Technical Report

Building Automation and Orchestration for Software-Define Storage with NetApp and VMare

Using NetApp OnCommand Workflow Automation, VMware vRealiez Automation, and vRealize Orchestration for Software Defined Storage

Mirko Van Colen, NetApp  
July 2019

Mirko Van Colen, NetApp

October 2019 | TR-4306 | Version 2.1 (Unofficial)

Contents

[Building Automation and Orchestration for Software-Define Storage with NetApp and VMare 1](https://netapp-my.sharepoint.com/personal/mirko_netapp_com/Documents/Documents/UCB/UCB%20MetroCluster%20Expansion%20v1.3.docx#_Toc21965727)

[Mirko Van Colen, NetApp July 2019 1](https://netapp-my.sharepoint.com/personal/mirko_netapp_com/Documents/Documents/UCB/UCB%20MetroCluster%20Expansion%20v1.3.docx#_Toc21965728)

[1 Introduction 3](#_Toc21965729)

[2 Versions 3](#_Toc21965730)

[3 Instructions 3](#_Toc21965731)

[3.1 Install the vRealize WFA package 3](#_Toc21965732)

[3.2 Add your WFA host(s) 4](#_Toc21965733)

[3.3 Install certificiate 8](#_Toc21965734)

[3.4 Determine the userinputs of your workflow 9](#_Toc21965735)

[3.5 Create a vRealize workflow 12](#_Toc21965736)

[3.6 Test your new workflow 20](#_Toc21965737)

[3.7 Fancy you workflow 22](#_Toc21965738)

# Introduction

I’m going to skip the whole “How to install WFA”, I assume you know how to do this. If not, search for the original TR-4306 (this is just an unofficial updated version)

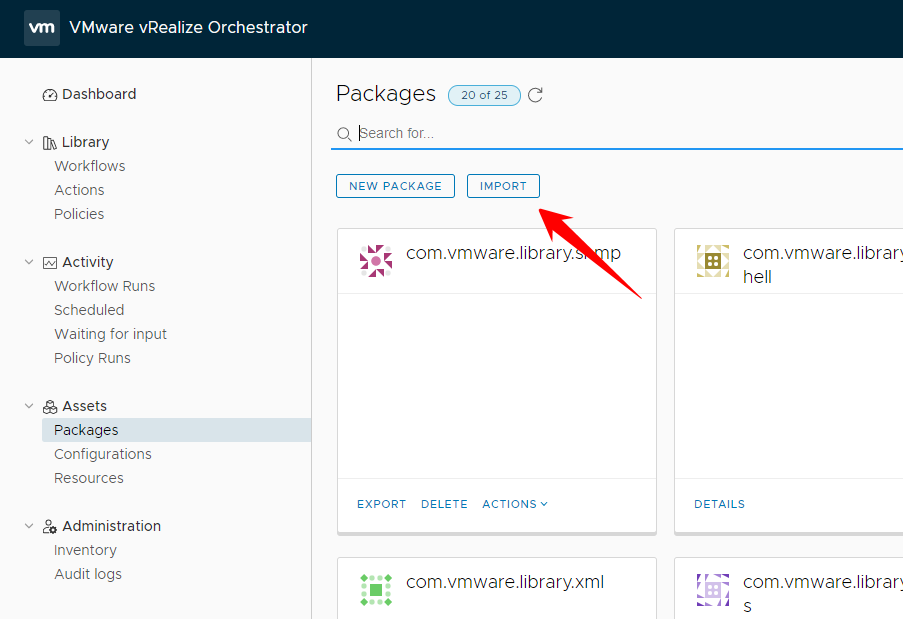
# Versions

This updated version is using vRealize v7.6 and WFA v4.2 (although I’m certain that 5.0 will work just as fine)

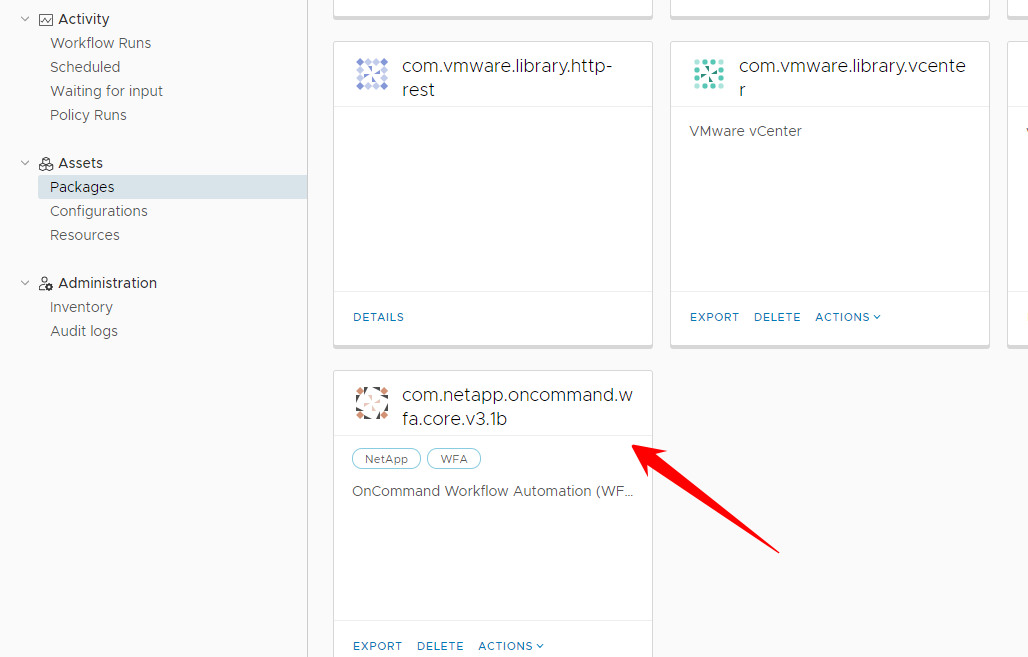
# Instructions

## Install the vRealize WFA package

Import the package

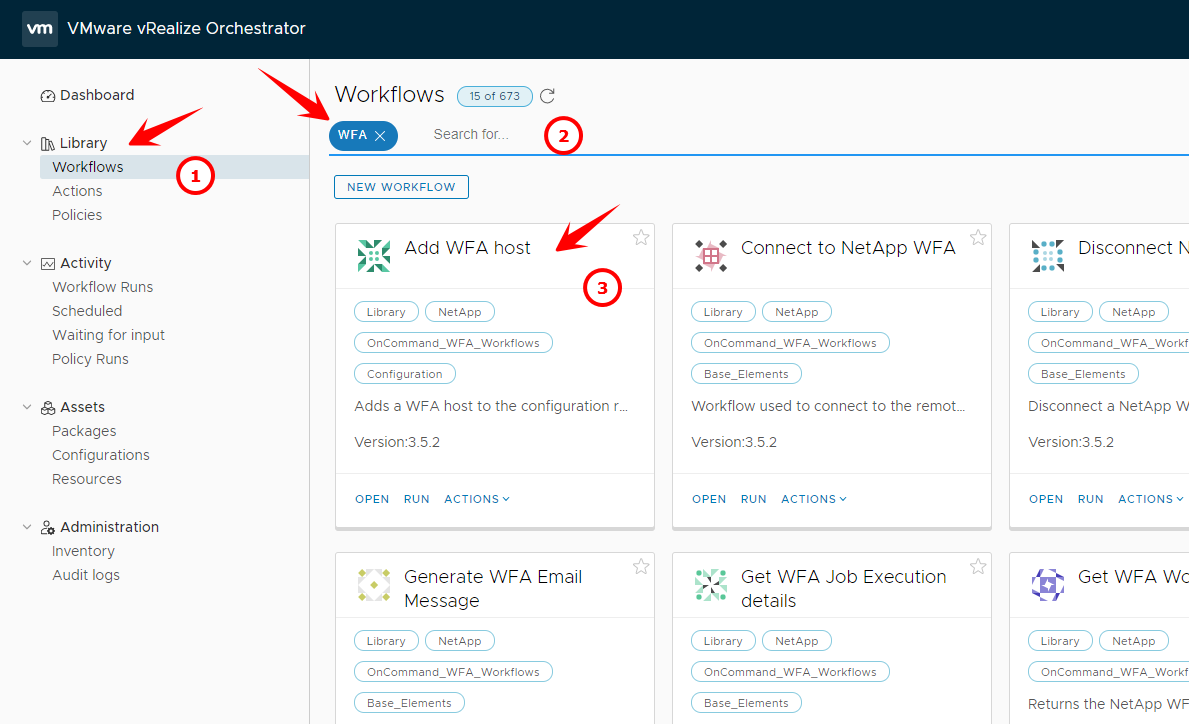


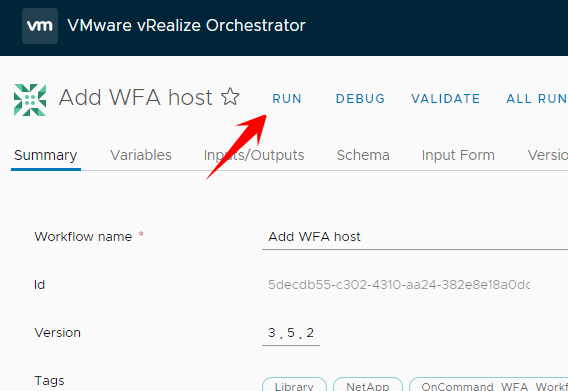
The package appears in the package list

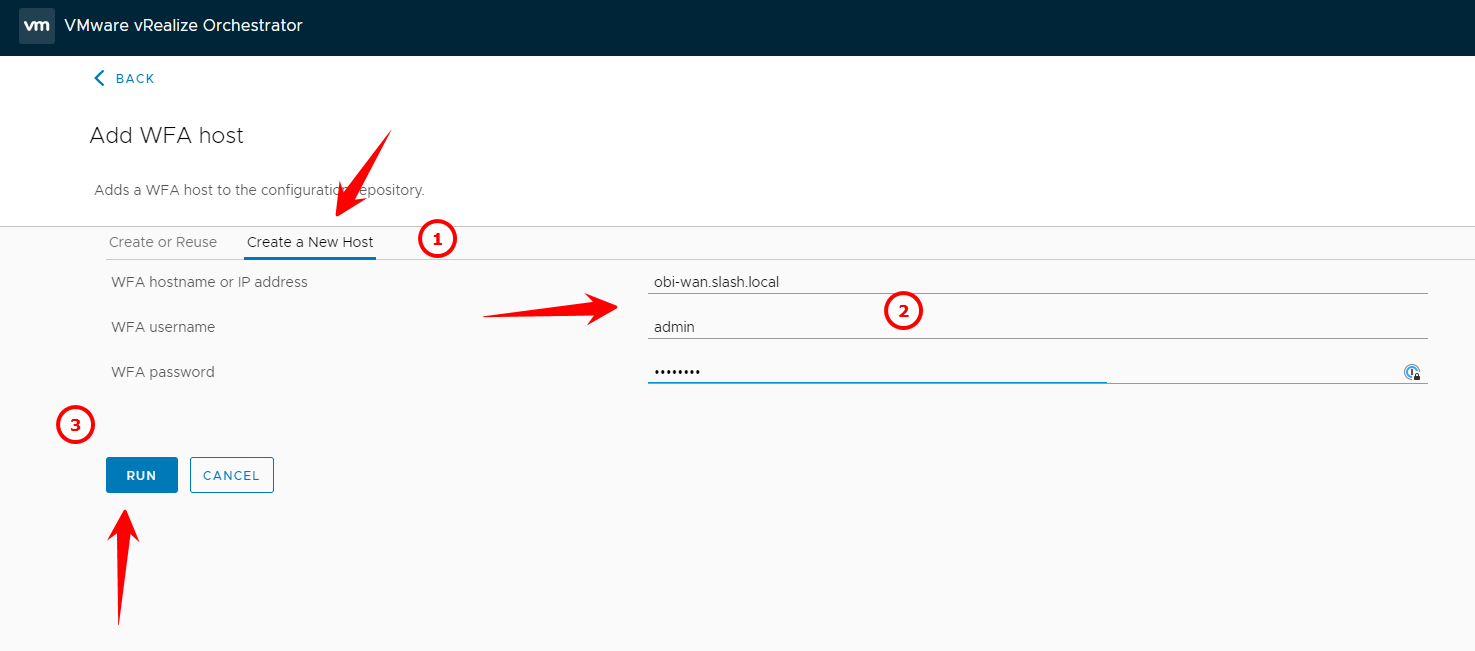


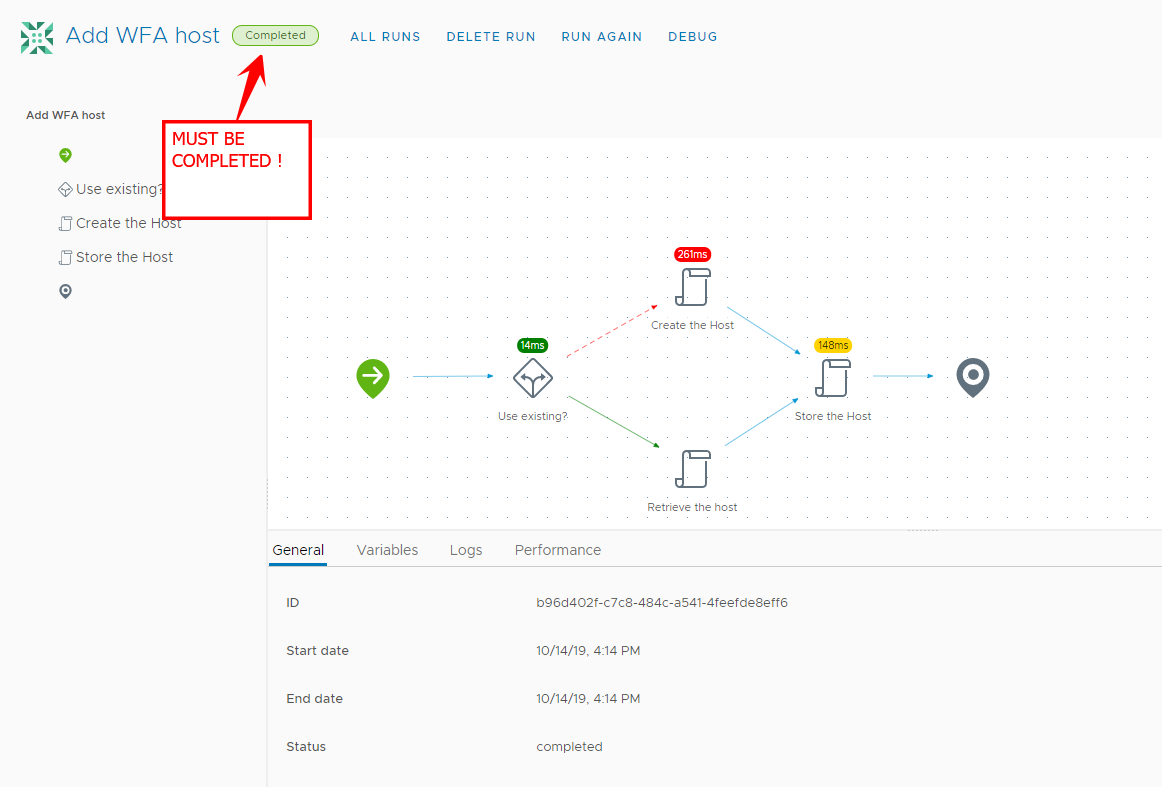
## Add your WFA host(s)

Go to workflows ; filter for WFA ; open “Add WFA host”



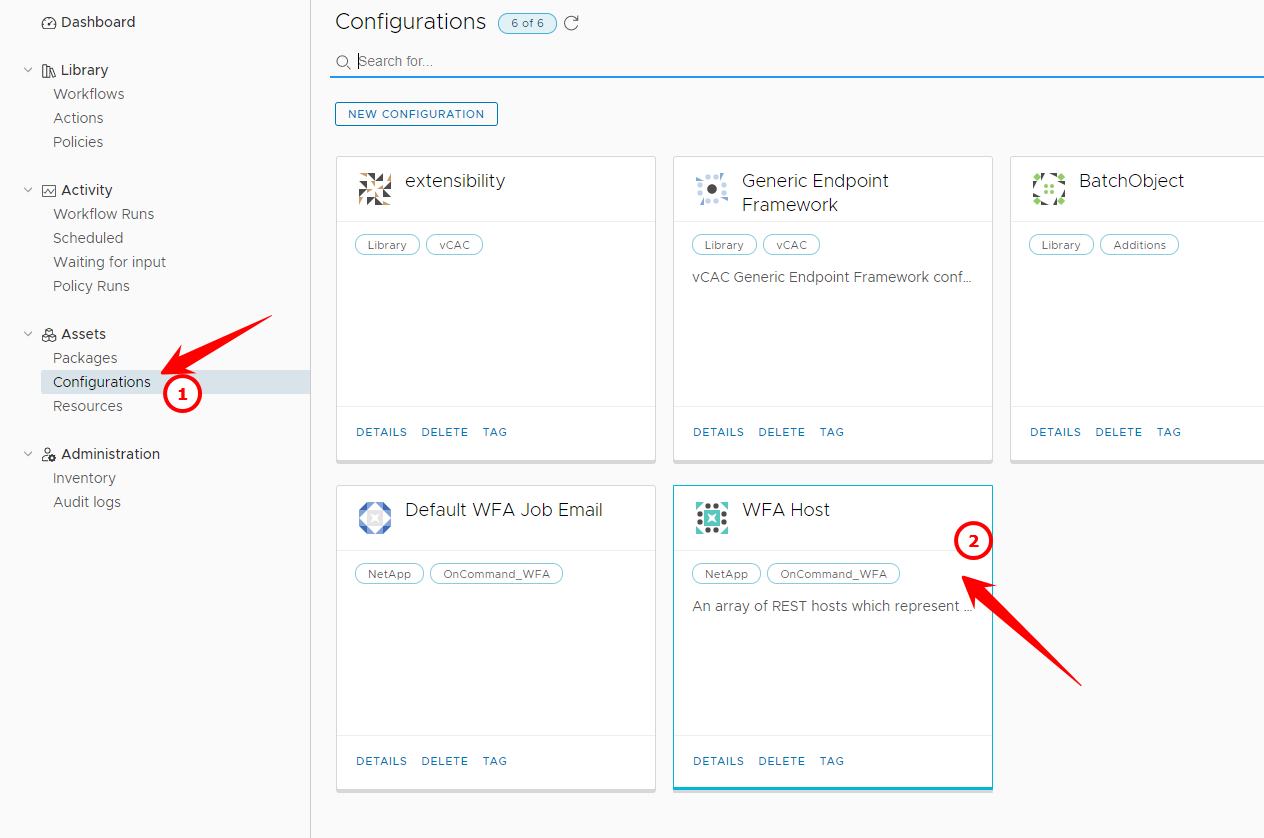




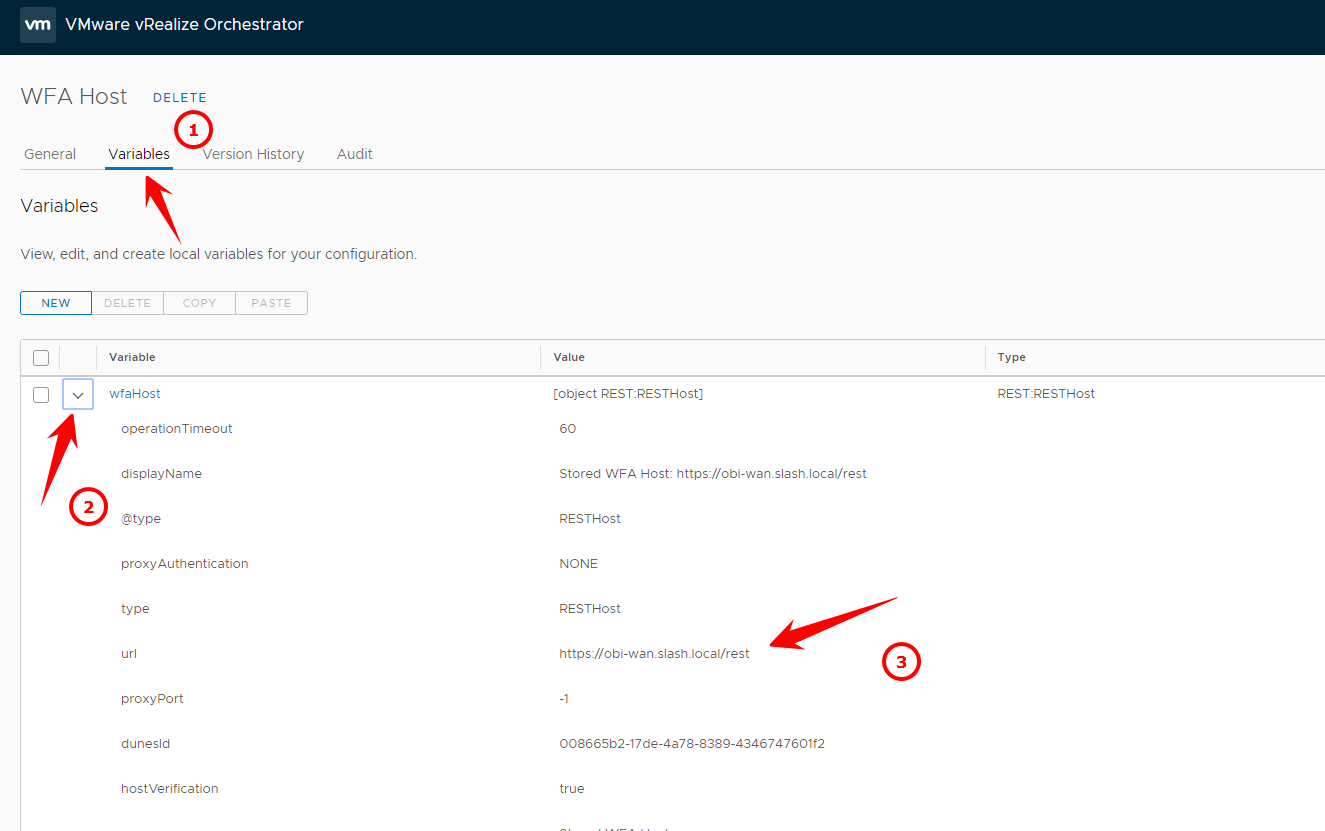


|  |
| --- |
| Note |
| You can run this multiple times and add more wfa servers |

Go to configuration and open WFA Host



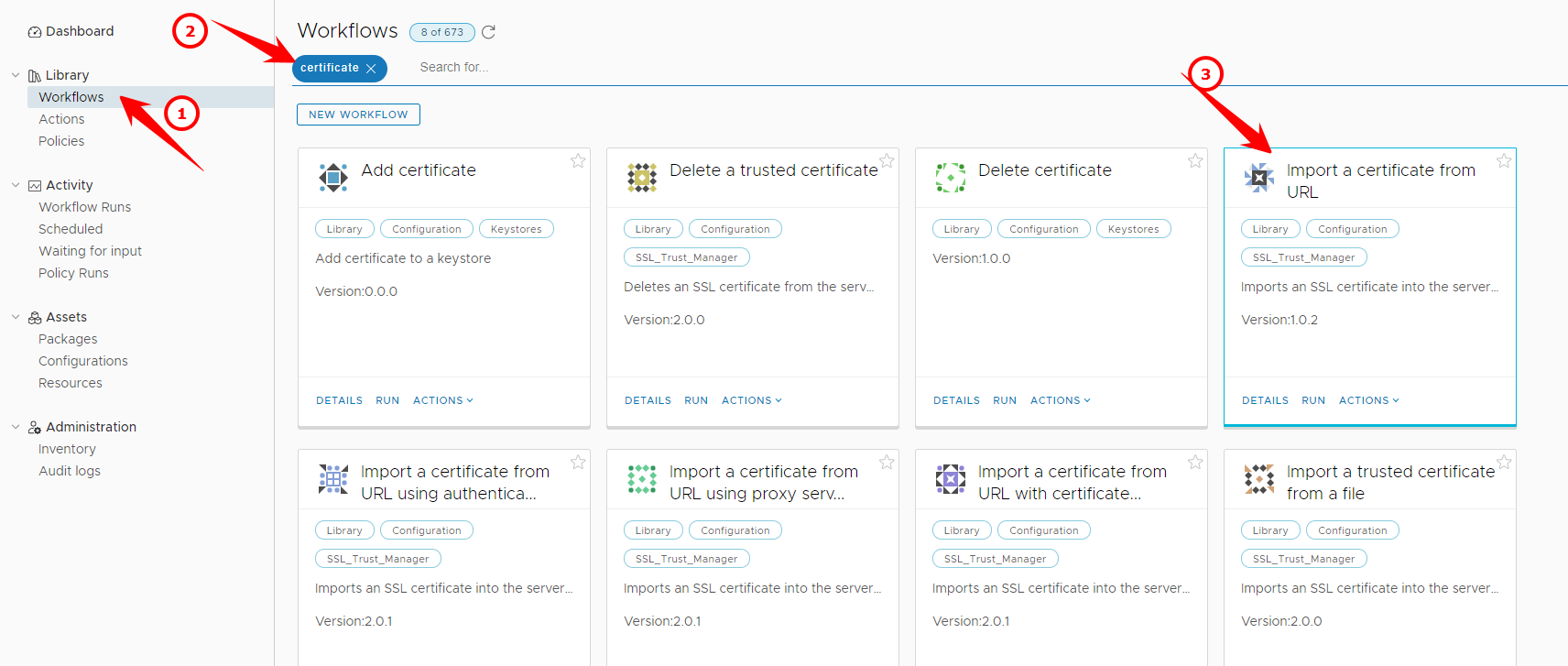
Double check that the WFA host is added (it will only show the last you added)



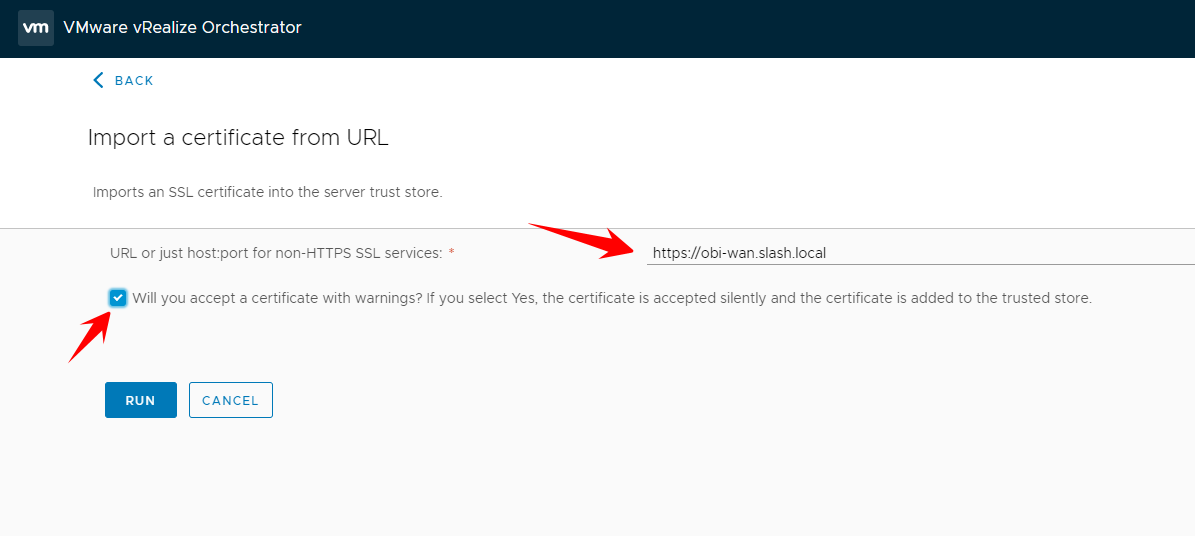
## Install certificiate

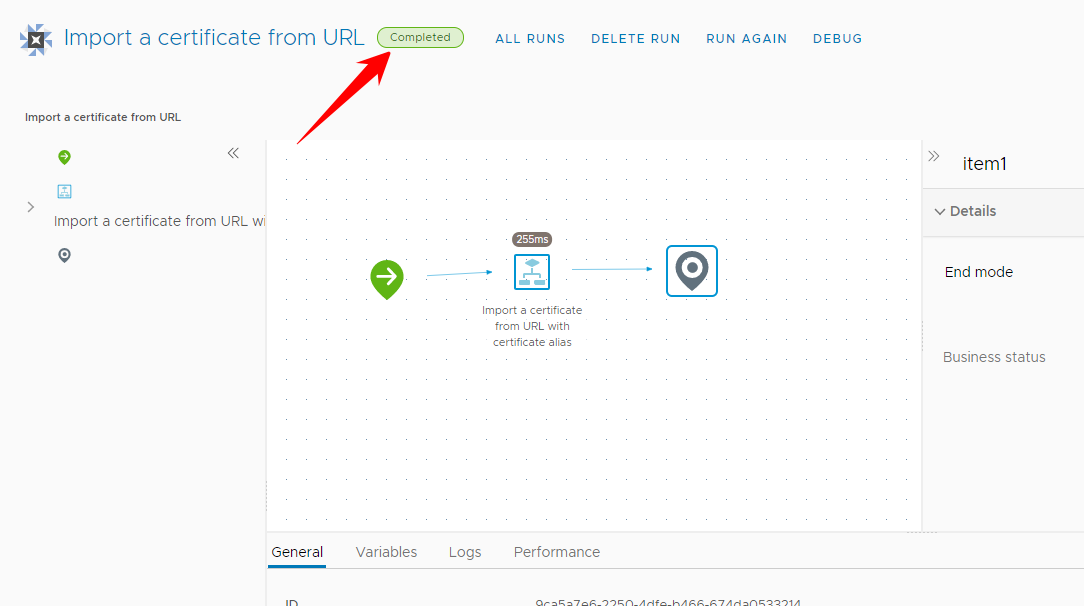
The WFA server will most likely have a self signed certificate, so we must add it to the trusted store

Run the “import a certificate from URL” workflow



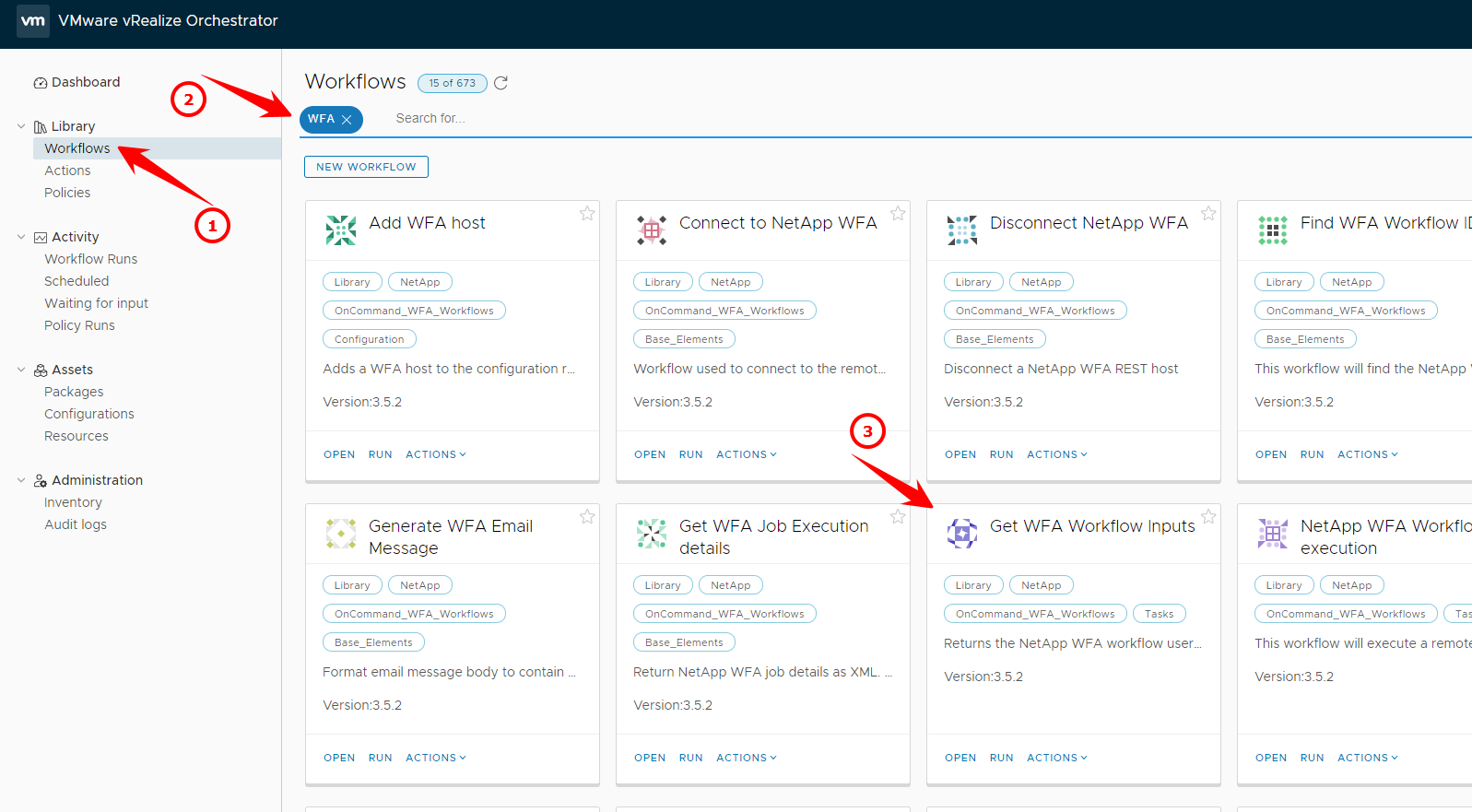
Run the workflow



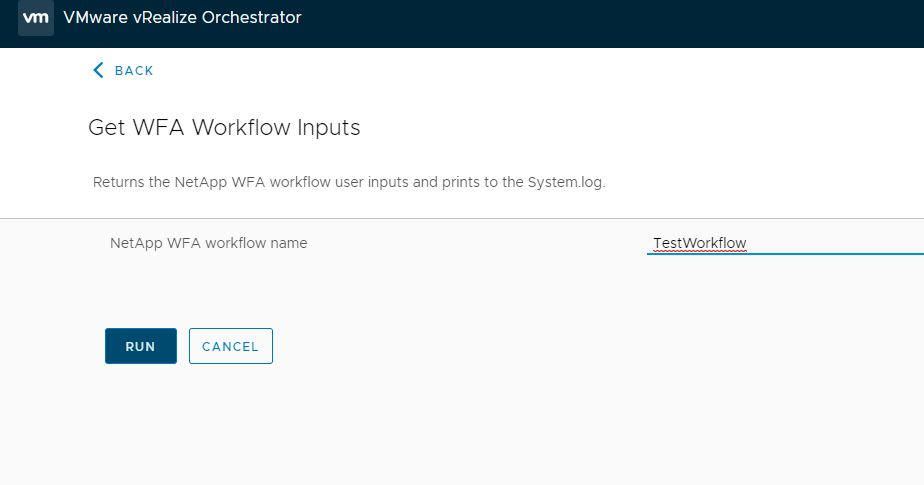


## Determine the userinputs of your workflow

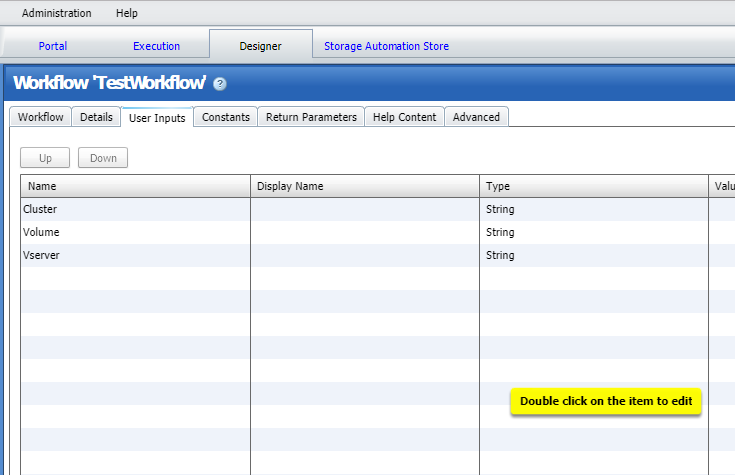
Infact, you probably know the userinputs if you created the workflow. These are simply the rest userinputs. Still there is are vRealize workflow for it, you can run.



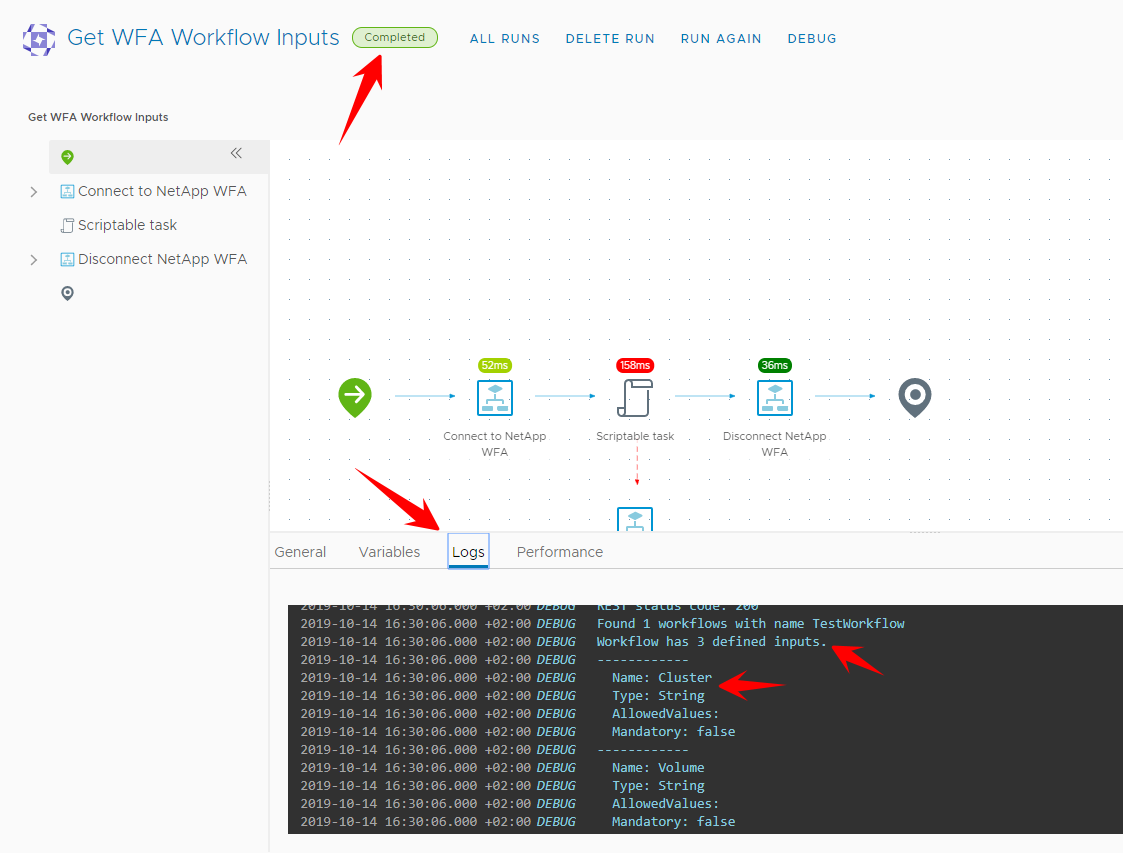
In my case, I have created a simple test workflow called “TestWorkflow”



Below the test workflow that searches for a volume (using 3 userinputs)

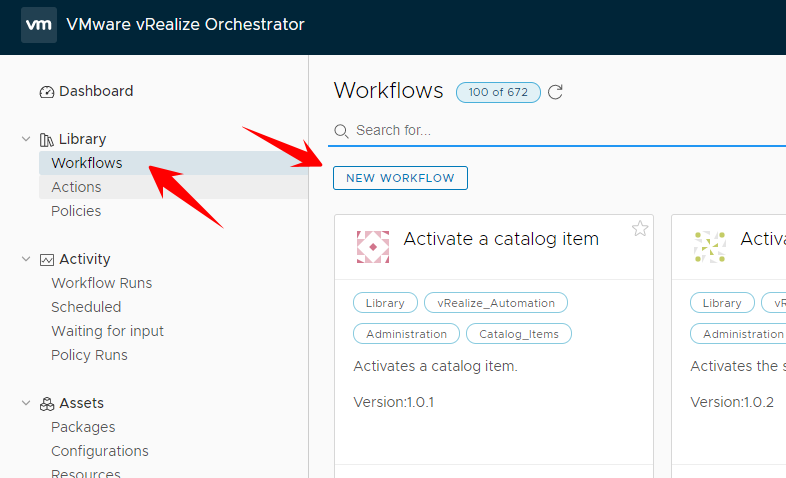


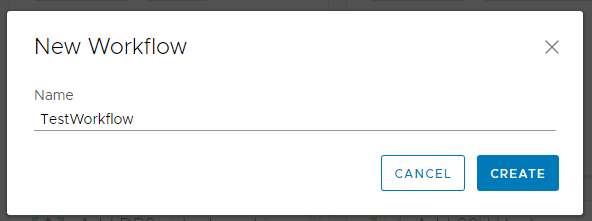
In the logs, you can see that it has found those 3 userinputs



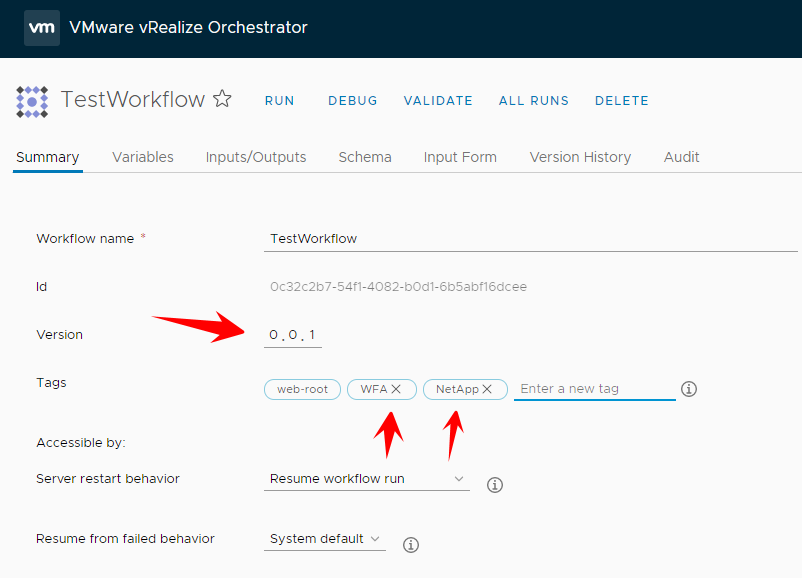
## Create a vRealize workflow

We encourage you to use the same name as the WFA workflow

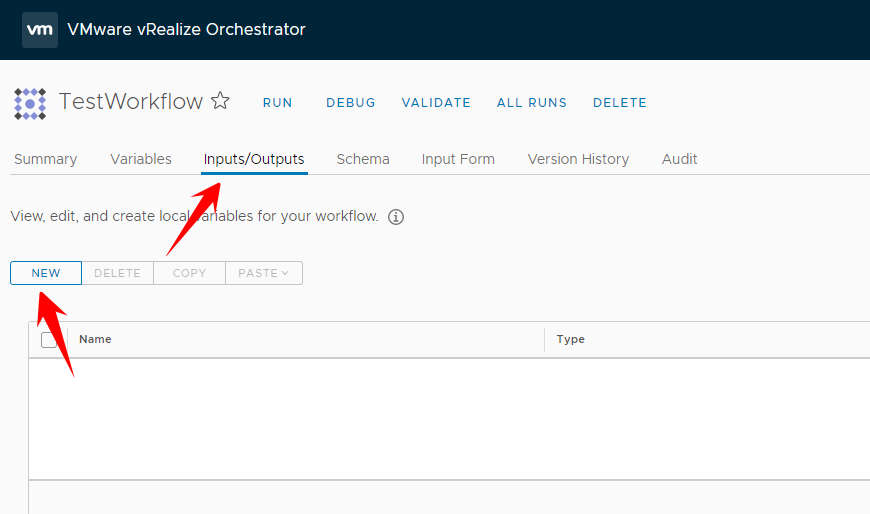


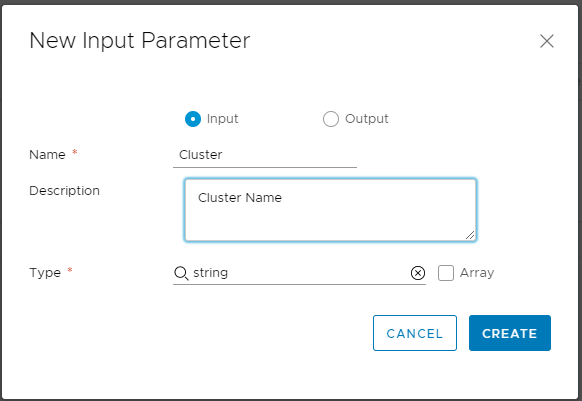


Give it a proper version and add tags if you want

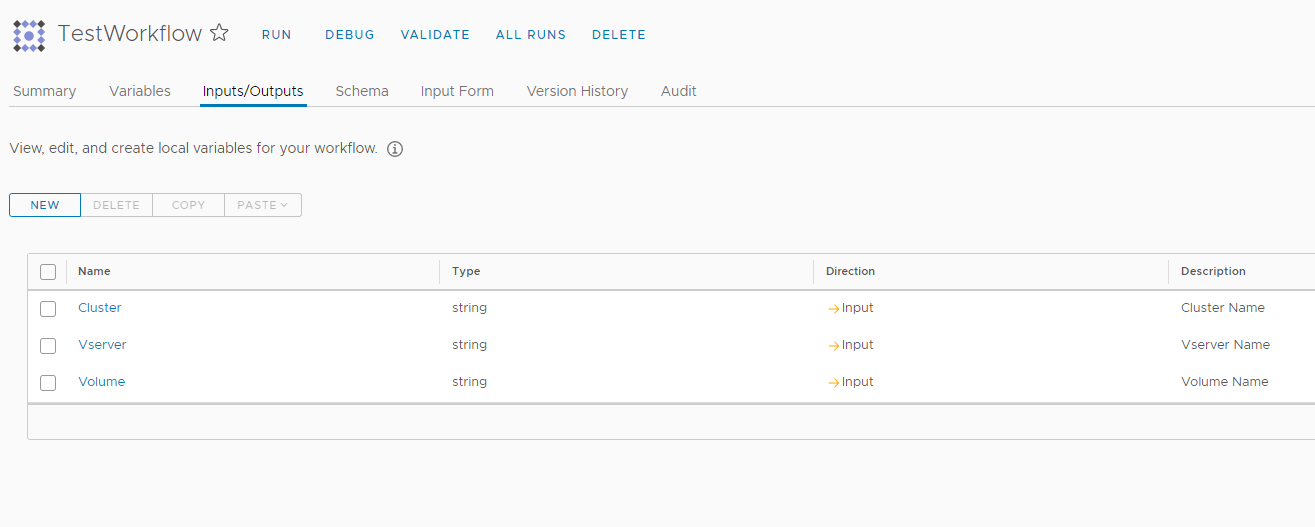


### Add the userinputs





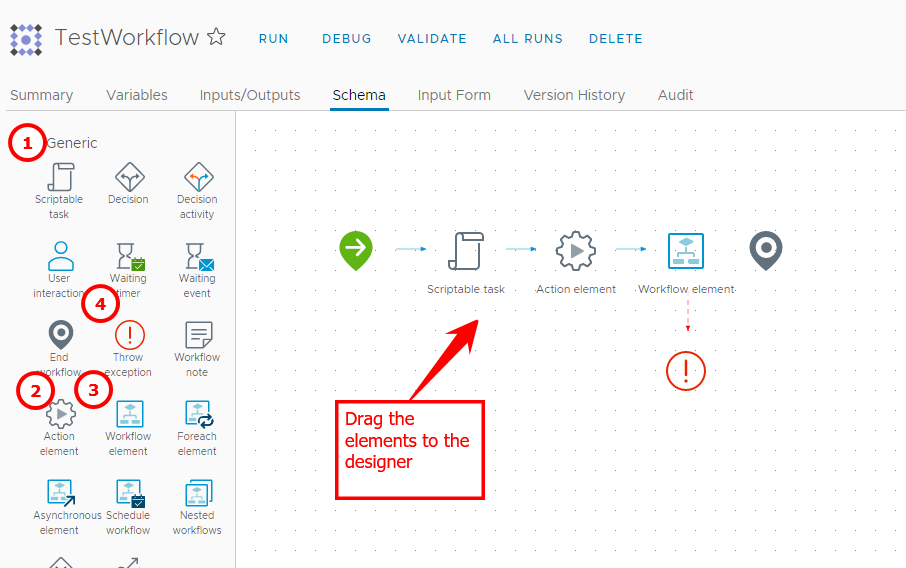
Do the same with the other user input



### Create the schema

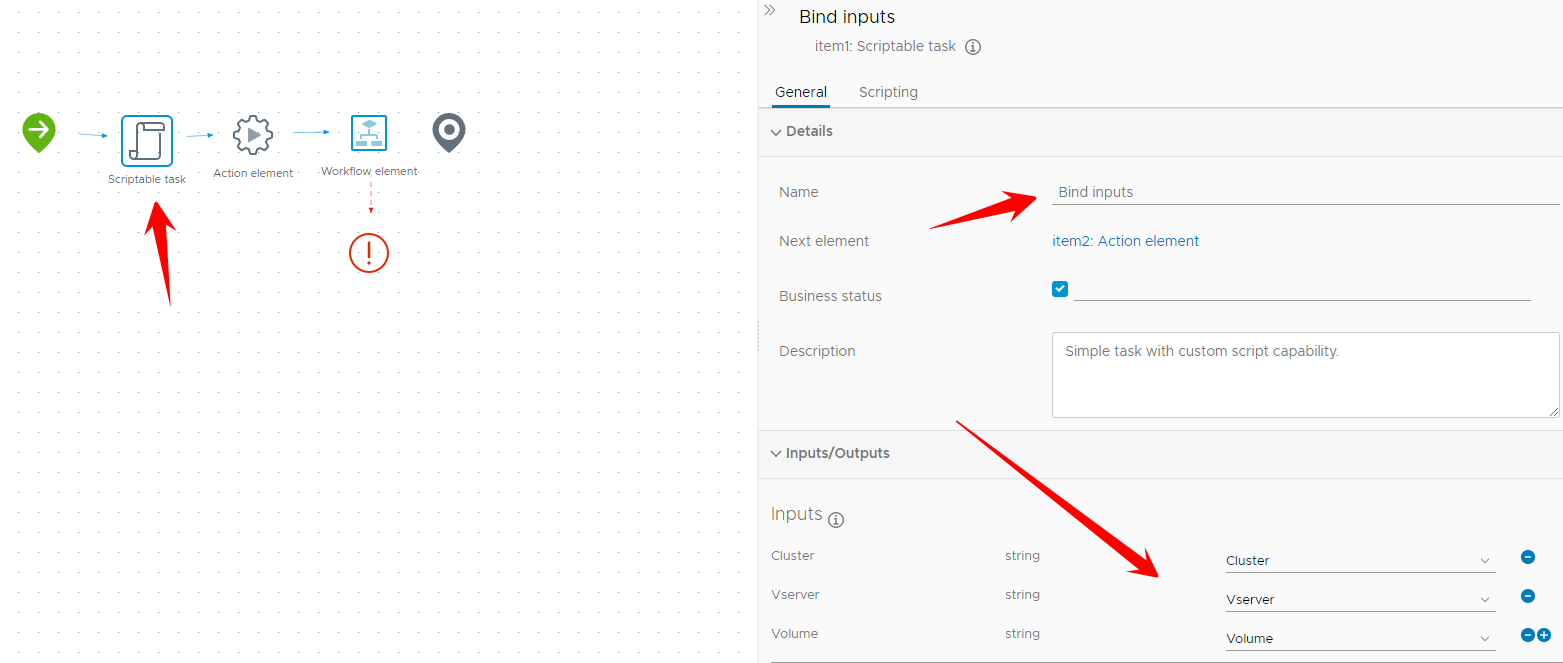
Go to “schema” and add the following 4 items

* Scriptable task
* Action element
* Workflow element
* Throw exception



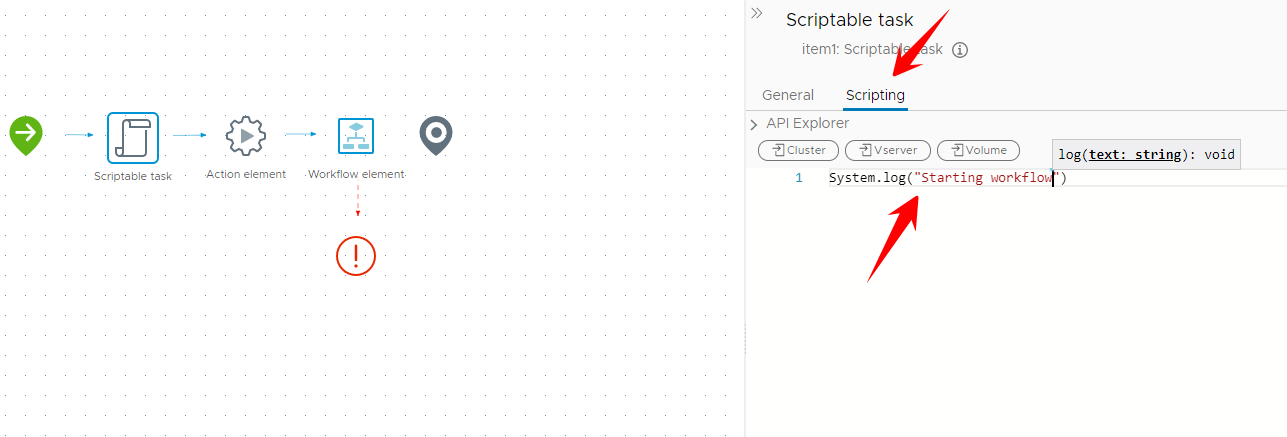
**On the “scriptable task” :**

Add the 3 inputs and bind them to the previously created userinput (we are binding them to 3 variables). Give the task the meaningful name “Bind inputs”



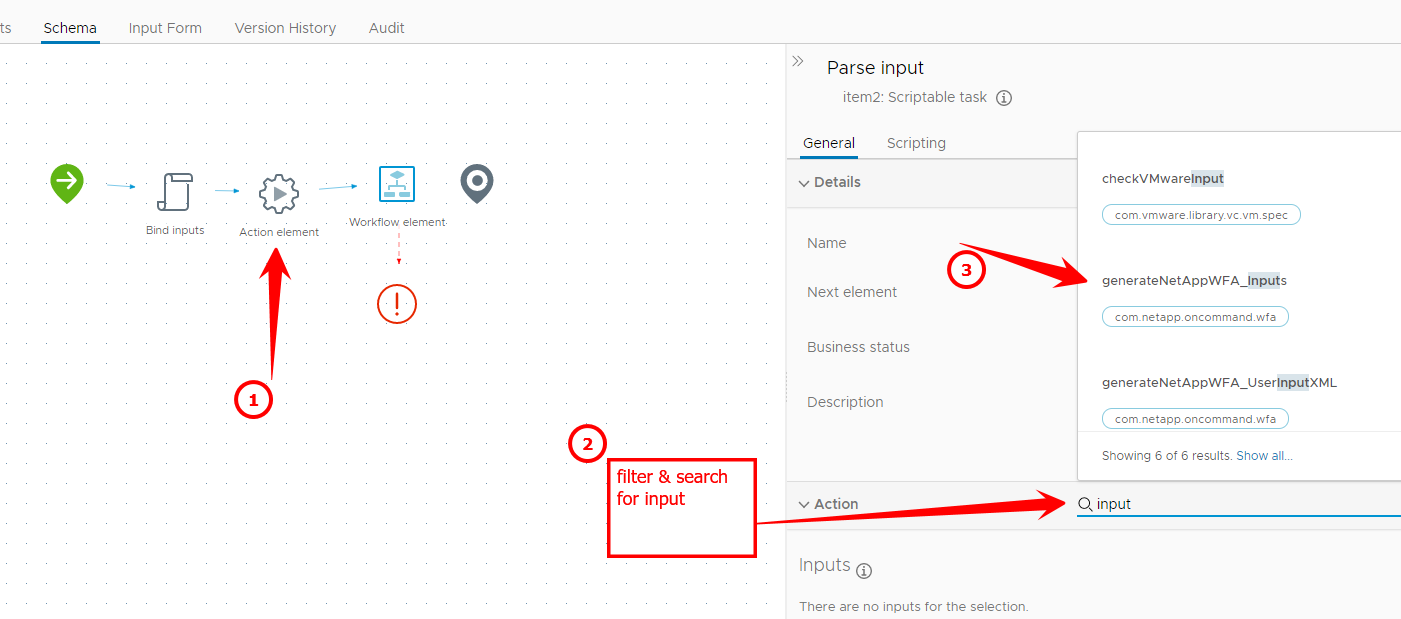
On the scripting tab, add this line of code.

System.log(“Starting workflow”)

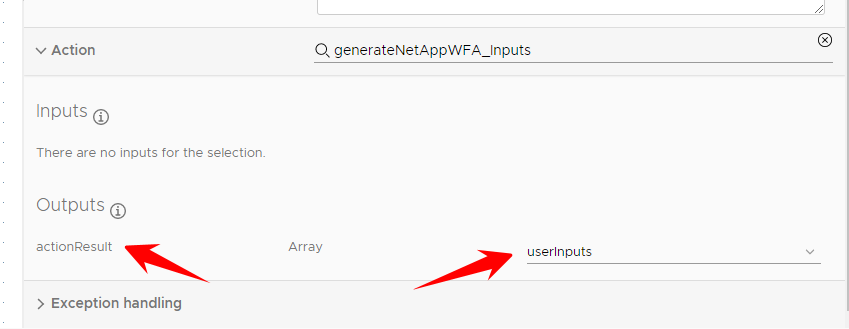


**On the “action element”**

Give it a proper name “Parse input” and Search for the action “generationNetAppWFA\_inputs”



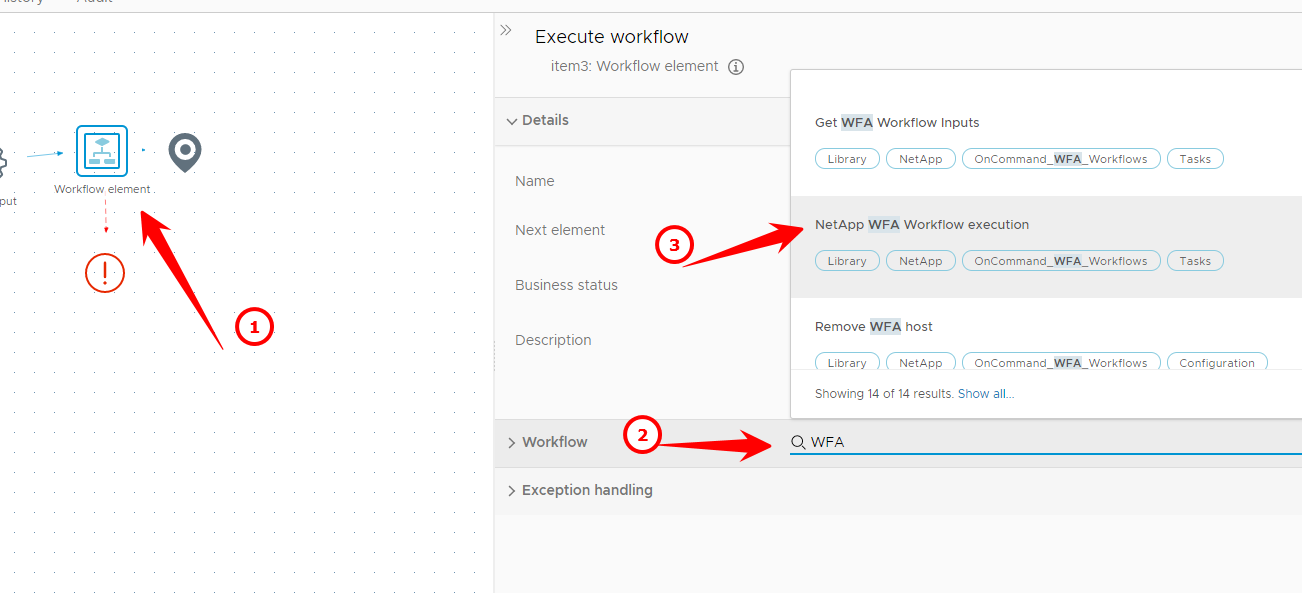
In the outputs, set the actionResult to “userInput”



**On the Workflow element :**

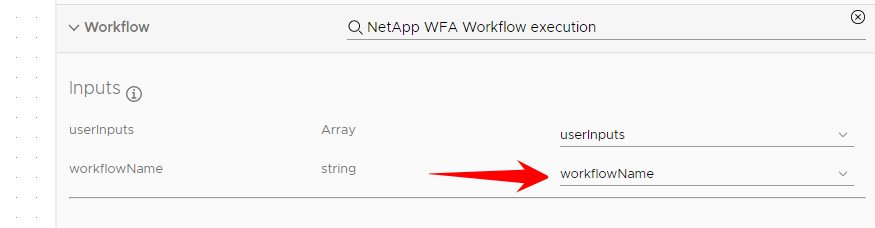
Give it a proper name “Execute workflow”

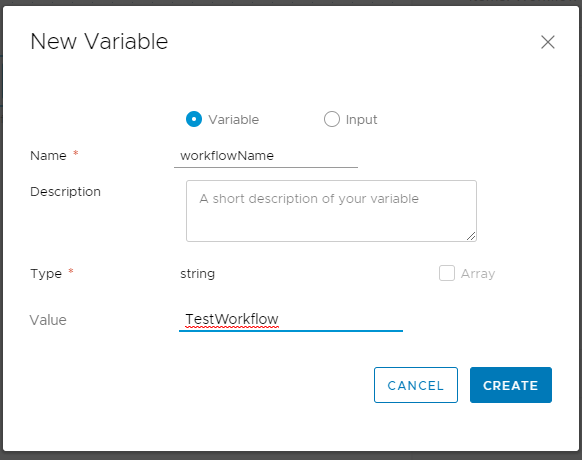
And search for the workflow “NetApp WFA Workflow execution”



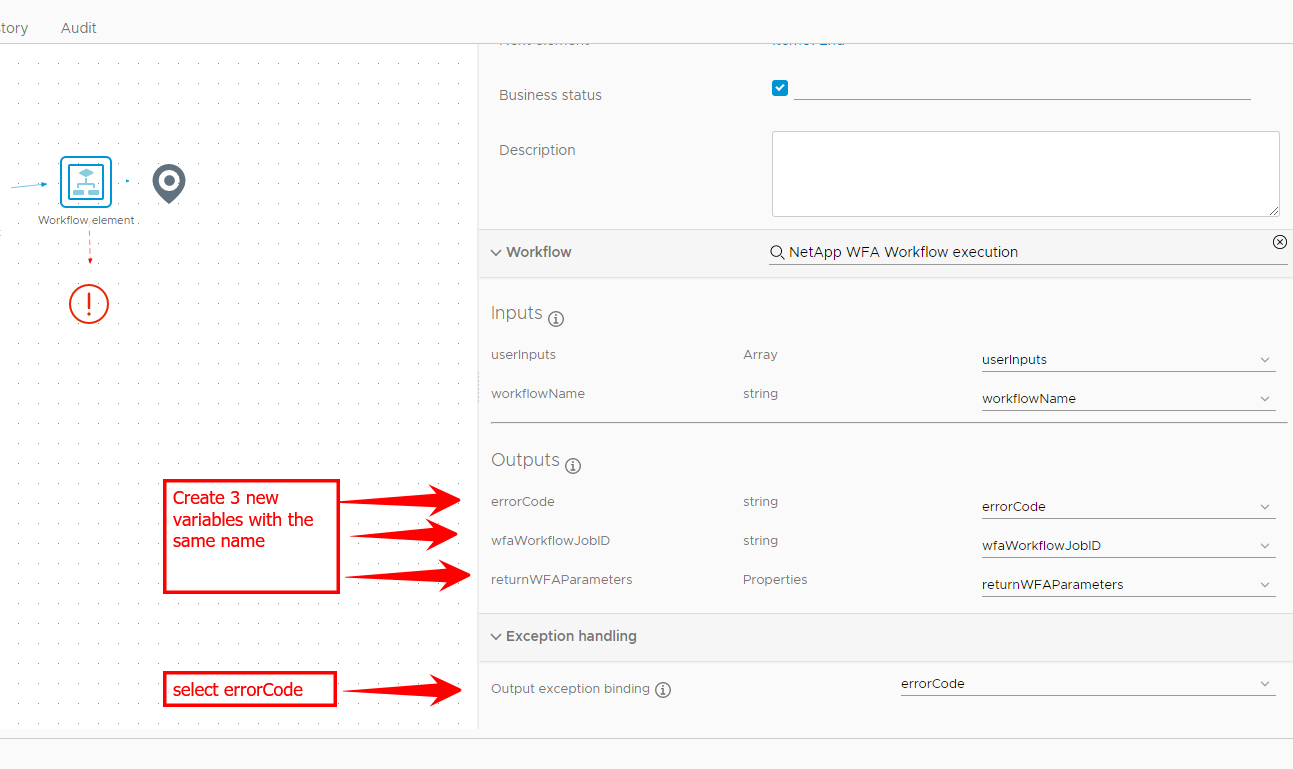
On the “workflowName” ; create a new variable and as value, use your WFA workflowname (TestWorkflow in my case)

The input “userInputs” must be mapped to the previously created variable “userInputs”



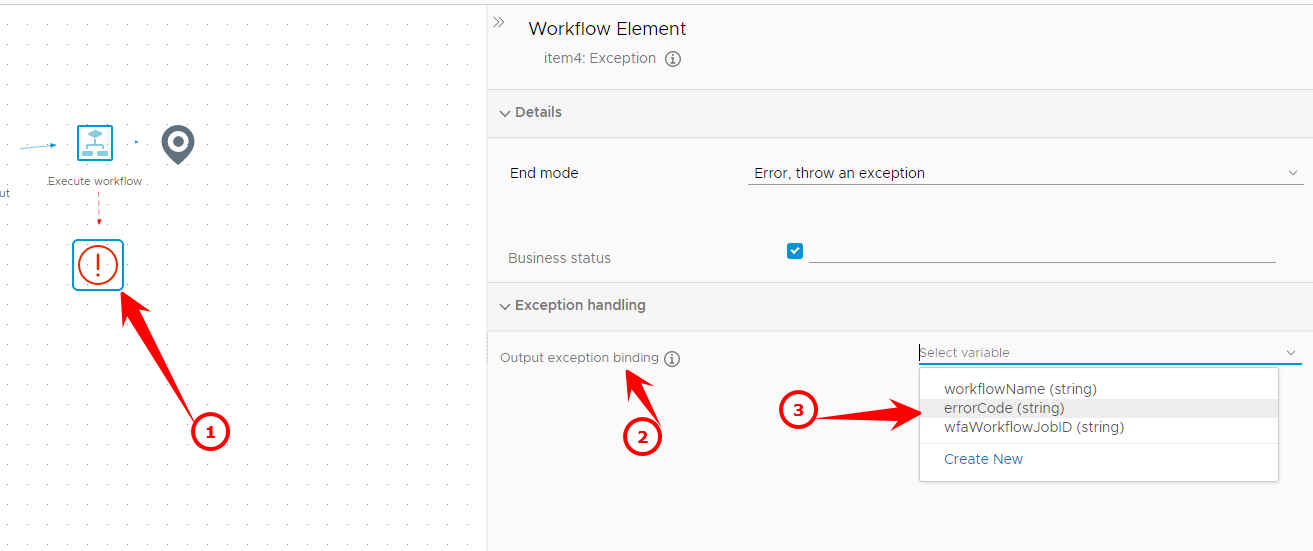


Complete the “outputs” and the “Exception binding”

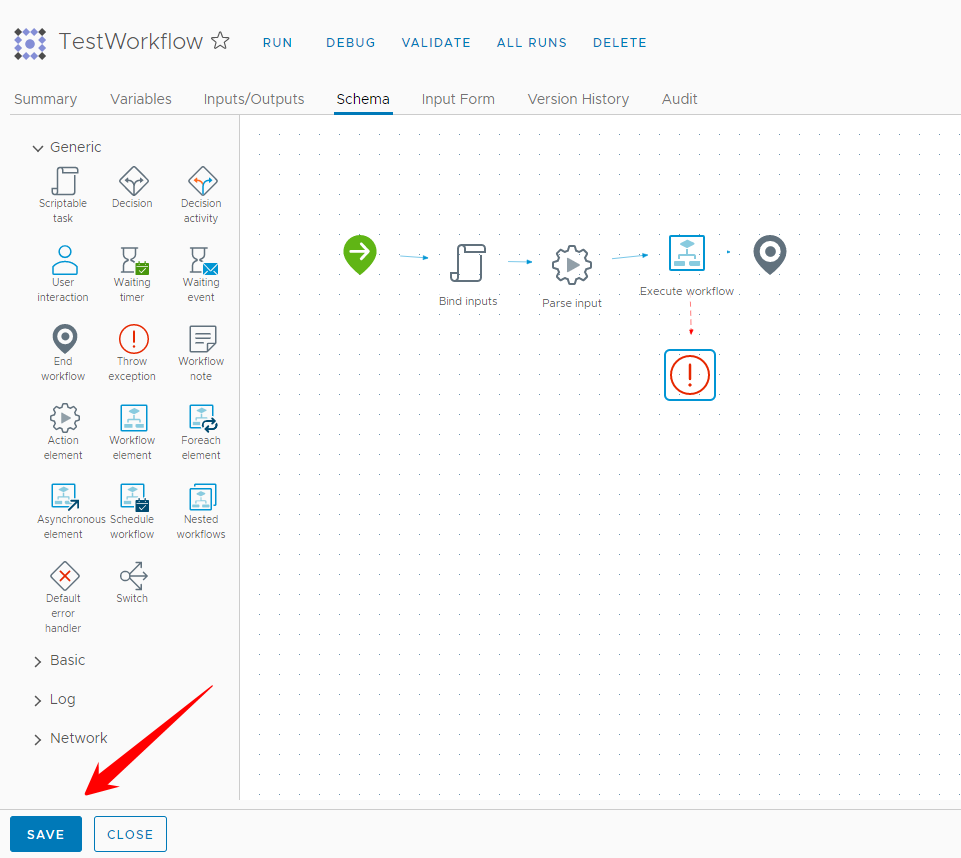


**On the “Throw Exception” element :**

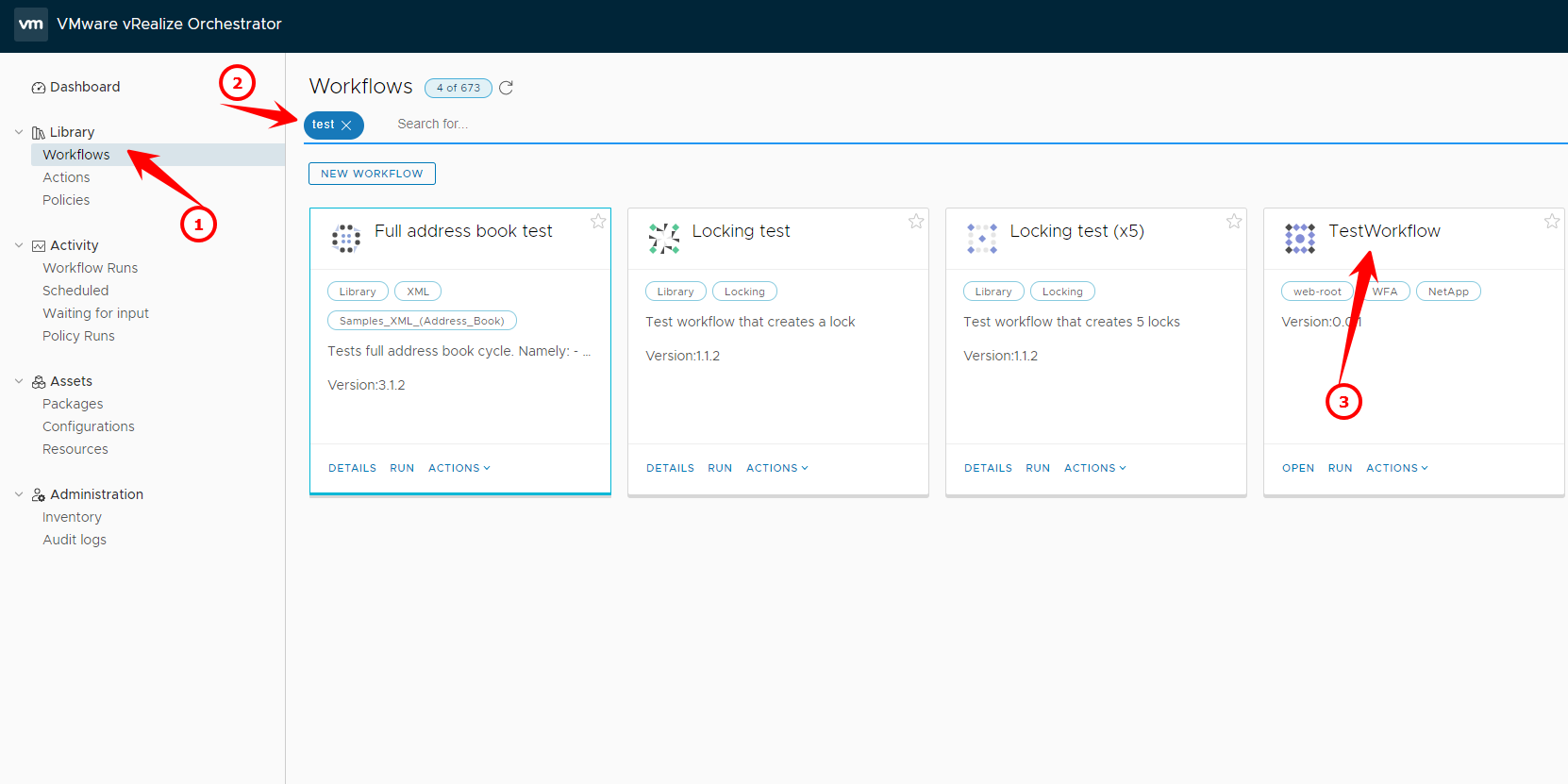
Bind the variable errorCode again

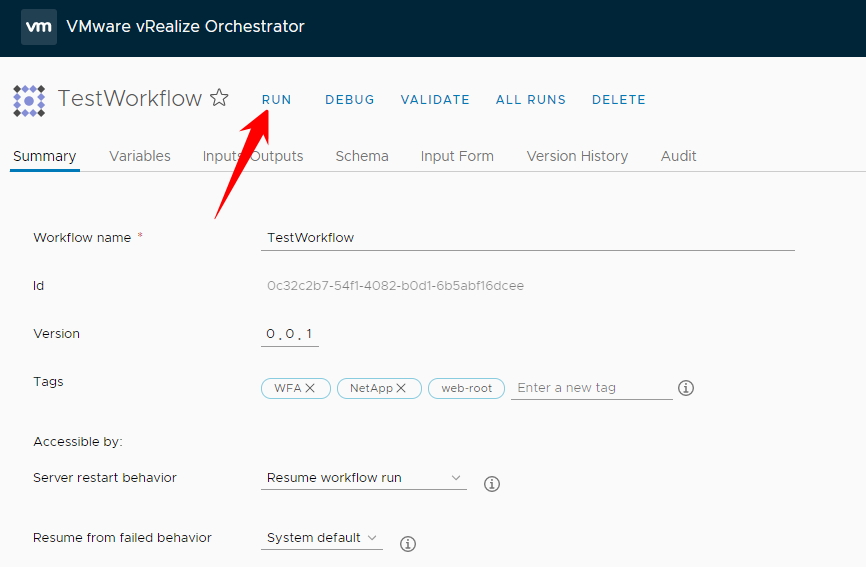


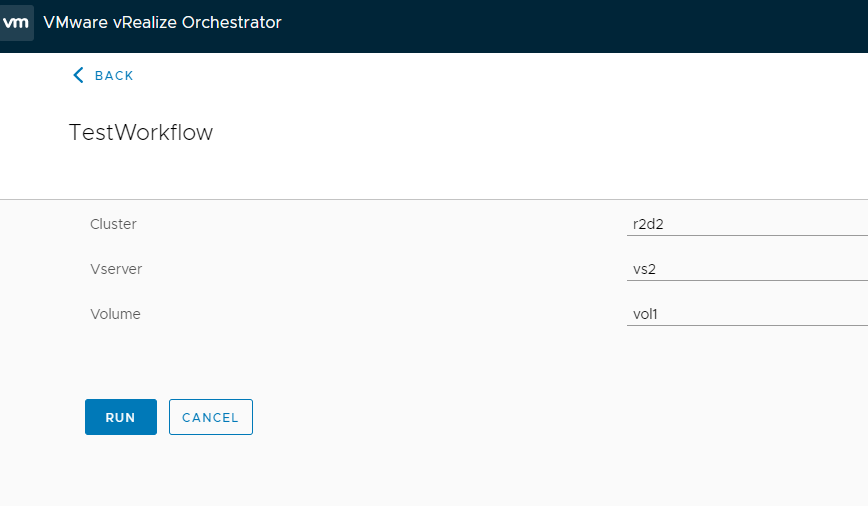
Save the workflow



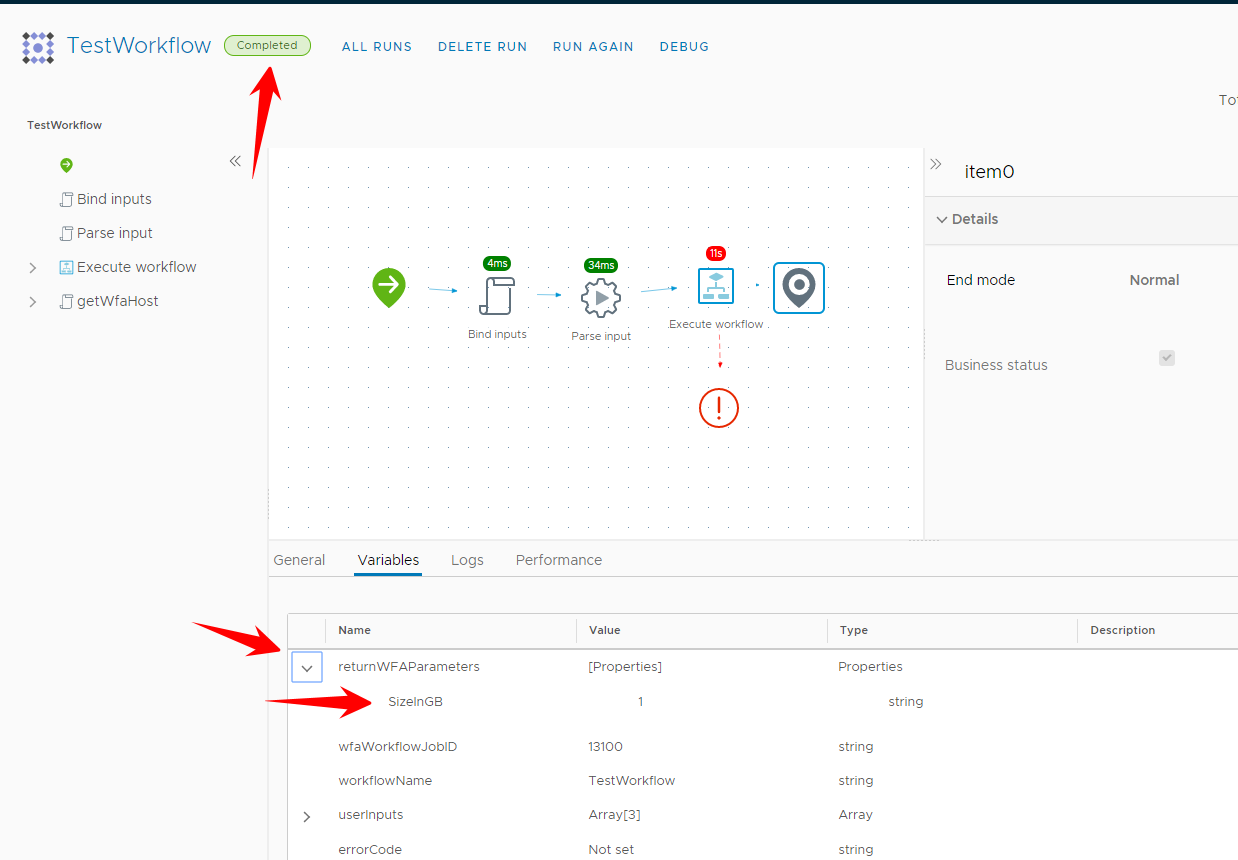
## Test your new workflow

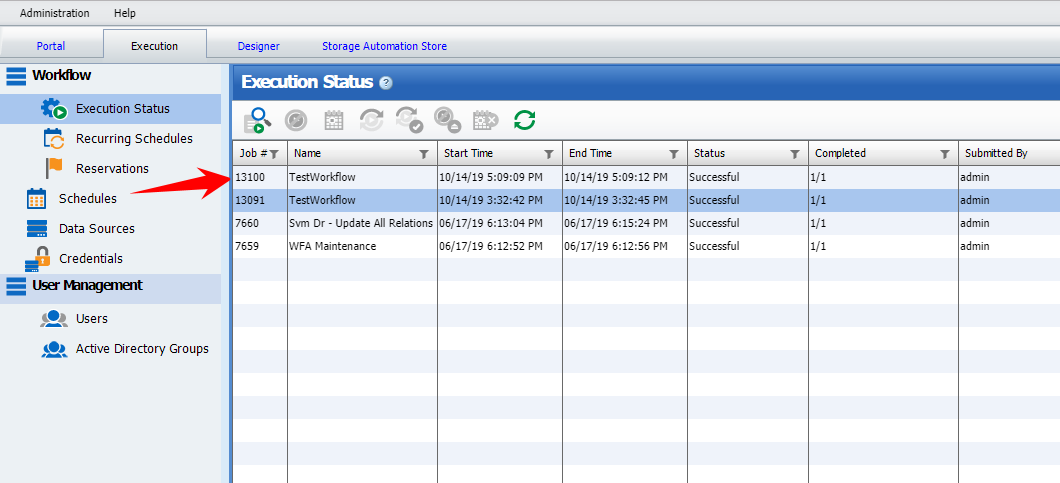






Note the returnParameters from WFA (SizeInGB in my case)





## Fancy you workflow

Of course, you can make the workflow more fancy. vRealize has all kinds of methods to visualize you userinput

