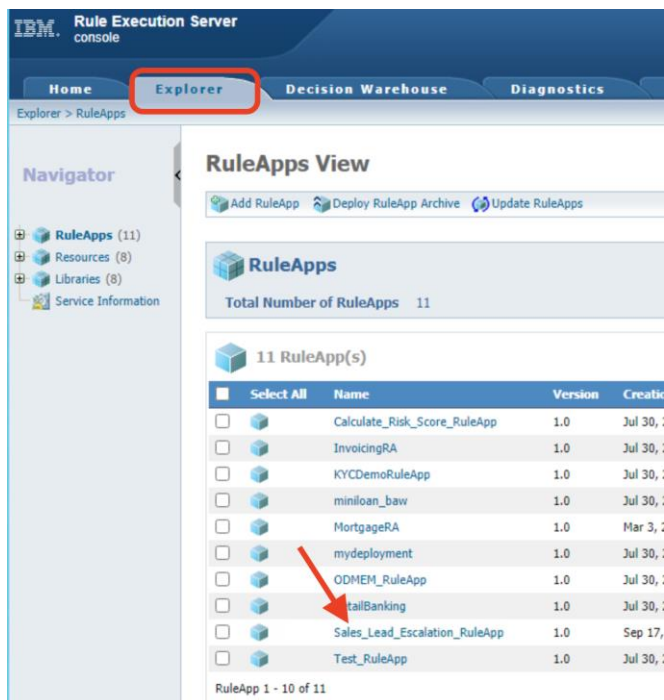
\_6. Decision table is just to set (override) the followUp information within the sales leads that our automation bot reads from the CSV file. If the recorded interest (Sales Focal Point) is for example for *Cloud* (the 1<sup>st</sup> row of the table), we set the followUp Boolean to false (not selected) and for *Robotic Process Automation*, we set it to true. Nice and simple.

\_7. Next, log into your **Rule Execution Server Console** with username **resAdmin** and password **resAdmin**. You can open the login page from bookmarks bar: *ODM* → *Rule Execution Server*.

Since the decision service is already installed to rule execution server (RES), we can test and retrieve its REST endpoint via the RES console.



\_8. Select **Explorer** at the top of the screen and find our *Sales\_Lead\_Escalation\_RuleApp* and **Click** on it.



\_9. On next view, **click** the latest *Sales\_Lead\_Escalation* Ruleset.

\_10. **Click Retrieve HTDS Description File** in the top right



## Ruleset View

| Direction | Name              | Kind   | XOM Type          |
|-----------|-------------------|--------|-------------------|
|           | Sales_Focal_Point | native | java.lang.String  |
|           | followUp          | native | java.lang.Boolean |

\_11. Select **REST** as the Service protocol type, **OpenAPI – JSON** as the Format, check **Latest ruleset version** and **click view**. This brings up the Swagger document for the decision service REST API.

Retrieve HTDS Description File

/Sales\_Lead\_Escalation\_RuleApp/1.0/Sales\_Lead\_Escalation

Service protocol type

☐ SOAP ☒ REST

Format: OpenAPI - JSON

☒ Latest ruleset version

☐ Latest RuleApp version

☐ Decision trace information

☐ Proxy for API Connect

Cancel View Download Test

\_12. For us, the most relevant information is highlighted below. We will use this information later on when we're conducting the actual REST call to ODM. **You do not need to copy any information from here, it's just for your reference and to understand how to get it from IBM ODM.**

```
{
  "swagger" : "2.0",
  "info" : {
    "description" : "API to invoke the execution of the decision service",
    "version" : "1.0.0",
    "title" : "Sales_Lead_Escalation API"
  },
  "host" : "localhost:9090",
  "basePath" : "/DecisionService/rest",
  "schemes" : [ "http" ],
  "consumes" : [ "application/json" ],
  "produces" : [ "application/json" ],
  "paths" : {
    "/Sales_Lead_Escalation_RuleApp/Sales_Lead_Escalation" : {
```

\_13. Go Back to Ruleset View (should be still open in one of your browser tabs) **click** on *Retrieve HTDS Description File* again.

\_14. This time please select TEST instead of VIEW (keep the other options same). This will show us a sample payload. If you want, you can test the service by inserting “Cloud” as Sales\_Focal\_Point value and **clicking Execute Request button** at the bottom of the view.

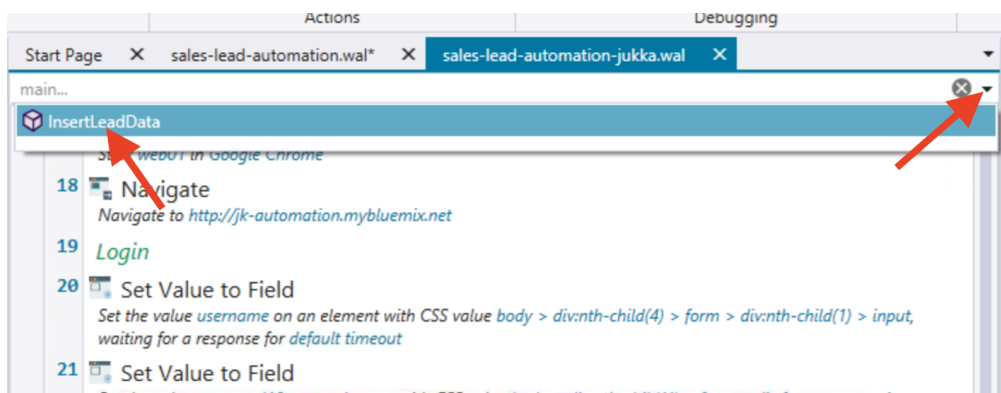
## Decision Service : /Sales\_Lead\_Escalation



Good. Now we know how to call the service using its REST API. You can close your browser now.

## Call the Decision Service

- \_1. Go back to your Studio and select Designer mode at the bottom of the screen, if not selected.
- \_2. Select the dropdown bar at the top of our script and select our subroutine *InsertLeadData* to open it.



\_3. We want to modify how we handle the follow flag, if the REST call to our decision service comes back as true. Navigate to the **IF** command (line 51) in our *InsertLeadData* Subroutine.

\_4. Drag a **HTTP Request** command from the toolbox just before the **IF** command. The configuration window will open.

\_5. Set the values to what you see listed below:

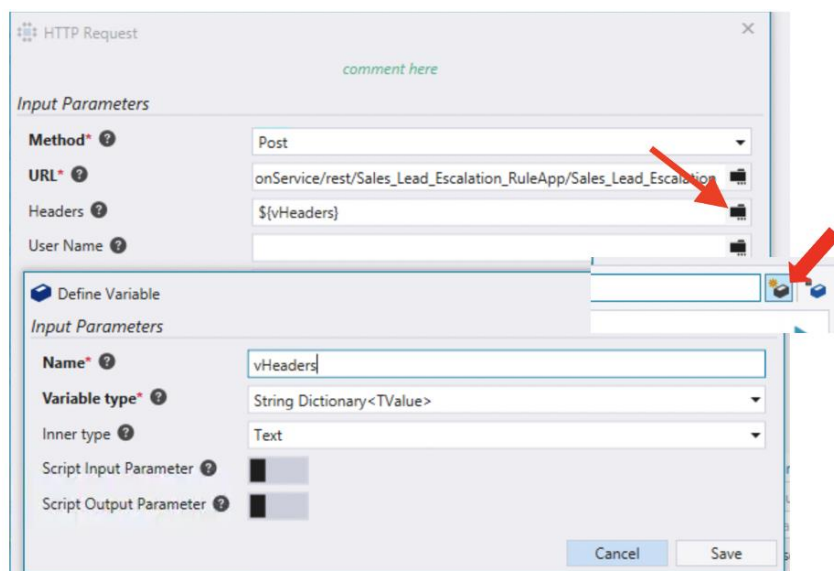
- Method: Post
- URL: [http://localhost:9090/DecisionService/rest/Sales\\_Lead\\_Escalation\\_RuleApp/Sales\\_Lead\\_Escalation](http://localhost:9090/DecisionService/rest/Sales_Lead_Escalation_RuleApp/Sales_Lead_Escalation)
- Headers: create a new variable named **vHeaders** (look below)
- Body: { "Sales\_Focal\_Point": "\${interest}" }
- Response: vRES

\*\*\*\*\* **IMPORTANT** \*\*\*\*\*

**Note:** *Sales\_Focal\_Point* and the *\${interest}* variable in the Body field are enclosed in double quotes! If you copy the text from here to your Studio, **you need to re-enter the double quotes manually in Studio**. Copying to VM changes the characters so that Studio does not register them as double quotes.

\*\*\*\*\* **IMPORTANT** \*\*\*\*\*

You will need to **Click the folder icon next to headers** to manually create a new String Dictionary of Type Text. First click the folder then **the brick with the star** next to it to open the Define Variable window.



Your configuration should look like this:

Click **Save**.

Your InsertLeadData subroutine should have now a HTTP Request command just before the IF command.

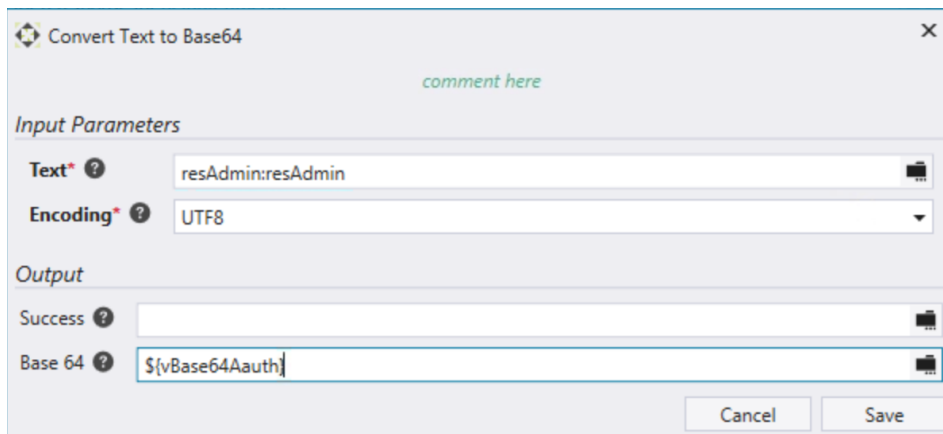
\_6. Now Navigate back to your main script by closing InsertLeadData inside the box using the X-icon

\_7. Create a new variable called **vBase64Auth** of type Text.

\_8. For the First Line of the Code (Line 20) add a **Convert Text to Base64** Command from the toolbox with your actual ODM RES Server credentials in a format of *username:password*. For our ODM RES Server the Text need to be **resAdmin:resAdmin**.

In real life we would store and get these from IBM RPA secure repository, but for the demo we take the easy way.

Set the *Base 64 output* to our newly created variable of **vBase64Auth**.

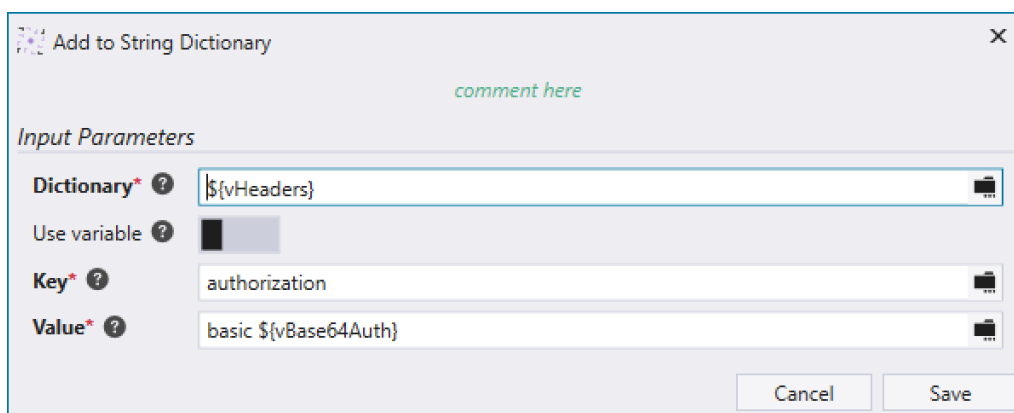


The screenshot shows the 'Convert Text to Base64' dialog box. It has a title bar with a close button. Below the title bar is a green text label 'comment here'. The 'Input Parameters' section contains two fields: 'Text\*' with the value 'resAdmin:resAdmin' and 'Encoding\*' with a dropdown menu set to 'UTF8'. The 'Output' section contains two fields: 'Success\*' and 'Base 64\*'. The 'Base 64\*' field is selected and contains the value '\${vBase64Auth}'. At the bottom right are 'Cancel' and 'Save' buttons.

\_9. Add an *Add to String Dictionary* Command on Line 21 just after the Convert Text to Base64 command and configure it as follows:

- Dictionary = \${vHeaders}
- Key = authorization
- Value = basic \${vBase64Auth}

Click **Save**.



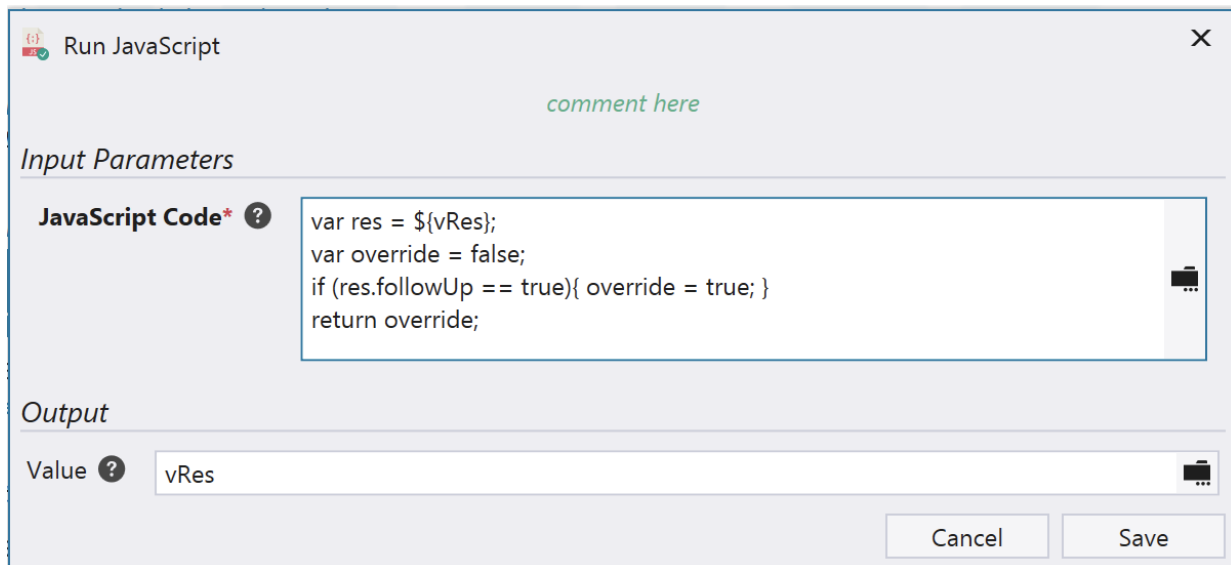
The screenshot shows the 'Add to String Dictionary' dialog box. It has a title bar with a close button. Below the title bar is a green text label 'comment here'. The 'Input Parameters' section contains four fields: 'Dictionary\*' with the value '\${vHeaders}', 'Use variable\*' with a checked checkbox, 'Key\*' with the value 'authorization', and 'Value\*' with the value 'basic \${vBase64Auth}'. At the bottom right are 'Cancel' and 'Save' buttons.



- \_10. Navigate back to our *InsertLeadData* subroutine using our dropdown selector
- \_11. Now, we can run some JavaScript to extract out the response, so right after our HTTP Request, add a **Run JavaScript** command.
- \_12. Insert JavaScript to parse out the response, ensuring no special chars (e.g. new line return) inserted along:

```
var res = ${vRes};  
var override = false;  
if (res.followUp == true) { override = true; }  
return override;
```

You should have the code above inserted as below:



**Run JavaScript**

*comment here*

**Input Parameters**

**JavaScript Code\*** ?

```
var res = ${vRes};
var override = false;
if (res.followUp == true){ override = true; }
return override;
```

**Output**

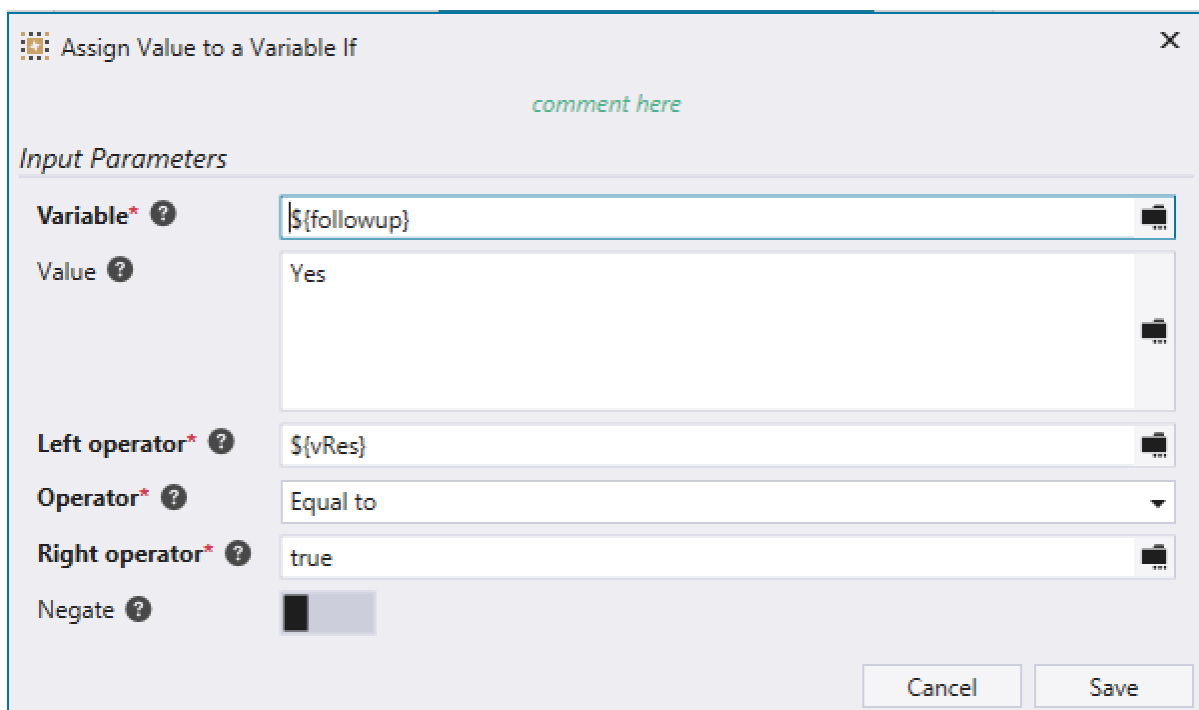
**Value** ? vRes

Cancel Save

Make sure to set also the output value to **vRes**!

Click **Save**.

\_13. Pull an **Assign Value to a Variable If** command from the Toolbox and place it below the Run JavaScript command. Configure as shown in the picture:



**Assign Value to a Variable If**

*comment here*

**Input Parameters**

**Variable\*** ? \${followup}

**Value** ? Yes

**Left operator\*** ? \${vRes}

**Operator\*** ? Equal to

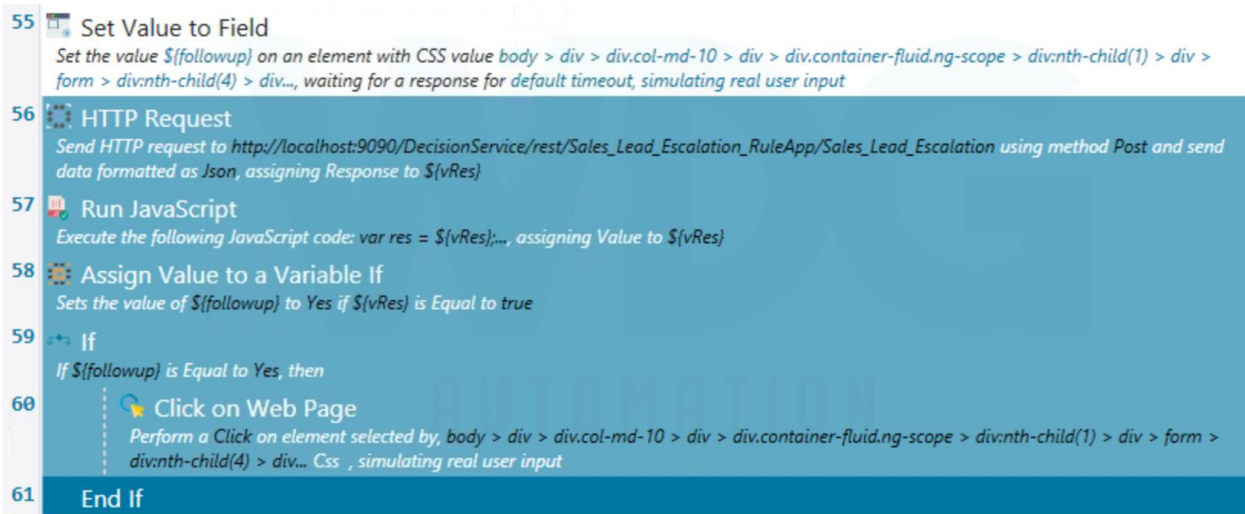
**Right operator\*** ? true

**Negate** ? ☐

Cancel Save

Click **Save** to close the configuration windows. Also **save your work** by hitting Ctrl+s or using the Save option from the toolbar.

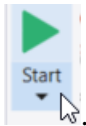
You should have similar sequence in your subroutine.



Nice work. You should be ready for testing!

## Test your automation

\_1. Click the green Start icon form the top toolbar. Alternatively, you can hit F5.





\_2. You should see the bot filling in the form automatically.

JK Automation x +

Not secure | jk-automation.mybluemix.net/#/leads

Demos - Filter... username

Download file

Leads

Claims

Orders

Parts

## Sales Leads

### Add New Lead

**First Name:** Dave

**Last Name:** Wakeman

**Email:** dwakeman@us.ibm.com

**Job Title:** Director IT

**Company:** Wakeman Aviation Enterprises

**Phone:** 515-555-4981

**Address:** 9321 Fleur Drive

**City:**

**State:**

**Zipcode:**

**Area of Interest**

**Follow up** ☐

**Submit**

\_3. Note the follow-up flag such as for **Pramond Prince** and **Sheri Williams** that we expect not to be flagged. Now they should be flagged since the overriding decision service.

|    | First Name | Last Name | Job Title               | Company                      | email                       | phone        | Client Address        | Client City | Client State | Client Zipcode | Area of Interest            | Followup Requested |
|----|------------|-----------|-------------------------|------------------------------|-----------------------------|--------------|-----------------------|-------------|--------------|----------------|-----------------------------|--------------------|
| 1  | Paul       | Pacholski | Consultant              | Hannibal Enterprises         | pacholsk@hannibalent.com    | 905-883-8234 | 119 Barkley Drive     | Thornhill   | ON           | L3T-6N2        | Auto ML                     | Yes                |
| 2  | Dave       | Wakeman   | Director IT             | Wakeman Aviation Enterprises | dwakeman@us.ibm.com         | 515-555-4981 | 9321 Fleur Drive      | Des Moines  | IA           | 50319          | Robotics Process Automation | Yes                |
| 3  | Pramond    | Prince    | IT Developer            | United Third Bank            | pramond@unitedthirdbank.com | 612-555-2341 | 332 Hennipen Ave      | Minneapolis | MN           | 55404          | Microservices               | No                 |
| 4  | Otto       | Priest    | CEO                     | Autonomous Cars, Inc.        | otto@autocars.com           | 669-338-8221 | 9676 Thirteenth Loop  | San Jose    | CA           | 95002          | Autonomous Cars             | Yes                |
| 5  | Sam        | Fishburne | VP Sales                | Crashover Computers          | stuf@crashover.com          | 512-555-1488 | 11501 Burger Rd       | Bustin      | TX           | 78702          | Business Process Management | Yes                |
| 6  | Sheri      | Williams  | Director of Operations  | Piston Manufacturing         | sheri@pistonmfg.com         | 313-555-2991 | 503 Fasbinder Parkway | Detroit     | MI           | 48202          | IT Service Management       | No                 |
| 7  | Monika     | Datar     | Data Scientist          | Covetere Pharmaceuticals     | monika@coveterepharma.com   | 212-555-5591 | 51 Astor Place        | New York    | NY           | 10046          | Machine Learning            | Yes                |
| 8  | Jeff       | Goodhue   | Director of Programming | Dojo Networks                | jeff@dojo.com               | 424-555-4398 | 33912 Sunset Blvd     | Culver City | CA           | 90233          | Content                     | Yes                |
| 9  | Satoshi    | Nakamoto  | Lead Developer          | Programming Anonymous        | satoshi@coinbit.com         | 929-555-2358 | 3928 Crypto Lane      | Seattle     | WA           | 98114          | Case                        | No                 |
| 10 | Timothy    | McGee     | Data Scientist          | Coach K Communications       | tim@coachk.com              | 919-555-0720 | 928 hoopsville Lane   | Raleigh     | NC           | 27676          | Data Science                | Yes                |
| 11 | Josh       | Bockman   | Director of IT          | HealthFirst Patient Services | josh@healthfirst.com        | 617-555-1103 | 321 King St           | Boston      | MA           | 2132           | Business Rules              | Yes                |
| 12 | Stuart     | Jones     | Enterprise Architect    | Wessex Financial Services    | stuart@wessex.com           | 630-555-9227 | 71 S Wacker           | Chicago     | IL           | 60624          | Cloud                       | No                 |

**THIS COMPLETES THIS HANDS-ON LAB**

### 3. Work with a Completed Automation

If you just want to run the automation without authoring it or if you just want to look at a completed solution, you can open the **sales-lead-automation-REST-completed.wal** from the lab folder (*Desktop → IBM RPA Lab Resources → Lab 3 – REST API Usage*).

Refer to [Test your automation](#) section to start your bot.