 Microsoft Power Automate Desktop

Lab 02: Automate web page extraction

Hands-on lab step-by-step

July 2024

Microsoft Power Automate Desktop – Advanced Workshop

Contents

[Microsoft Power Automate Desktop 1](#_Toc171496583)

[Goals for this lab 1](#_Toc171496584)

[Prerequisites 1](#_Toc171496585)

[Exercise 1: Automate a desktop application with PAD 2](#_Toc171496586)

[Terms of Use 20](#_Toc171496587)

# Microsoft Power Automate Desktop

This lab is subject to the Terms of Use found at the end of this document.

## Goals for this lab

|  |  |
| --- | --- |
| After this lab you will be able to:   * Use website UI elements to scrape data from lists * Write logic to navigate through multiple pages on a website * Save web scraped results into an Excel file | The time to complete  this lab is [40] minutes. |

## Prerequisites

Please note that some labs, especially later labs, do reference previous labs in reference to capabilities and previous tasks. The labs have been designed so if you have access to a Microsoft Power Automate Desktop trial, you can get started from most lab without having to complete the previous module to be able to move forward. However, for the best experience that shows the features and functionality that is possible within the product, it is recommended you have completed specific modules before starting some of the labs.

For Lab 01: Automate using UI elements, you need:

* A computer with internet access.
* The application Power Automate Desktop installed in your computer. If you don’t have the application installed, please download it here: <https://go.microsoft.com/fwlink/?linkid=2102613>
* Be able to log into your corporate tenant.

## Exercise 1: Automate a desktop application with PAD

### Task 1: Log into Power Automate Desktop

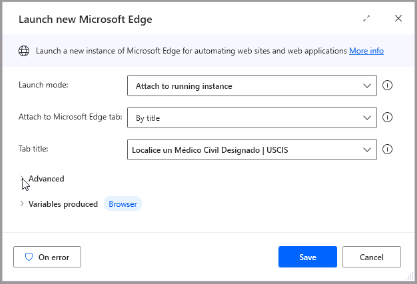
1. Open the **Power Automate Desktop** app on your computer
2. Log into the application using your corporate account
3. If you don’t have a Power Automate Premium license, start a trial by clicking on the **Go Premium** button at the top right corner of the application



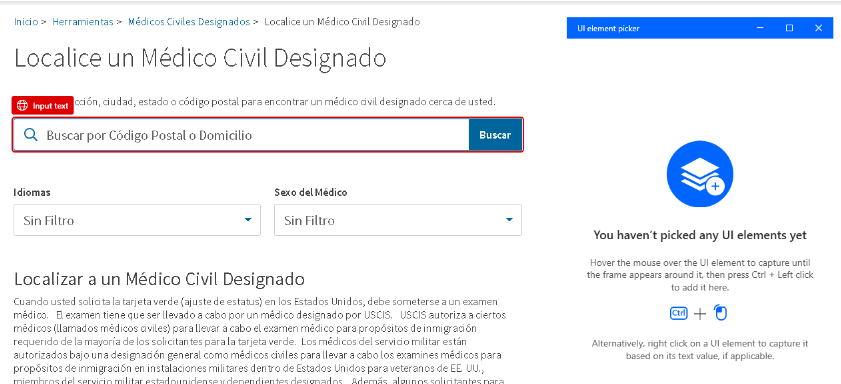
### Task 2: Automate a basic search on a webpage

1. Open a new Microsoft Edge window and navigate to <https://www.uscis.gov/es/herramientas/localice-un-medico-civil-designado>
2. Make sure you create your automations in your own **Personal Development environment**. If you don’t have one, follow the steps here: [Get your developer environment - Power Apps | Microsoft Learn](https://learn.microsoft.com/en-us/power-apps/maker/maker-create-environment#create-your-own-developer-type-environment)
3. Click on **+ New Flow** and create a flow named Lab 02: Automate web page extraction
4. Once on the flow designer, look for action **Launch new Microsoft Edge** from the **Browser automation** actions group
5. Configure this action with the following parameters:

* **Launch mode:** Attach to running instance
* **Attach to Microsoft Edge tab:** By title
* **Tab title:** Localice un Médico Civil Designado | USCIS

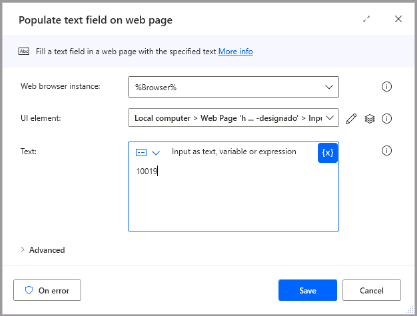


1. Click the **Save** button of the **Launch new Microsoft Edge** action
2. Add the action **Populate text field on webpage** from the group **Web form filling**
3. Expand the **UI element** parameter of this action and select the button **Add UI element**
4. Once the UI element picker is active, while pressing the **Ctrl** key on your keyboard, select the text input field **Buscar por Código Postal o Domicilio**

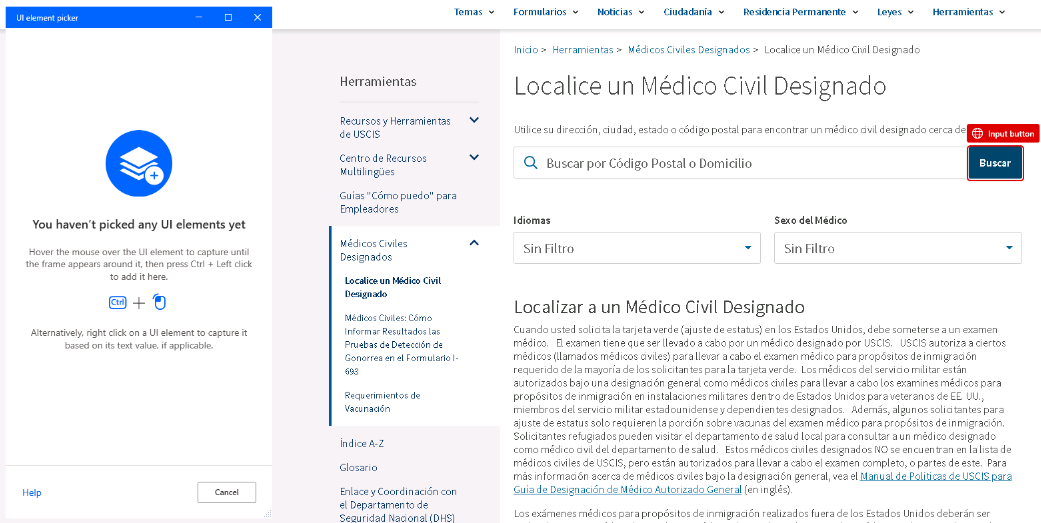


1. Finish configuring this action as shown below:

* **Web browser instance:** %Browser%
* **Text:** Any US ZIP code, for example 10019



1. Click the **Save** button of the **Populate text field on web page** action
2. Add the action **Press button on webpage** to your flow
3. After expanding the **UI element** selector of this action, press **Add UI element**
4. Once the UI element picker is active, while pressing the **Ctrl** key on your keyboard, select the button **Buscar**



1. Click on **Save** for the action **Press button on webpage**

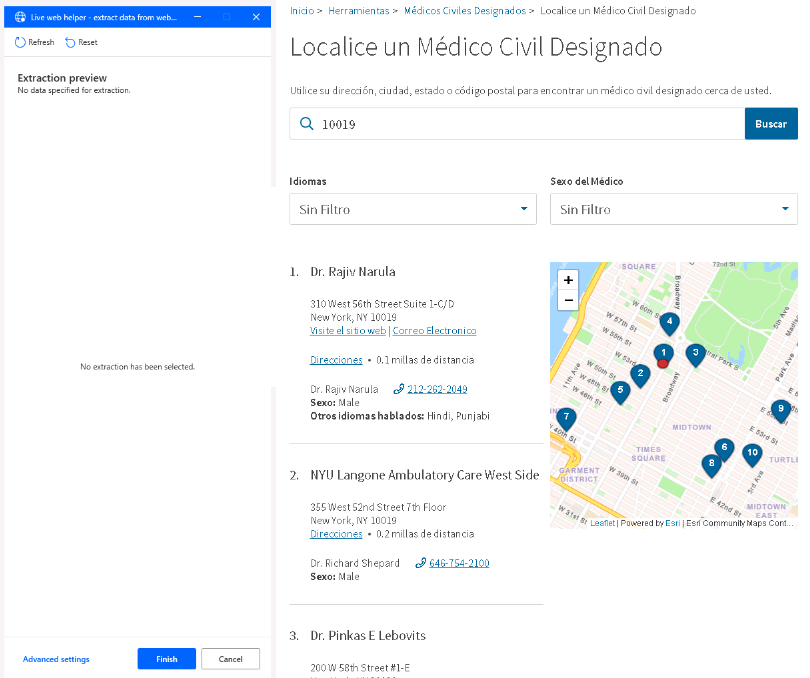
A screenshot of a computer

Description automatically generated

1. **Run** your flow and make sure it’s performing the search on the web page.

### Task 3: Web scrape search results

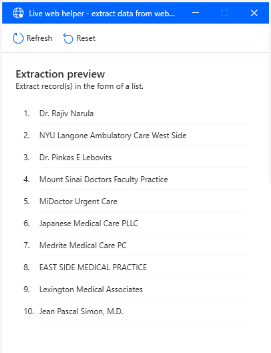
1. Add a **Extract data from webpage** action after the **Press button on webpage** action
2. Before saving this action, navigate back to the website. You should see the **Extraction preview** dialog open right next to the page



1. Right-click on the name of the first doctor on the list. Then select the option **Extract element value > Text (‘Dr. Rajiv Narula’)**



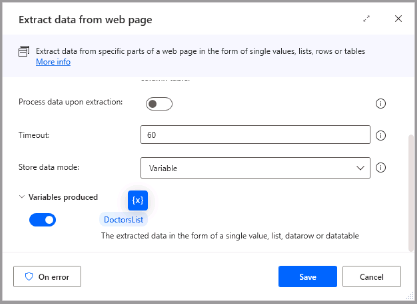
1. Repeat the last step for the name of the second doctor on the list. Notice that the **Extraction preview** dialog now recognizes all the other results on this page and automatically creates a list



1. Add the Street number, City, and Spoken languages as part of your scraped data by right clicking on each attribute and selecting **Extract element value > Text (‘*The text value*’)**



1. Click on the **Finish** button of the Extract preview dialog box
2. Back on the Flow editor, rename the variable where your scraped data will be stored to DoctorsList



1. **Save** this action
2. **Run** your flow and inspect the current value of the **DoctorsList** variable on the Variables pane at the right of the screen. It should look like the image below:

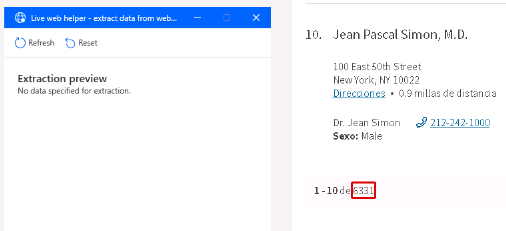


### Task 4: Extract data from multiple search result pages

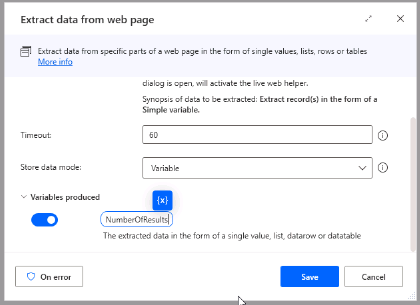
Before starting, notice that the search results on this website are paginated in groups of 10 results. We will now attempt to navigate to each page of the results to extract all the data:



1. Search for the action **Extract data from web page** and drag it to your process, right after thelast action
2. Go to the website and select the total number of results to extract the value.



1. Click on the **Finish** button of the **Extraction preview** dialog box
2. Rename the variable produced by this step to NumberOfResults

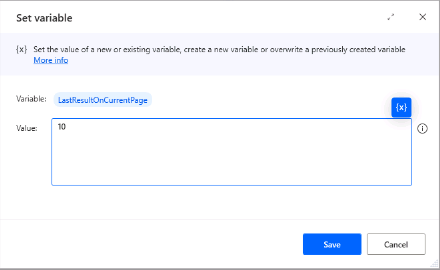


1. **Run** your flow and inspect the current value of the **NumberOfResults** variable on the Variables pane at the right of the screen. Notice that the value is stored as a datatable:

A white background with black lines

Description automatically generated

1. Add a **Set variable** action to your flow. Name this variable LastResultOnCurrentPage and set its value to 10. Click **Save**



1. Add a **Loop condition** action to your flow
2. Configure this action as shown below:

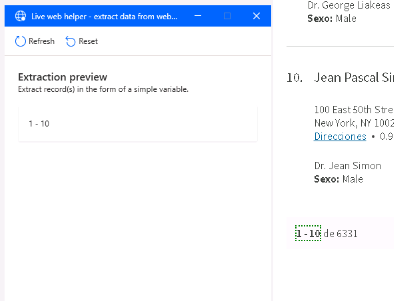
* **First operand:** %LastResultOnCurrentPage%
* **Operator:** Less than or equal to (<=)
* **Second operand:** %NumberOfResults[0][0]%

A screenshot of a computer

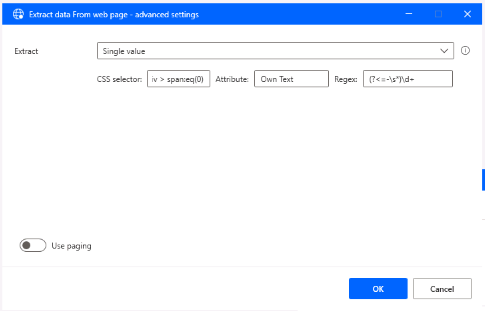
Description automatically generated

|  |  |
| --- | --- |
| Lights On outline | **Pro tip**: Notice that we are using the **[0][0]** notation to select the value on the first row and first column of the **NumberOfResults** datatable variable |

1. Within the **Loop condition** action, drag and drop a new **Extract data from webpage** action
2. On the webpage, extract the value of the number of results as shown here:

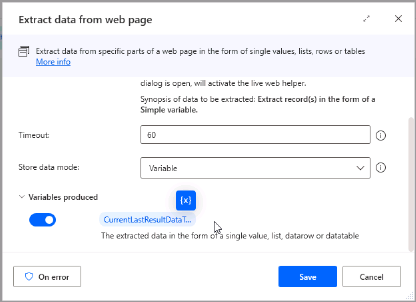


1. Click on the **Advanced settings** of the **Extraction preview** pane
2. On the **Regex** input, type the expression (?<=-\s\*)\d+ and click **OK**



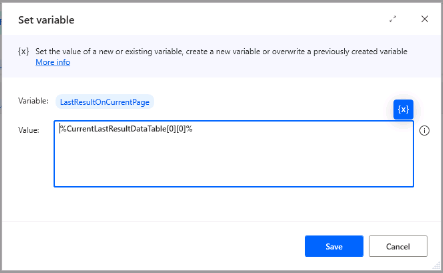
|  |  |
| --- | --- |
| Lights On outline | **Pro tip**: RegEx expressions help to manipulate a text string and extract only the value of interest. In this case, we are extracting the numeric value after the hyphen on a string that follows the pattern **## - ##** (for example, for the string “1 – 10”, the extracted value would be 10. |

1. Click on the **Finish** button of the **Extraction preview** pane
2. Rename the variable produced by this step to CurrentLastResultDataTable. Click **Save** for this action



1. Below the previous action, add a **Set variable** action
2. Configure this action as shown below. **Save** this step:

* **Variable:** %LastResultOnCurrentPage%
* **Value:** %CurrentLastResultDataTable[0][0]%

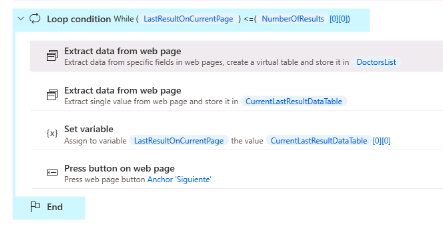


1. Add the action **Press button on web page** and expand the **UI Element** selector. Press **Add UI element**
2. Back on the web page, add the UI element for the button to continue to the next page (“Siguiente >”). Ensure that you are selecting the **Anchor** element of this button, as shown below:

A white rectangular object with a pink background

Description automatically generated

1. **Save** the **Press button on webpage** action
2. Drag and drop step 4 of your flow (Extract data from webpage and store into variable DoctorsList) into the **Loop condition** action, as shown below:



1. **Run** your flow. You don’t need to wait for the flow to scrape all results, just let it scrape a few pages and stop it afterwards.

|  |  |
| --- | --- |
| Lights On outline | **Pro tip**: Notice that every time a new search page loads, the scrape results stored on the **DoctorsList** variable are overwritten. We will prevent this on the next task by writing the results to an Excel sheet. |

### Task 5: Save results from web scraping to an Excel sheet

1. Search for the **Launch Excel** action and drag it to your process, above the **Launch new Microsoft Edge** step
2. Configure this action as indicated:

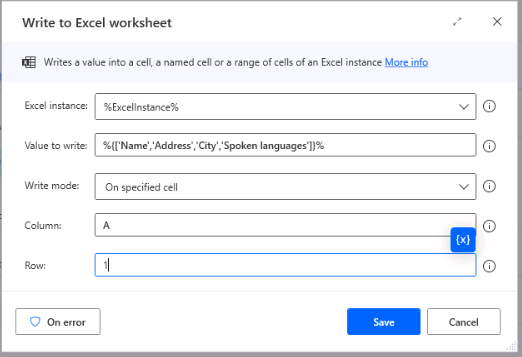
* **Launch Excel:** With a blank document
* **Make instance visible:** *Disable this toggle*
* **Variables produced:** %ExcelInstance%

A screenshot of a computer

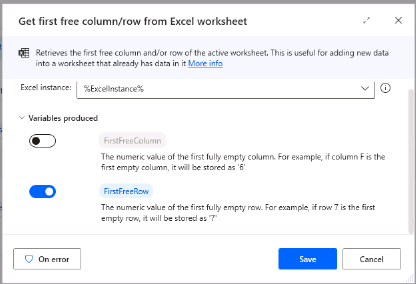
Description automatically generated

1. Drag and drop a **Write to Excel worksheet** action below the **Launch Excel** step and configure it as shown:

* **Excel instance:** %ExcelInstance%
* **Value to write:** %{[‘Name’,’Address’,’City’,’Spoken languages’]}%
* **Write mode:** On specified cell
* **Column:** A
* **Row:** 1

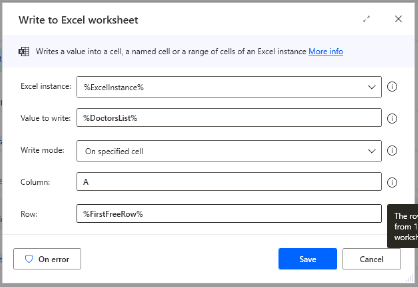


1. Search for the action **Get first free column/row from Excel worksheet** and drag it inside the **Loop condition**, after the **Extract data from webpage** step
2. Disable the variable produced on this step called **FirstFreeColumn** and click on **Save**



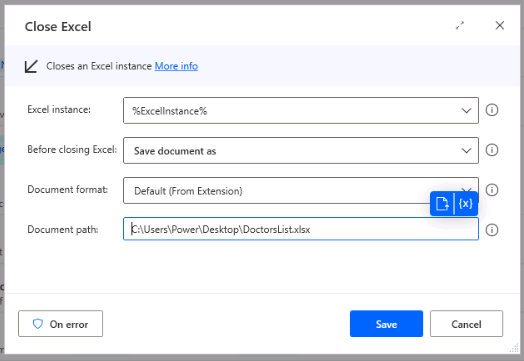
1. After this action, add a **Write to Excel worksheet** step and configure it as shown:

* **Excel instance:** %ExcelInstance%
* **Value to write:** %DoctorsList%
* **Write mode:** On specified cell
* **Column:** A
* **Row:** %FirstFreeRow%

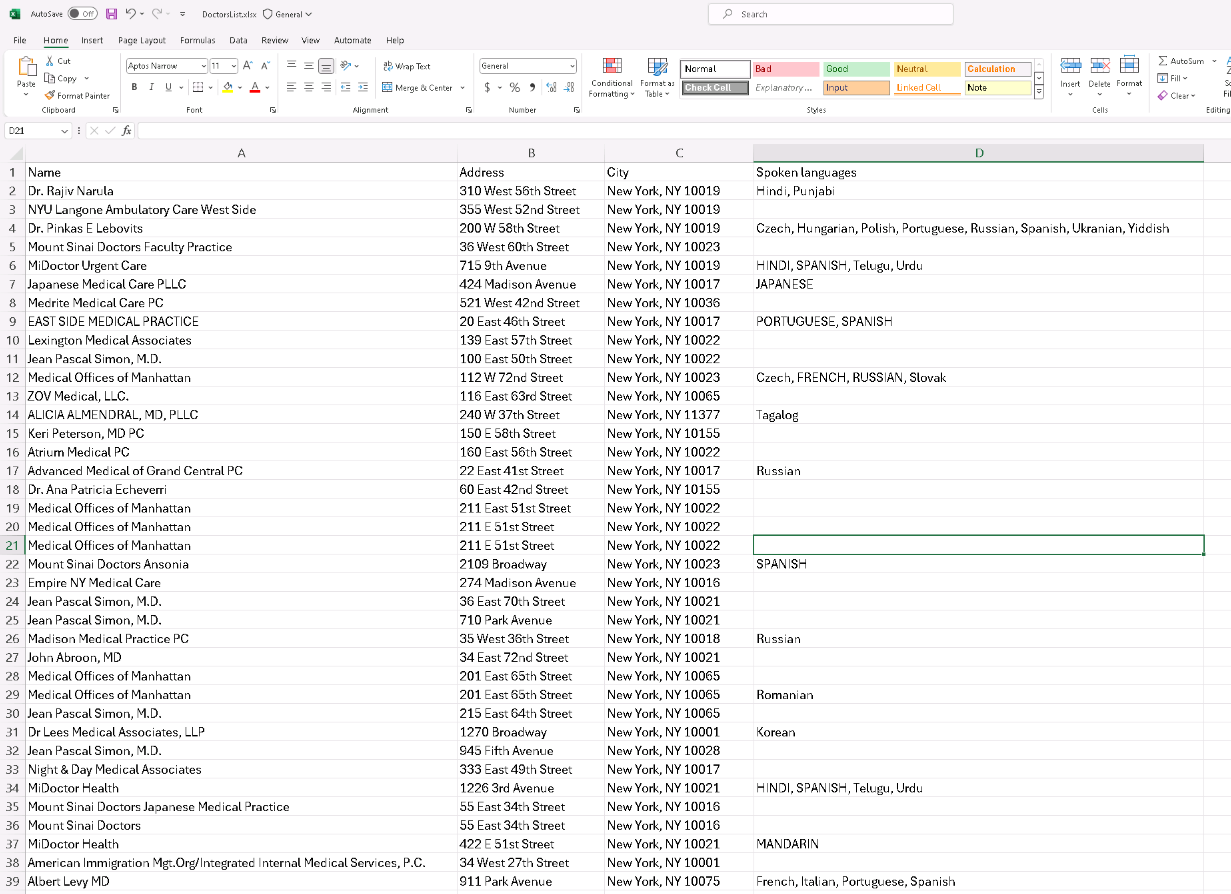


1. At the end of your flow, add a **Close Excel** action and configure it as shown:

* **Excel instance:** %ExcelInstance%
* **Before closing Excel:** Save document as
* **Document format:** Default (from extension)
* **Document path:** Type a document path for your new file, for example: C:\Users\YourUser\Desktop\DoctorsList.xlsx



1. **Save** your flow and **Run** it. If you don’t want to wait until the flow scrapes all results, replace your **Loop condition** to iterate until the variable %LastResultOnCurrentPage% is equal to a number, like 50.
2. **Open**  the Excel document and ensure that all the data you scraped from the website is on the worksheet.



## Terms of Use

By using this document, in whole or in part, you agree to the following terms:

### **Notice**

Information and views expressed in this document, including (without limitation) URL and other Internet Web site references, may change without notice. Examples depicted herein, if any, are provided for illustration only and are fictitious. No real association or connection is intended or should be inferred. This document does not provide you with any legal rights to any intellectual property in any Microsoft product.

### **Use Limitations**

Copying or reproduction, in whole or in part, of this document to any other server or location for further reproduction or redistribution is expressly prohibited. Microsoft provides you with this document for purposes of obtaining your suggestions, comments, input, ideas, or know-how, in any form, ("Feedback") and to provide you with a learning experience. You may use this document only to evaluate its content and provide feedback to Microsoft. You may not use this document for any other purpose. You may not modify, copy, distribute, transmit, display, perform, reproduce, publish, license, create derivative works from, transfer, or sell this document or any portion thereof. You may copy and use this document for your internal, reference purposes only.

### **Feedback**

If you give Microsoft any Feedback about this document or the subject matter herein (including, without limitation, any technology, features, functionality, and/or concepts), you give to Microsoft, without charge, the right to use, share, and freely commercialize Feedback in any way and for any purpose. You also give third parties, without charge, the right to use, or interface with, any Microsoft products or services that include the Feedback. You represent and warrant that you own or otherwise control all rights to such Feedback and that no such Feedback is subject to any third-party rights.

### **DISCLAIMERS**

CERTAIN SOFTWARE, TECHNOLOGY, PRODUCTS, FEATURES, AND FUNCTIONALITY (COLLECTIVELY "CONCEPTS"),

INCLUDING POTENTIAL NEW CONCEPTS, REFERENCED IN THIS DOCUMENT ARE IN A SIMULATED ENVIRONMENT

WITHOUT COMPLEX SET-UP OR INSTALLATION AND ARE INTENDED FOR FEEDBACK AND TRAINING PURPOSES

ONLY. THE CONCEPTS REPRESENTED IN THIS DOCUMENT MAY NOT REPRESENT FULL FEATURE CONCEPTS AND MAY

NOT WORK THE WAY A FINAL VERSION MAY WORK. MICROSOFT ALSO MAY NOT RELEASE A FINAL VERSION OF SUCH

CONCEPTS. YOUR EXPERIENCE WITH USING SUCH CONCEPTS IN A PHYSICAL ENVIRONMENT MAY ALSO BE DIFFERENT.

THIS DOCUMENT, AND THE CONCEPTS AND TRAINING PROVIDED HEREIN, IS PROVIDED “AS IS”, WITHOUT WARRANTY

OF ANY KIND, WHETHER EXPRESS, IMPLIED, OR STATUTORY, INCLUDING (WITHOUT LIMITATION) THE WARRANTIES OF

MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE, AND NONINFRINGEMENT. MICROSOFT DOES NOT

MAKE ANY ASSURANCES OR REPRESENTATIONS WITH REGARD TO THE ACCURACY OF THE RESULTS, THE OUTPUT THAT DERIVES FROM USE OF THIS DOCUMENT OR THE CONCEPTS, OR THE SUITABILITY OF THE CONCEPTS OR INFORMATION CONTAINED IN THIS DOCUMENT FOR ANY PURPOSE.