

# Kelverion Automation

## Standard Tasks Application for Azure Automation

### Deployment Guide

Version 2.6

Email: [info@kelverion.com](mailto:info@kelverion.com)

Web: [www.kelverion.com](http://www.kelverion.com)

# 1 Table of Contents

<b>2</b>	<b>Overview.....</b>	<b>3</b>
<b>3</b>	<b>General Configuration Steps.....</b>	<b>4</b>
3.1	Pre-Installation Information .....	4
3.1.1	<i>Kelverion Items Required .....</i>	<i>4</i>
3.1.2	<i>Other Products Required .....</i>	<i>4</i>
3.2	Installation Steps.....	4
3.3	Persistent Data Store.....	5
3.3.1	<i>Encrypting Data.....</i>	<i>5</i>
3.4	Configure the Automation Portal .....	5
3.4.1	<i>Portal Queries .....</i>	<i>5</i>
3.4.2	<i>Portal Offerings.....</i>	<i>6</i>
3.5	Create an Azure system assigned managed identity for starting runbooks.....	6
3.6	Create a Hybrid Worker .....	6
3.7	Load Integration Modules .....	6
3.8	Configure your Automation account using the Runbook Studio.....	6
3.8.1	<i>Design Time (Smart Connections).....</i>	<i>6</i>
3.8.2	<i>Azure Variables.....</i>	<i>9</i>
3.8.3	<i>Azure Runtime connections.....</i>	<i>10</i>
3.9	Import Runbooks using the Runbook Studio .....	12
3.10	Runbook Customisation .....	13
3.11	Runbook Process Flow .....	13
3.11.1	<i>Gathering the Data .....</i>	<i>14</i>
3.11.2	<i>Worker Runbook Input Data.....</i>	<i>14</i>
3.11.3	<i>Returning the Data to your Service Desk .....</i>	<i>14</i>
3.11.4	<i>Calling Multiple Worker Runbooks .....</i>	<i>15</i>
3.12	Scheduling runbook execution.....	16
3.12.1	<i>Logic Apps .....</i>	<i>16</i>
<b>4</b>	<b>Data Discovery .....</b>	<b>19</b>
4.1	PDS Discovered Data .....	19
4.2	PDS Manual Insert Data.....	19
4.3	Portal Static Data.....	19
4.4	Portal Query Data .....	19

## 2 Overview

Businesses would like to give team leaders and managers access to create their own users and groups via self-service facilities to reduce the workload on the heavily utilised service desk and front-line support teams. Equally with I.T. Security being in the forefront of everyone's minds it is a challenge that requires careful balancing between providing the right facilities to end users and not compromising the security or integrity of your configuration.

Therefore, companies are increasingly looking for Automation to improve consistency and supportability, along with the self-service facilities whilst maintaining the security controls. To meet this need we have built the Standard Tasks Application for Azure Automation.

The Kolverion Standard Tasks application for Azure Automation has been developed based on our experiences of common IT tasks. It is designed to meet the needs of customers who want to provide a simplified experience of carrying out common IT tasks as well as make use of Microsoft's hybrid cloud and on-premises architecture.

All the actions are carried out by your Azure Automation account, so every change is logged and carried out in a consistent way.

The Runbooks have been written using the Runbook Studio authoring application and leverage the integration and smart discovery capabilities provided by the Integration Module for SQL Server. These Integration Modules are also available in the PowerShell Gallery.

## 3 General Configuration Steps

This solution was designed with the Keverion Automation Portal. The following configuration steps are a guideline for setting the solution with this front end.

**Important:** If you are using a different service desk to drive the runbooks then you will need to adjust the runbooks.

### 3.1 Pre-Installation Information

The Standard Tasks application package contains the following elements:

- Persistent Data Store (PDS) SQL configuration script
- Standard Tasks Azure Automation Runbooks
- Azure Automation Service Export

#### 3.1.1 Keverion Items Required

The application requires the following Keverion products:

- Keverion Runbook Studio
- Keverion Integration Module for SQL Server
- Keverion Integration Module for Keverion Automation Portal
- Keverion Automation Portal

If you do not already have Keverion Integration Modules, Keverion Automation Portal, or the Keverion Runbook Studio they can be downloaded for evaluation from our website.

This guide assumes that you have already installed the Runbook Studio and the Automation Portal. If you have not yet installed those products, then please do so before you continue. Each of the product downloads contains its own documentation to guide you through the initial configuration.

#### 3.1.2 Other Products Required

The following Microsoft PowerShell modules are required:

- ActiveDirectory
- Az.Accounts
- Az.Automation
- Az.Compute
- Exchange (For Mailbox Offerings)

## 3.2 Installation Steps

As a guide the steps taken are as followed:

1. Configure the PDS database
2. Import and configure the Portal Service (If using the Keverion Automation Portal)
3. Configure the Service Offerings (If not using the Keverion Automation Portal)
4. Create the Azure components
  - a. Resource Group
  - b. Automation Account
  - c. Managed Identity
  - d. Load Integration Modules

- e. Import Runbooks
  - f. Create Smart Connections
  - g. Create Azure Variables
5. Create the Logic Apps

### 3.3 Persistent Data Store

The Persistent Data Store or PDS is a SQL Server database that is used by these runbooks to allow all the actions that the runbooks take to be carried out in a robust way. The use of the database at each “step” allows us to design the runbooks such that each runbook is simple and can be considered a discrete unit. In programming terms, it allows the runbooks to be modular.

To best exploit the power and flexibility of Azure, the PDS should be deployed to a SQL instance within your Azure subscription.

We will be using a PDS on Azure using Azure's SQL offerings, rather than building a VM and installing SQL. This allows us to deploy the SQL instance and PDS database quickly and with the minimum of maintenance.

1. Create a new Azure SQL Server. Create the SQL Server in a New Resource Group "Automation", ensure that the "Allow azure services to access server" check box is ticked. This means that ALL Azure resources will be allowed to access the SQL Server through the firewall
2. Give your desktop access to the SQL Server through the firewall
3. Create a new Database "AutomationData". It's important to use this name for the PDS, as it is held within the runbooks. The **BASIC** tier is ample performance for testing and evaluation of the application. As it is trivial to scale up or down the databases this should also be the starting point for your deployments unless you know that there is going to be a high volume of alerts right from the outset.
4. Connect to the database using SQL Management Studio
5. Execute the SQL script (*PDS\_STSK.sql*) from the package to Create the database tables and views.

#### 3.3.1 Encrypting Data

The data stored in the PDS table [dbo.].[RequestData] may have data that is sensitive and requires encryption. The SQL Integration Module allows the runbooks to handle encrypted data, so it is best practice to encrypt the following columns:

- RequestData
- RunbookData

### 3.4 Configure the Automation Portal

Import the service request definition `Kelverion_STSK_Services.export`

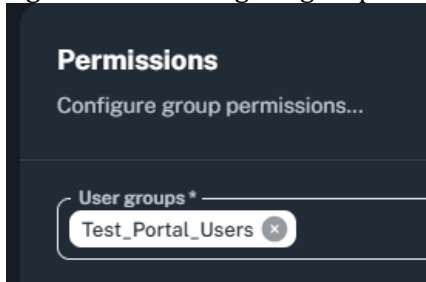
#### 3.4.1 Portal Queries

The following Connection must be updated to point to the customers PDS

- AutomationData

### 3.4.2 Portal Offerings

Check the User Group matches the correct Group and change accordingly.  
e.g. Default offering AD group is Test\_Portal\_Users.



**Important:** Changing the offering inputs will require you to adjust the runbook that is called.

### 3.5 Create an Azure system assigned managed identity for starting runbooks

In the automation account, go to Identity. Enable the System assigned managed identity.

Allocate the role of “Automation Operator” for the resource group that the automation account is in.

### 3.6 Create a Hybrid Worker

The runbooks require a Hybrid Worker to use the on-premise integration modules (ActiveDirectory \ Exchange).

### 3.7 Load Integration Modules

The Integration modules will need to be installed in the following locations

- The Automation Account ("Assets" > "Modules")
- The Hybrid Worker(s)
- The machine where you are running the Runbook Studio.

The modules can easily be installed on the machine from the PowerShell Gallery.

Visit <https://docs.microsoft.com/en-us/azure/automation/automation-update-azure-modules> for more information about loading Integration modules into your Automation Account.

The Integration Modules mention in section 3.1.1. and 3.1.2 are required.

### 3.8 Configure your Automation account using the Runbook Studio

Connect the Runbook studio to your target Azure subscription. You can find more information on the initial configuration of the runbook studio on pages 5 and 6 of the User's Guide.

#### 3.8.1 Design Time (Smart Connections)

The runbook studio needs to have connections configured for use at design time, these smart connections are used by the discovery process within the Runbook Studio to allow the Runbook Studio to discover information about your target systems. This discovery process accelerates the process of runbook development.

Create the following smart connections within the Runbook Studio.

AutomationData	<div><div>Update Smart Connection</div><div><div>Name</div><div>AutomationData</div></div><div><div>Connection type</div><div>Kelverion.SqlServer</div></div><div><div>Tenant</div><div>KUK</div></div><div><div>Description</div><div>Smart Connections created via manual consolidation</div></div><div><div>ServerName* ⓘ</div><div>kuk-lab-sql.database.windows.net</div></div><div><div>UserName ⓘ</div><div>localadmin</div></div><div><div>Password ⓘ</div><div>.....</div></div><div><div>UseWindowsAuthentication ⓘ</div><div>False</div></div><div><div>ConnectTimeout ⓘ</div><div>60</div></div><div><div>OK</div><div>Cancel</div></div></div>
----------------	--

<p>AutomationPortalAPI</p>	<div><div>Update Smart Connection</div><div><div>Name</div><div>AutomationPortalAPI</div></div><div><div>Connection type</div><div>Kelverion.AutomationPortal</div></div><div><div>Tenant</div><div>KUK</div></div><div><div>Description</div><div></div></div><div><div>PortalUrl* ⓘ</div><div>https://svr-svcs-port40.ad.codeswiftly.net:8443/</div></div><div><div>TenantId* ⓘ</div><div>e7594b0e-15aa-4919-b521-16fef1c5ab1b</div></div><div><div>ClientId* ⓘ</div><div>6e31b71b-1653-4126-b292-fe4c06ca7b13</div></div><div><div>Username ⓘ</div><div></div></div><div><div>Password ⓘ</div><div></div></div><div><div>ClientSecret ⓘ</div><div>.....</div></div><div><div>DisableServerCertificateValidation ⓘ</div><div>.....</div></div><div><div>OK</div><div>Cancel</div></div></div>
----------------------------	---



### 3.8.2 Azure Variables

Azure variables allow us to remove static configuration information from the code in our runbooks and store the information in an easy to access place. This helps us to build consistent configuration between all our runbooks. These values are accessed at design time, and at runtime by both the Hybrid workers, and the Azure Worker sandboxes.

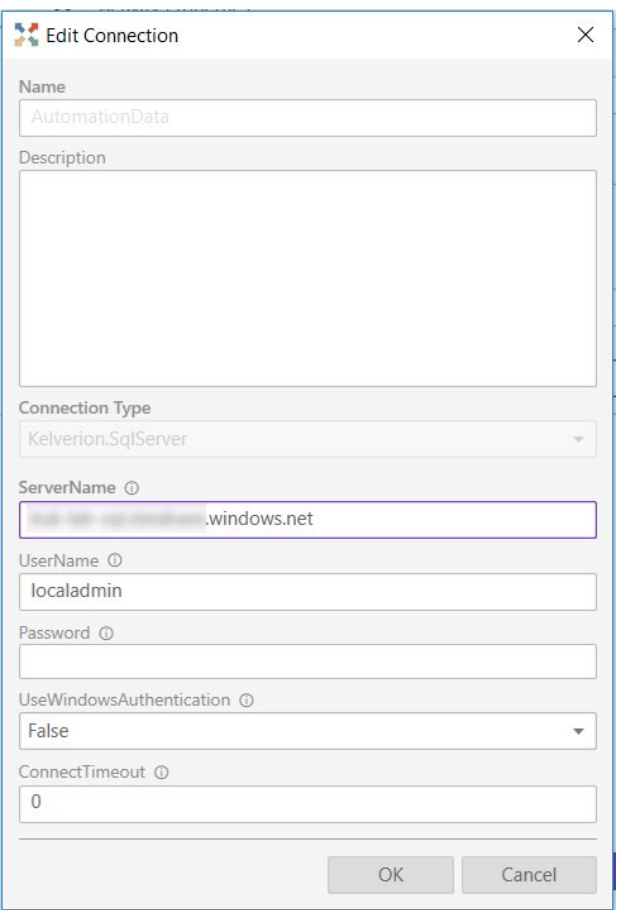
Create the following variables in the Automation Account using the Runbook Studio.

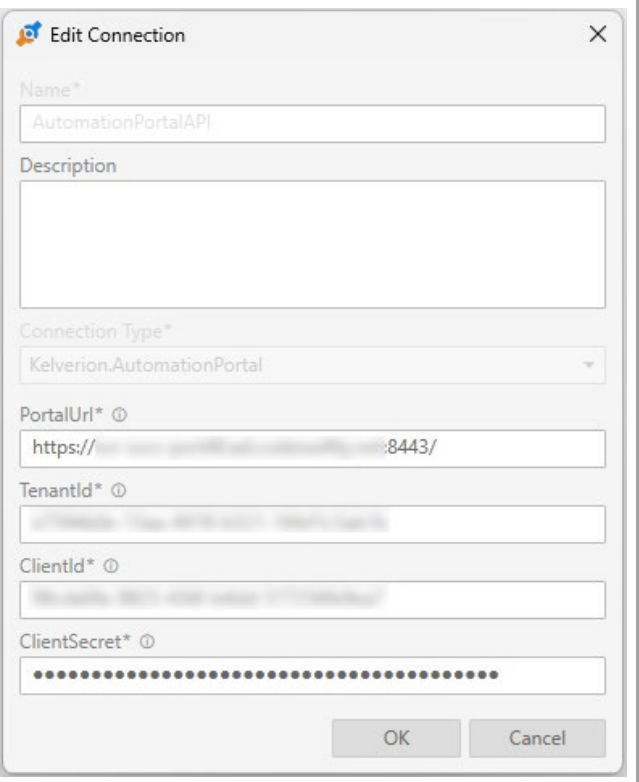
Variable Name	Value
AutomationAccountName	The Name of the Automation Account
ResourceGroupName	The Resource Group that contains the Automation Account
HybridWorkerGroup	The name of the Hybrid Worker Group name
TenantID	The TenantID of the host environment
SubscriptionID	The name of the subscription that you are launching the runbooks in
VMSubscription	The subscription where your Azure VMs are

3.8.3 Azure Runtime connections

Azure connection assets are used at runtime to define a reusable connection configuration. The connection types available are dependent upon module that have been loaded into your Automation Account. If you cannot see the connection types listed below when you attempt to create the connection assets, then this indicates an issue with the modules that are loaded into your automation account. Please verify that all the required modules are loaded.

Create the following Connection in Azure using the Runbook Studio.

AutomationData	The "AutomationData" connection asset in Azure defines the connection that will be used at runtime for the runbooks to connect to the PDS database.	
----------------	---	---

AutomationPortalAPI	<p>The "AutomationPortal" connection asset in Azure defines the connection that will be used at runtime for the runbooks to connect to the Kelverion Automation Portal Database.</p> <p>N.B. Only required if you are using the Kelverion Automation Portal as a front end for the service.</p>	
---------------------	---	---

### 3.9 Import Runbooks using the Runbook Studio

Connect to the Automation Account using the Runbook Studio.

Check that you have the correct Automation Account set as the default target (if there is more than one Automation Account associated with the subscription)

Open each of the following runbooks, then "publish draft" and then "publish" the runbooks.

Runbook Name	Brief Description
Kelverion_STSK_10-1_Worker_Restart-Service	Restarts a service on a selected device
Kelverion_STSK_11-1_Worker_Remote-Reboot	Reboots a selected device
Kelverion_STSK_12-1_Worker_Remote-Shutdown	Shutdown one or more devices
Kelverion_STSK_13-0_Process-Start-AzureVM	Start one or more Azure virtual machines
Kelverion_STSK_14-0_Process-Stop-AzureVM	Stop one or more Azure virtual machines
Kelverion_STSK_15-0_Process-Restart-AzureVM	Restart one or more Azure virtual machines
Kelverion_STSK_20-1_Worker_AD-Create-User	Creates a new AD user account
Kelverion_STSK_21-0_Worker_AD-Add-Group-Members	Adds group members to an AD group
Kelverion_STSK_22-1_Worker_AD-Reset-Password	Resets an AD user account password
Kelverion_STSK_23-1_Worker_AD-Enable-Disable	Enables or Disables an AD user account
Kelverion_STSK_24-0_Worker_AD-Remove-Group-Members	Removes group members from an AD group
Kelverion_STSK_25-0_Worker_AD-Add-User-To-Groups	Adds an AD user to AD groups
Kelverion_STSK_26-0_Worker_AD-Remove-User-From-Groups	Removes an AD user from AD groups
Kelverion_STSK_30-1_Worker_Ping-Check	Sends an ICMP request to one or more devices

Kelverion_STSK_31-1_Worker_Ping-Url	Checks a specified URL from a chosen location (HybridWorker)
Kelverion_STSK_32-1_Worker_Tracert	Runs a Tracert against a specified IP address from a chosen location (HybridWorker)
Kelverion_STSK_33-1_Worker_Diags-Events	Searches Windows Event Logs on selected chosen device(s). Optional filters available
Kelverion_STSK_35-1_Worker_Clear-Diskspace	Deletes files from (Recycle Bin \ Downloads \ Custom Folder) on one or more selected devices
Kelverion_STSK_40-1_Worker_Add-User-Mailbox	Creates an AD user account and matching Exchange mailbox
Kelverion_STSK_41-1_Worker_Add-Shared-Mailbox	Creates a shared mailbox in Exchange
Kelverion_STSK_42-1_Worker_Delete-Mailbox	Removes or permanently deletes an Exchange mailbox
Kelverion_STSK_80-1_Discovery-AD-Groups.GraphRunbook	Discovery runbook for AD Groups
Kelverion_STSK_80-2_Discovery-AD-Users.GraphRunbook	Discovery runbook for AD Users
Kelverion_STSK_80-3_Discovery-AD-GroupMembers.GraphRunbook	Discovery runbook for AD Group Members
Kelverion_STSK_90-01_KAP_Get_NewRequest	Gathers the request data from the Kelverion Automation Portal.
Kelverion_STSK_92-01_ReturnData	Used as a child runbook to handle return data to the Kelverion Automation Portal.

### 3.10 Runbook Customisation

This application is designed to allow flexibility across multiple service desk applications. The initial runbooks are configured for using the Kelverion Automation Portal as the main user portal for initiating any requests and receiving the status updates.

If you are not using the Kelverion Automation Portal, then the installation engineer will be required to create a gathering runbook (90\_XX) and a return runbook (92\_XX).

### 3.11 Runbook Process Flow

This section covers a more detailed description of how the runbook logic fits together. You should be able to use it to configure and customise the application.

### 3.11.1 Gathering the Data

The runbook `Kelverion_STSK_90-01_KAP_Get_NewRequest` is built to gather data from the Kelverion Automation Portal. If you are using a different service desk to gather your request data, then you will need to build a runbook to retrieve the data and store it into the PDS table `[dbo].[RequestData]`.

### 3.11.2 Worker Runbook Input Data

Each worker runbook has been designed to accept a JSON input, that is stored in a SQL table called `[dbo].[RequestData]`. You can check the description of a worker runbook to see what the data input requirements are:

**E.g.**

```
$StopVM = @'
{
  "Server": {
    "Server": 'Test-Server',
    "ResourceGroup": "KUKLab"
  }
}'@
```

To validate this Code, you can place it into a PowerShell ISE and pipe it to `ConvertFrom-JSON` to see the output.

### 3.11.3 Returning the Data to your Service Desk

The runbook `Kelverion_STSK_92-01_ReturnData` is built to be used as an inline child runbook. It has been configured with the Kelverion Automation Portal, but the same logic should apply if you need to adjust this for a different service desk.

N.B. It is not recommended to use `Start-AzAutomationRunbook` with this runbook, as it was designed as an inline child runbook.

#### Runbook Inputs:

Input Name	Mandatory	Description
<b>ID</b>	Yes	The parent request ID
<b>Message</b>	Yes	The message you want sent back to update in your service desk
<b>Runbook</b>	Yes	Name of the runbook that you wish to log the message about
<b>Activity</b>	No	(Optional) The name of the runbook activity that you want to record the message about
<b>ErMsg</b>	No	(Optional) Any error message to log back
<b>Offering</b>	No	(Optional) Name of the request

<b>State</b>	No	(Optional) The request state e.g. Complete \ In Progress \ Failed
--------------	----	--

The runbook also accepts from the parent runbook the variable **ChildRunbook**.

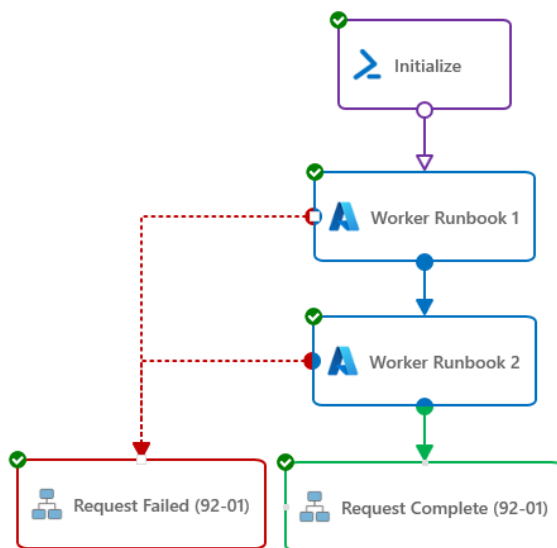
By default, **ChildRunbook** is set as False for the worker runbooks. See the section 'Calling Multiple Worker Runbooks' for more information on this.

### 3.11.4 Calling Multiple Worker Runbooks

The worker runbooks in this solution have been designed to run on their own, or as a child runbook from a parent request. You may have specific scenario where you need to do multiple actions from a single service desk request, that require more than one worker runbook to run. To do this you will need to:

1. Build a parent request runbook
2. Analyse the JSON inputs for each worker runbook you are calling
  - a. Ensure all worker JSON input is gathered in your Parent JSON
3. Call the required worker runbooks from the parent runbook
  - a. Set the ChildRunbook property to \$true
  - b. Pass the Parent request ID to the child runbook

Example Parent Runbook:



The return runbooks process the request as either Complete or Failed by calling the inline child return runbook.

The parent runbook has an input of ID, that is passed through to the child runbooks.

Each worker runbook will pass the additional parameter of ChildRunbook, set to True

```

Parameters ①
(PowerShell expression)

Data source
PowerShell expression

Expression
$RunPar = New-Object PSObject -Property $hash @{
    ID = $ID
    ChildRunbook = $true
}
$RunPar
  
```

### 3.12 Scheduling runbook execution

Once all the runbooks (and other assets) are in place and the runbooks have been tested you will need to schedule the runbooks for repeated execution.

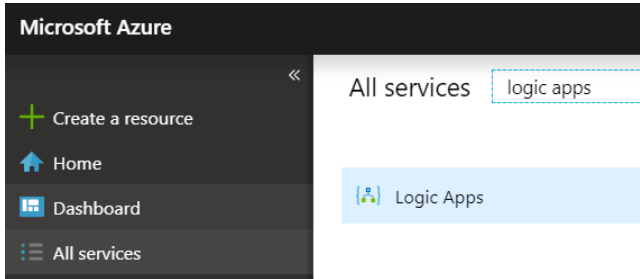
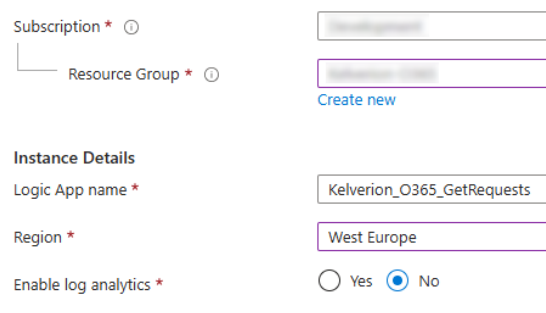
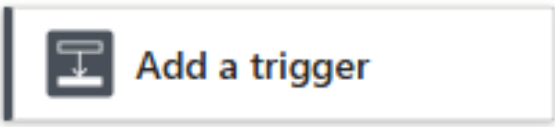
Azure provides the following options for scheduling runbooks:

- Runbook Schedules
- Webhooks
- Logic Apps

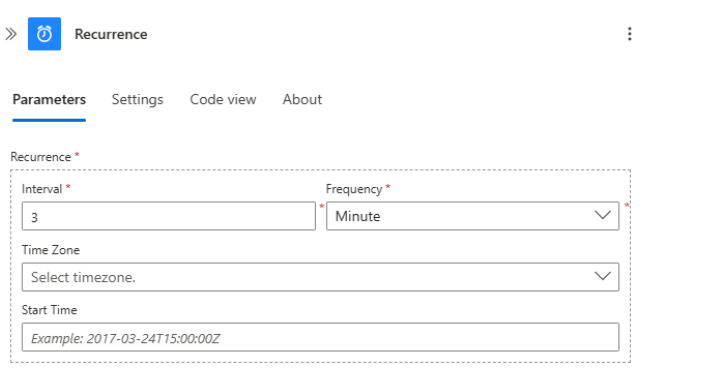
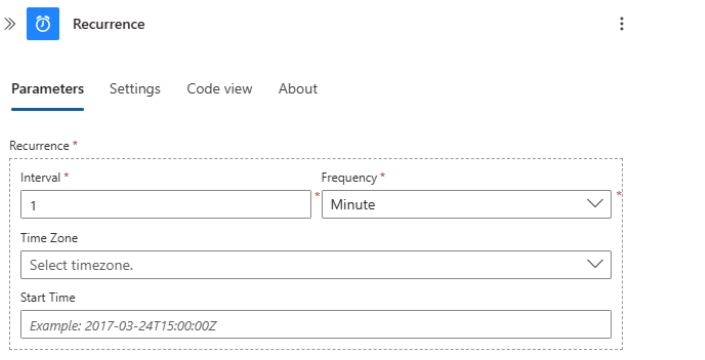
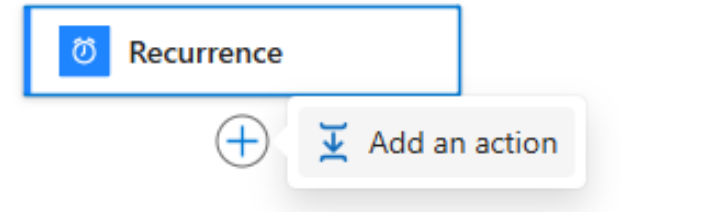
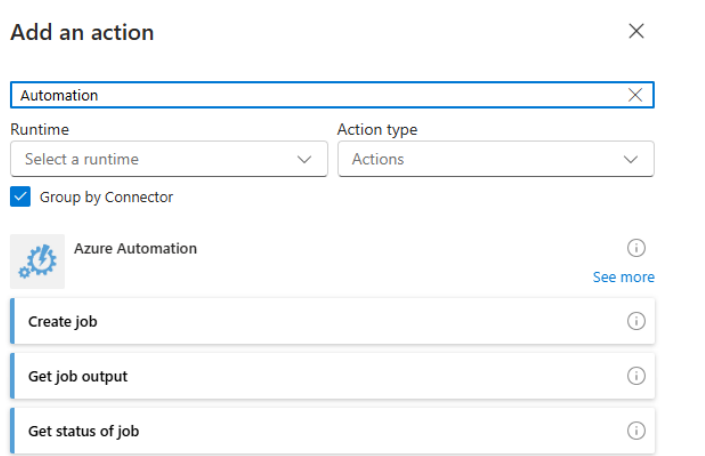
If you are using the Kolverion Automation Portal, then we recommend that you use a Webhook. Please refer to the Kolverion Automation Portal user guide on setting this up. If you are not using a webhook with the Kolverion Automation Portal, then you will need to look at other scheduling methods.

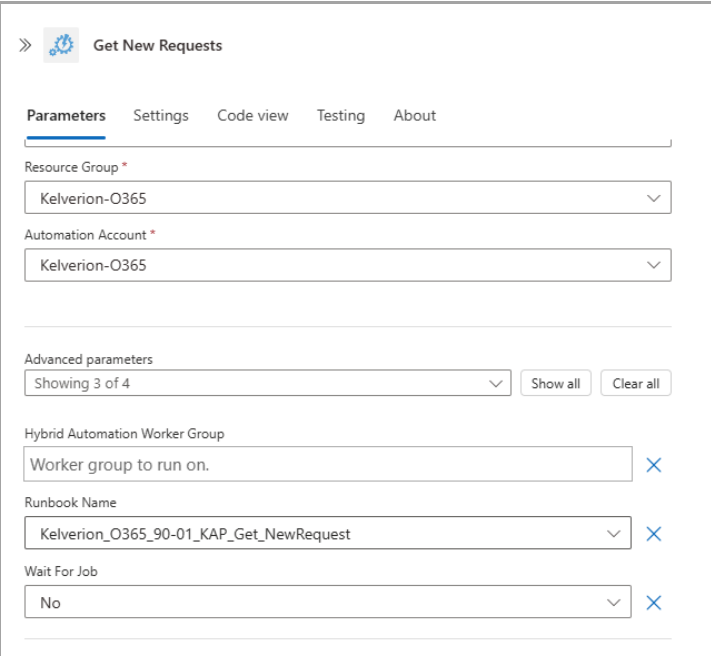
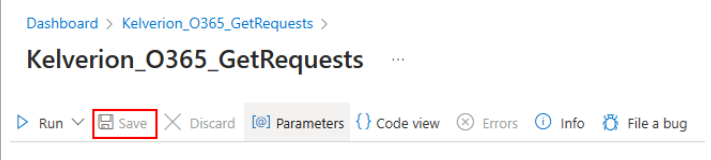
#### 3.12.1 Logic Apps

A single Logic App should be sufficient to schedule a main gathering runbook from your service desk. This Logic App will run on a chosen recurrence to schedule calling the runbook (90-XX) e.g. Kolverion\_STSK\_90-01\_KAP\_Get\_NewRequest.

Step	
Login to the Azure Portal and go to "Logic Apps"	
Create a New Logic App and name it appropriately for this application. e.g. Kolverion_O365_GetRequests  Add it to the same Resource Group that you have deployed the runbooks too	
Open the newly created Logic App and select "Add a trigger"	



Search for “Schedule” and select the Trigger “Recurrence”	
Decide how frequently you need to check for new requests from the portal and edit accordingly.	
Click the + and add an action	
Search for “Automation” and select “Create Job”	

<p>Change the activity name to “Get New Requests”.</p> <p>Select the appropriate Resource Group \ Automation Account.</p> <p>Select the runbook “Kelverion_O365_90-01_KAP_Get-NewRequest” in the Runbook Name.</p> <p>N.B. This activity will require a Logic App Managed Identity with Automation Operator role to the resource group with the runbooks.</p>	
<p>Click Save to finish</p>	

## 4 Data Discovery

The following section lists the data discovery tables used by the runbooks that are present in the PDS.

### 4.1 PDS Discovered Data

The following data has runbooks provided to gather the data:

- AD Users [SaTasks].[ADUsers]
- AD Groups [SaTasks].[ADGroups]
- AD Group Members [SaTasks].[ADGroupMembers]

### 4.2 PDS Manual Insert Data

The following data is stored in the PDS. This data has been determined to rarely change and should be updated by either:

- Populate the PDS tables with a bespoke script
- Populate the PDS tables with a discovery runbook

#### Data Types:

PDS Table: [SaTasks].[MSEExchange]

- Exchange Mailbox Database
- Exchange Mail Domain

### 4.3 Portal Static Data

The following data is stored in static Lists in the automation portal.

If the customer wishes to change this to dynamic data, then they would need to be changed to queries with appropriate backed SQL tables \ views.

#### Data Types:

- User OUs
- AD Domain Name
- Windows Services

### 4.4 Portal Query Data

The following data is gathered from queries in the portal. By default, these queries are pointing at PDS tables, that will have demo data in them.

It is advisable for the end customer to either:

- Populate the PDS tables with a bespoke script
- Populate the PDS tables with a discovery runbook
- Re-point the queries at their own CMDB tables

#### Data Types:

PDS Table: [SaTasks].[Machines]

- Servers

Kelverion UK Limited  
Compass House, Vision Park,  
Chivers Way, Histon  
Cambridge CB24 9AD  
United Kingdom  
Email: [info@kelverion.com](mailto:info@kelverion.com)  
Web: [www.kelverion.com](http://www.kelverion.com)