UiPath® | **Complete Exercise Walkthrough**

# Generate Yearly Report

## Contents

# 1. Overview

For this exercise, we will use the producer-consumer model. This means we will build two projects:

- A Dispatcher – using the REF without Queue Items design.
- A Performer – using the REF with Queue Items design.

We encourage you to use the two checklists and to identify reusable components.

# 2. The Dispatcher process

## 2.1. Prerequisites

The automation will need to identify if a page number is available on the Work Items page by using the transaction number. If a section is available, it will scrape it, filter it for WI4 type work items, and add a queue item for each resulting Datatable row. The TransactionItem will be of type Int32.

## 2.2. Configuration in UiPath Studio

### *2.2.1. Create a new project*

1. Create a new project using the Robotic Enterprise Framework template.
2. Set a proper name for the project.
3. Provide a proper description.

### *2.2.2. Whiteboard your workflows*

| Module Name | Description | Pre-condition | Post-action | Arguments |
|---|---|---|---|---|
| System1_Login | Login into Acme with the desired account | N/A | Browser opened to the required URL Logged in with the credentials provided | in_System1URL - String in_System1Credential - String |
| System1_Close | Close browser | Browser open to ACME page | Browser is closed | N/A |
| System1_NavigateTo_WI | Navigate to the Work Items Page | Be logged into the ACME website | Navigated successfully to the desired URL | in_URL - String |
| System1_ScrapeDataTable | Extract data from the Work Items DataTable | Data exists and the Work Items Page is opened | All Work Items are extracted as a DataTable | out_DataTable - DataTable |

| Module Name | Description | Pre-condition | Post-action | Arguments |
|---|---|---|---|---|
| System1_FilterWIDatatable | Filters the input DataTable for rows with the desired Work Item | DataTable exists | The filtered datatable is passed to the PopulateQueue workflow. | in_Type - String<br>in_DataTable - DataTable<br>out_FilteredDataTable - DataTable<br>in_Status - String |
| SendEmail | Sends Exception Email using Outlook and the Input Arguments | None | None | in_To - String<br>in_Subject - String<br>in_Body - String |

## *2.2.3. Develop your workflows*

1. Create two new folders in the Project folder. Name them as System1 and Common.

### 2.2.3.1. System1_Login.xaml

1. Inside the System1 folder, create a new Sequence type workflow called System1_Login.
2. Provide a description to the workflow.
3. Open the Arguments panel and create two in arguments of type String:
   a. in_System1URL
   b. in_System1Credential
4. Add a Log message activity to mark the start of the workflow execution.
5. Add a Get Credential activity. Configure the following properties:
   a. Asset Name – Provide the in_System1Credential argument.
   b. Password – Create a new variable called Password of type SecureString.
   c. Username – Create a new variable called Username of type String.
6. Open System 1 in Edge Browser.
7. Add an Use Application/Browser in the System1_Login workflow and indicate the opened browser and replace the URL with the argument in_System1URL.
8. Set the properties as below:
   a. Close: Never
   b. Open: IfNotOpen
   c. Resize Window: None
9. Inside the Do Container of the Use Application/Browser activity add:
   a. A Type Into activity where you indicate the email field in the browser and provide the Username variable for the Text.
   b. A Type Into activity where you indicate the password field in the browser and provide the Password variable for the Secure text.
   c. A Click activity where you indicate the Login button in the browser as target.
10. Log into ACME in your browser.
11. After the Use Application/Browser activity, add a Check App State activity.
    a. Indicate the Dashboard header in the browser as a target.
    b. Create a new Boolean variable LogInSuccessful. And set to true in “Target appears” section and to false in “Target does not appears” section
12. Add an If activity.
    a. Set the condition to Not LogInSuccessful.

13. Go to ACME, return to the login page, try to log in with the wrong credentials.
14. Return to Studio. Inside the Then block of the If activity:
    a. Add a Throw activity with the Exception property set to: new Exception("Incorrect Credentials supplied to System1")
    b. Note that later during development, we will invoke a workflow here to send an email to the process owner.
15. After the If activity, add a Log Message to mark the end of the workflow.

#### 2.2.3.2. System1_Close.xaml

1. Make sure you are logged into ACME System1 in your browser.
2. In Studio, add a Log Message activity to mark the start of the workflow.
3. Add an Use Application/Browser activity and indicate the browser logged into ACME.
4. Set properties as below:
    a. Close: Always
5. Inside the Do container of the Use Application/Browser activity:
    a. Add a Click activity and indicate the Log Out button.
6. Add a Log Message activity to mark the end of the workflow.

#### 2.2.3.3. System1_ NavigateTo_WI.xaml

1. Open the Arguments tab and create a new in argument of String type called in_URL.
2. Add a Log Message activity to mark the start of the workflow.
3. Make sure the ACME System 1 Dashboard page is open in your browser.
4. In Studio, add an Use Application/Browser and indicate the ACME System 1 page.
5. Inside the Do Container of the Use Application/Browser activity, add a Go To URL activity and provide the in_URL argument for the URL property.
6. Add a Log Message activity to mark the end of the workflow.

#### 2.2.3.4. System1_ ScrapeDataTable.xaml

1. Open the Arguments panel and create a new out argument of DataTable type called out_DataTable.
2. In your browser, navigate to the Work Items page on ACME System1.
3. In Studio, use the Data Scraping wizard to extract the work items data. Configure its properties as follows:
    a. For the DataTable property, enter out_DataTable.
    b. Set Number of items to 0
4. Add a Log Message to note the number of rows extracted.
    a. For the Message property enter: "Found " + out_Datatable.Rows.Count.ToString + " elements in the datatable"
5. Add a Log Message activity to mark the end of the workflow.

#### 2.2.3.5. System1_ FilterWIDataTable.xaml

1. Open the Arguments panel and create four new arguments:
    a. in_Type of type String
    b. in_DataTable of type DataTable
    c. out_FilteredDataTable of type DataTable
    d. in_Status of type String
2. Add a Log message activity to mark the start of the workflow execution.
3. Add a Filter Data Table activity. Configure it as follows:
    a. Open the Filter Wizard, set the FilterRowsMode to Keep and set:
        i. “Type” = in_Type
        ii. “Status” = in_Status
    b. Close the Filter Wizard.
    c. Set the Input DataTable property to in_DataTable.
    d. Set the Output DataTable property to out_FilteredDataTable.
4. Add a Log Message activity to mark the end of the workflow.

#### 2.2.3.6. SendEmail.xaml

1. Inside the Common folder, create a new workflow file called SendEmail.xaml.
2. Open the Arguments panel and create three in arguments of type String.
    a. in_To
    b. in_Subject
    c. in_Body
3. Add a Log Message activity to mark the start of the workflow.
4. Add a Send Outlook Mail Message activity with the following properties:
    a. To = in_To
    b. Subject = in_Subject
    c. Body = in_Body
5. Add a Log Message activity to mark the end of the workflow.
6. Go to System1_Login.xaml.
7. Invoke SendEmail.xaml as the first activity inside the Then block of the If activity.
8. Click Import Arguments and map the following values:
    a. in_To = “exceptions@acme-test.com”
    b. in_Subject = "Incorrect Credentials"
    c. in_Body = "Hello,"+Environment.NewLine+"The email or the password is incorrect. Please check and restart"+Environment.NewLine+"Thank you"

### *2.2.4. Edit the Configuration file*

1. Provide the following values in the Settings sheet:

| Name | Value |
|---|---|
| OrchestratorQueueName | Acme4Queue |
| logF_BusinessProcessName | ACME4_Dispatcher |

| | |
|---|---|
| System1_URL | https://acme-test.uipath.com |
| System1_Credential | System1_Credential |
| System1_WorkItemsURL | https://acme-test.uipath.com/work-items |

2. In the Constants sheet, set the MaxRetryNumber to 2.
3. In Orchestrator, create a Credential type Asset for ACME System 1 with the name System1_Credential.

## 2.3. Change the TransactionItem data type

### *2.3.1. In Main.xaml*

1. Locate the TransactionItem variable and change its type to Int32.

### *2.3.2. In the Get Transaction Data State*

1. Find the 'New Transaction' Statement under the Get Transaction Data and change the Condition for New Transaction to TransactionItem > 0 and the Condition for No Data to TransactionItem = 0.
2. Configure the End Process (Stop process requested) to:
   a. To: TransactionItem
   b. Value: 0
3. Select the Invoke GetTransactionData workflow activity. Open the Arguments list and change the data type for the out_TransactionItem argument from QueueItem to Int32.
4. Expand the Exception section in the Try GetTransactionData section and locate the End Process (Could not get new transaction) activity.
5. Configure it to:
   a. To: TransactionItem
   b. Value: 0

### *2.3.3. In the Process Transaction State*

1. Locate the Invoke Process workflow activity and open the Arguments list; change the data type for the in_TransactionItem argument from QueueItem to Int32.
2. Locate the Invoke SetTransactionStatus workflow activities (including the Business Exception and System Exception scenarios) and open the Arguments list; locate the entry for the in_TransactionItem argument and change the value (not the data type!) from TransactionItem to Nothing.

## 2.4. Applications Used: open/close/kill

### *2.4.1. Edit the InitiAllApplications.xaml workflow*

1. Add a Log Message activity to mark the start of the workflow.
2. Invoke the System1\System1_Login.xaml workflow.
   a. Click Import Arguments and make the following changes:

| Name | Direction | Type | Value |
|---|---|---|---|
| in_System1URL | In | String | in_Config("System1_URL").ToString |
| in_System1Credential | In | String | in_Config("System1_Credential").ToString |

3. Invoke the System1\System1_NavigateTo_WI.xaml workflow.
   a. Click Import Arguments and make the following changes:

| Name | Direction | Type | Value |
|---|---|---|---|
| in_URL | In | String | in_Config("System1_WorkItemsURL").ToString |

4. Add a Log Message activity to mark the end of the workflow.

### *2.4.2. Edit the Framework/CloseAllApplication.xaml workflow*

1. Add a Log Message activity to mark the start of the workflow.
2. Invoke the System1\System1_Close.xaml workflow.
3. Add a Log Message activity to mark the end of the workflow.

### *2.4.3. Edit the Framework/KillAllProcesses.xaml workflow*

1. Add a Log Message activity to mark the start of the workflow.
2. Add a Kill Process activity.
   a. For the ProcessName property, enter the process name for your browser (for example, “msedge” for Edge)
3. Add a Log Message activity to mark the end of the workflow.

## 2.5. Business Process: Transaction Data and Process

### *2.5.1. Edit GetTransactionData.xaml workflow*

1. Set the type of the out_TransactionItem argument to Int32.
2. Delete the Retry Get transaction item activity.
3. At the top of the Get Transaction Data sequence, add a Check App State activity.

   a. Indicate the Next Page Button on the Work Items page.
   b. Create a dynamic selector for the Strict selector with the xml code:
      <html title='ACME System 1 - Work Items' />
      <webctrl tag='A' aaname='{{in_TransactionNumber}}' parentclass='page-numbers' />
   c. Create a new variable of Boolean type called PageExists. Set to True in "Target Appear" section and to False in Target does not appear section.
4. Add an If activity and set the Condition to PageExists.
   a. In the Then block, add an Assign Activity:
      i. To: out_TransactionItem.
      ii. Value: in_TransactionNumber.
   b. In the Else branch, add another Assign activity:
      i. To: out_TransactionItem.
      ii. Value: 0.
5. Set the Condition of the If a new transaction item is retrieved, get additional information about it activity to out_TransactionItem > 0.

### *2.5.2. Edit Process.xaml workflow*

1. Change the data type for the in_TransactionItem argument from QueueItem to Int32.
2. Create three new variables:
   a. CurrentPageAlreadyOpen of type Boolean
   b. WorkItems of type DataTable
   c. FilteredWorkItems of type DataTable
3. Add a Check App State activity:
   a. Indicate the first page button in the Work Items page.
   b. Set the Selector to:
      <html title='ACME System 1 - Work Items' />
      <webctrl aaname='{{in_TransactionItem}}' tag='A' class='page-numbers current' />
   c. Create a Boolean variable CurrentPageAlreadyOpen and set it to true in "Target appears" section and set to false "Target does not appear".
4. Add an If Activity.
   a. Set the Condition to CurrentPageAlreadyOpen.
   b. Inside the Then branch:
      i. Add a Log Message activity with the Message: "Already on the correct page for transaction " + in_TransactionItem.ToString
   c. Inside the Else branch:
      i. Add a Use Application/Browser and indicate the Work Items Page.
      ii. Add a Click activity, indicate the first page button in the Work Items page and set the Selector to:
         <html title='ACME System 1 - Work Items' />
         <webctrl aaname='{{in_TransactionItem}}' tag='A' parentclass='page-numbers' />
5. Invoke the System1\ System1_ScrapeDataTable.xaml
   a. Click Import Arguments and make the following changes:

| **Name** | **Direction** | **Type** | **Value** |
|---|---|---|---|
| out_DataTable | Out | DataTable | WorkItems |

6. Invoke the System1\System1_FilterWIDatatable.xaml
    a. Click Import Arguments and make the following changes:

| **Name** | **Direction** | **Type** | **Value** |
|---|---|---|---|
| in_Type | In | String | “WI4” |
| in_Datatable | In | DataTable | WorkItems |
| out_FilteredDataTable | Out | DataTable | FilteredWorkItems |
| in_Status | In | String | “Open” |

7. Add an If activity and set the condition to FilteredWorkItems.Rows.Count > 0
    a. Inside the Then branch of the If activity, add a Bulk Add Queue Items activity. Configure its properties to:
        i. Data Table: FilteredWorkItems
        ii. Queue Name: in_Config("OrchestratorQueueName").ToString

# 3. The Performer process

## 3.1. Prerequisites

Make sure the queue indicated in the previous section is available in Orchestrator. Set the Max # of retries value to 2.

## 3.2. Configuration in UiPath Studio

### *3.2.1. Create a new project*

1. Create a new project using the Robotic Enterprise Framework template.
2. Set a proper name for the project.
3. Provide a proper description.

### *3.2.2. Whiteboard your workflows*

| **Module Name** | **Description** | **Pre-condition** | **Post-action** | **Arguments** |
|---|---|---|---|---|
| System1_Login | same as in Dispatcher | | | |
| System1_Close | same as in Dispatcher | | | |
| System1_NavigateTo | Navigate to the ACME System 1 page using the in_URL argument | Logged into the ACME website | Navigated successfully to the desired URL | in_URL - String |

| Module Name | Description | Pre-condition | Post-action | Arguments |
|---|---|---|---|---|
| System1_GetClientDetails | Get the Client Information: Tax ID from the Work Items Details Page | Logged in and the Work Items Details Page is opened | Tax ID is obtained from the Client Information | out_TaxID - String |
| System1_DownloadMonthlyReports | Enter tax ID, enter year and for each month download report if exists (treat exceptions) | Logged in and on the Download Monthly Report Page, report directory exists | Downloaded all available reports for a given TaxID and a given year into a given location | in_TaxID - String<br>in_Year - String<br>in_ReportDirPath - String |
| System1_MergeMonthlyReports | Merge the downloaded csv files from a given TaxID and a year, located into a certain folder, into one Excel file - yearly report | Monthly reports exist in the certain folder | Constructed the merged excel file and save it with name format "Yearly-Report-[year]-[taxid].xlsx" | in_TaxID - String<br>in_Year - String<br>in_ReportDirPath - String<br>out_YearlyReportPath - String |
| System1_UploadYearlyReport | Upload a given yearly report to the Upload Yearly Report Page for a given Tax ID and a given year | Yearly report exists in the specified folder and the Update Yearly Report Page is opened | Report successfully uploaded and the Confirmation ID is retrieved | in_TaxID - String<br>in_Year - String<br>in_YearlyReportPath - String<br>out_Confimation - String |
| System1_UpdateWorkItems | Add hash into the comment section and change status to "Completed" | Update Work Item Page is opened | Work Item is update with the hash and the status along with closing the popup. | in_Comment - String<br>in_Status - String |
| SendEmail | Sends Exception Email using Outlook and the Input Arguments | None | None | in_To - String<br>in_Subject - String<br>in_Body - String |

### *3.2.3. Develop your workflows*

1. Create two new folders in the Project folder. Name them as System1 and Common.

### 3.2.3.1. System1_Login.xaml

Note: We recommend saving the System1_Login.xaml workflow from the Dispatcher project as a component and reusing it here. Below are the steps to develop the workflow.

1. Inside the System1 folder, create a new Sequence type workflow called System1Login.
2. Provide a description of the workflow.
3. Open the Arguments panel and create two in arguments of type String:
    a. in_System1URL
    b. in_System1Credential
4. Add a Log message activity to mark the start of the workflow execution.
5. Add a Get Credential activity. Configure the following properties:
    a. Asset Name – Provide the in_System1Credential argument.
    b. Password – Create a new variable called Password of type SecureString.
    c. Username – Create a new variable called Username of type String.
6. Open System1 in Edge Browser.
7. Add an Use Application/Browser in the System1_Login workflow and indicate the opened browser and replace the URL with the argument in_System1URL.
8. Set the properties as below:
    a. Close: Never
    b. Open: IfNotOpen
9. Inside the Do Container of the Use Application/Browser activity add:
    a. A Type Into activity where you indicate the email field in the browser and provide the Username variable for the Text.
    b. A Type Into activity where you indicate the password field in the browser and provide the Password variable for the Secure text.
    c. A Click activity where you indicate the Login button in the browser as target.
10. Log into ACME in your browser.
11. After the Use Application/Browser activity, add a Check App State activity.
    a. Indicate the Dashboard header in the browser as a target.
    b. Create a new Boolean variable LogInSuccessful. And set to true in “Target appears” section and to false in “Target does not appears” section
12. Add an If activity.
    a. Set the condition to Not LogInSuccessful.
13. Go to ACME, return to the login page, try to log in with the wrong credentials.
14. Return to Studio. Inside the Then block of the If activity:
    a. Add a Throw activity with the Exception property set to: new Exception("Could not login successfully")
    b. Note that later during development, we will invoke a workflow here to send an email to the process owner.
15. After the If activity, add a Log Message to mark the end of the workflow.

### 3.2.3.2. System1_Close.xaml

Note: We recommend saving the System1_Close.xaml workflow from the Dispatcher project as a component and reusing it here. Below are the steps to develop the workflow.

1. Make sure you are logged into ACME System1 in your browser.
2. In Studio, add a Log Message activity to mark the start of the workflow.
3. Add an Use Application/Browser activity and indicate the browser logged into ACME.
4. Set properties as below:
    a. Close: Always
5. Inside the Do container of the Use Application/Browser activity:
    a. Add a Click activity and indicate the Log Out button.
6. Add a Log Message activity to mark the end of the workflow.

### 3.2.3.3. System1_NavigateTo.xaml

Note: We recommend saving the System1_NavigateTo.xaml workflow from the Dispatcher project as a component and reusing it here. Below are the steps to develop the workflow.

1. Open the Arguments tab and create a new in argument of String type called in_URL.
2. Add a Log Message activity to mark the start of the workflow.
3. Make sure the ACME System 1 Dashboard page is open in your browser.
4. In Studio, add an Use Application/Browser activity and indicate the ACME System 1 page.
5. Inside the Do Container of the Use Application/Browser activity, add a Go to URL activity and provide the in_URL argument for the URL property.
6. Add a Log Message activity to mark the end of the workflow.

### 3.2.3.4. System1_GetClientDetails.xaml

1. Add a Log Message activity to signify the start of the workflow.
2. Open the Arguments panel and create a new out argument of String type called out_TaxID.
3. Navigate to the Work Item Details page in your web browser.
4. In Studio, add a Use Application/Browser activity and indicate the Work Item Details page.
5. Inside the Do container of the Use Application/Browser activity, add a Get Text activity, indicate the Client Information Details text box of a WI4 details page and configure the following properties:
    a. Selector - "<webctrl tag='P' idx='1' />"
    b. Save to – Create a new String type variable called ClientInformation
6. After the Use Application/Browser activity, add an Assign activity with the following:
    a. out_TaxID = ClientInformation.Substring(ClientInformation.IndexOf("Tax ID: ") + "Tax ID: ".Length).Split(Environment.NewLine.ToCharArray)(0)
7. Add a Log Message activity to note value of each out argument and mark the end of the workflow.

### 3.2.3.5. System1_DownloadMonthlyReports.xaml

1. Open the Arguments panel and create three in arguments of type String:
   a. in_TaxID
   b. in_Year
   c. in_ReportDirPath
2. Add a Log message activity to mark the start of the workflow execution.
3. In your browser, navigate to the Reports – Download Monthly Report page in ACME System 1.
4. In Studio, add an Use Application/Browser activity and indicate the Download Monthly Report page.
5. Include all following activities inside the Do container.
6. Add a Type Into activity.
   a. Indicate the Vendor TaxID field.
   b. Provide the in_TaxID for the Text property.
7. Add a Click activity.
   a. Indicate a the first year  in the Year drop down.
   b. Edit the Selector to provide the following string:  <webctrl aaname='{{in_Year}}' parentid='searchForm' tag='SPAN' />
8. Add an Assign activity.
   a. In the To field, create a new variable of type String[] called Months.
   b. Import the System.Globalization namespace to enable the usage of DateTimeFormatInfo Class
   c. In the Value field, add the expression: DateTimeFormatInfo.CurrentInfo.MonthNames
9. Add a For Each activity.
   a. Name the iterator “month”.
   b. Provide Months in the List of items property.
   c. Change the TypeArgument to String.
10. Inside the Body of the For Each activity add an If activity.
    a. Name it If not an empty month
    b. Provide the condition: not String.IsNullOrEmpty(month)
11. Inside the Then Block of the If activity add a Click Activity.
    a. Indicate January in the Month drop down menu.
    b. For the selector, provide: "<webctrl aaname='" + month +"' parentid='searchForm' tag='A' />"
12. Add a Click activity and indicate the Download Report button.
13. In your browser, trigger a Report Download in ACME.
14. In Studio, depending on your browser and settings, handle the potential cases encountered during download using If and Check App State activities:
    a. Incomplete information provided
    b. The Report does not exist
    c. Report Found
15. Implement the logic to store the downloaded report to the file path provided in the in_ReportDirPath argument.
16. Add a Log Message activity to mark the end of the workflow.

### 3.2.3.6. System1_MergeMonthlyReports.xaml

1. Open the Arguments panel and create four arguments of type string:
   a. in_TaxID
   b. in_Year

        c. in_ReportDir
        d. out_YearlyReportPath
2. Add a Log message activity to mark the start of the workflow execution.
3. Add an Assign activity:
    a. To: Create a new variable of type String[] called ReportsByTaxID and add it to the field
    b. Value: Directory.GetFiles(in_ReportDir, "Report-"+in_TaxID+"*.csv")
4. Add a For Each activity:
    a. Set the iterator name to "file"
    b. Set the List of items property to ReportsByTaxID
    c. Set the TypeArgument to String
5. Inside the body of the For Each activity:
    a. Add a Read CSV activity
        i. Set the FilePath to file
        ii. Set the Output to a new variable of DataTable type called MonthlyReportDt
        iii. Make sure the Has headers option is checked
    b. Add a Merge Data Table activity
        i. Set the Destination to a new variable of DataTable type called YearlyReportsDt. From the Variables tab, have it initialized by entering New DataTable in the Default value.
        ii. Set the Source to MonthlyReportDt
    c. Add a Delete activity and provide file for the Path property
6. After the For Each activity, add an Assign activity:
    a. To: out_YearlyReportPath
    b. Value: Path.Combine(in_ReportDir, "Yearly-Report-"+ in_Year + "-" + in_TaxID + ".xlsx")
7. Add a Write Range Workbook activity
    a. For the DataTable property enter YearlyReportsDt
    b. For the Workbook path enter out_YearlyReportPath
    c. Enter a SheetName
    d. Make sure the AddHeaders option is enabled
8. Add a Log Message activity to mark the end of the workflow.

### 3.2.3.7. System1_UploadYearlyReport.xaml

1. Open the Arguments panel and create four arguments of type string:
    a. in_TaxID
    b. in_Year
    c. in_YearlyReportPath
    d. out_Confirmation
2. Add a Log message activity to mark the start of the workflow execution.
3. In your browser, navigate to the Reports – Upload Yearly Report page in ACME
4. In Studio, add an Use Application/Browser activity and indicate the Reports – Upload Yearly Report page in ACME.
5. Until further notice, all the following activities should be added inside the Do Container of the Use Application/Browser activity.
6. Add a Type Into activity and indicate the Vendor TaxID field.
    a. Provide the in_TaxID argument for the Text.
7. Add a Click activity and indicate the first year in the Year drop down list.

    a. For the Selector, provide: <webctrl aaname='{{in_Year}}' parentid='searchForm' tag='SPAN' />
8. Add a Click activity and indicate the Select Report File button.
9. Click the Select Report File button in your browser window.
10. In Studio, add a Type Into activity and indicate the File name input field.
    a. For the Text enter in_YearlyReportPath + "[k(enter)]"
11. Add a Click activity and indicate the Upload Report button.
12. In your browser, make a successful upload and leave the resulting pop-up window open.
13. In Studio, add a Check App state activity and indicate the text box for the pop-up window.
    a. For the Selector property, enter: <ctrl name='Report was uploaded - confirmation id is *' role='text' />
    b. Create a new Boolean variable called UploadedSuccessfully, set to True in “Target Appear” section and False in “Target does not appear” section.
14. Add an If activity and name it If uploaded successfully.
    a. Add the condition UploadedSuccessfully.
15. Inside the then branch of the If activity:
    a. Add a Get Text activity and indicate the Success message in the pop-up window.
        i. For the Selector property, enter: <ctrl name='Report was uploaded - confirmation id is *' role='text' />
        ii. For the Value property, create a new String variable called MessageText
    b. Add an Assign activity.
        i. To: Add out_Confirmation
        ii. Value: Add MessageText.Split(" "c).Last.ToString.Trim
    c. Add a Click activity and indicate the OK button.
    d. Add an Info level Log Message activity.
        i. For the Message property, enter "Confirmation id is : " + out_Confirmation.
16. After the For Each activity, add a Log Message activity to mark the end of the workflow.

### 3.2.3.8. System1_UpdateWorkItems.xaml

1. Open the Arguments panel and create two in arguments of type string:
    a. in_Comment
    b. in_Status
2. Add a Log message activity to mark the start of the workflow execution.
3. In your browser, navigate to the Work Items page, open a WI4 work item and click Update Work Item. Leave the resulting window open.
4. In Studio, add an Use Application/Broswer activity and indicate the Update Work Item window.
5. Until further notice, all following activities should be added in the Do container of the Use Application/Browser activity.
6. Add a Type Into activity and indicate the Add Comments input field.
    a. For the Text, provide in_Comment
7. Add a Click activity and indicate the first status in the New Status dropdown menu.

a. For the Selector property, enter "<webctrl aaname='{{in_Status}}' tag='SPAN' />"

8. Add a Click activity and indicate the Update Work Item button.
9. In your browser, make a successful update. Leave the resulting pop-up window open.
10. In Studio, add a Check App State activity and indicate the OK button in the pop-up window.
    a. Create a new Boolean type variable called UpdatedSuccessfully, set to True in “Target Appear” section and False in “Target does not appear” section.
11. Add an If activity and provide the Condition UpdatedSuccessfully.
12. Inside the Then branch of the If activity:
    a. Add an Info level Log Message with the Message: "Uploaded with ID “ + in_Comment
    b. Add a Click activity and indicate the OK button.
13. Inside the Else branch of the If activity:
    a. Add an Error level Log Message with the Message: "Could not update work item"
    b. Add a Throw activity with the Exception: New BusinessRuleException("Work Item was not not update successfully")
14. After the Use Application/Browser activity, add a Log Message activity to mark the end of the workflow.

#### 3.2.3.9. SendEmail.xaml

Note: We recommend saving the SendEmail.xaml workflow from the Dispatcher project as a component and reusing it here. Below are the steps to develop the workflow.

1. Inside the Common folder, create a new workflow file called SendEmail.xaml.
2. Open the Arguments panel and create three in arguments of type String.
    a. in_To
    b. in_Subject
    c. in_Body
3. Add a Log Message activity to mark the start of the workflow.
4. Add a Send Outlook Mail Message activity with the following properties:
    a. To = in_To
    b. Subject = in_Subject
    c. Body = in_Body
5. Add a Log Message activity to mark the end of the workflow.
6. Go to System1_Login.xaml.
7. Invoke SendEmail.xaml as the first activity inside the Then block of the If activity.
8. Click Import Arguments and map the following values:
    a. in_To = “exceptions@acme-test.com”
    b. in_Subject = "Incorrect Credentials"
    c. in_Body = "Hello,"+Environment.NewLine+"The email or the password is incorrect. Please check and restart"+Environment.NewLine+"Thank you"

### *3.2.4. Edit the Configuration file*

1. Provide the following values in the Settings sheet:

| Name | Value |
|---|---|
| OrchestratorQueueName | Acme4Queue |
| logF_BusinessProcessName | ACME4_Performer |
| ReportDirectory | C:\Users\lavinia.nastase\Desktop\Reports |
| System1_Credential | System1_Credential |
| System1_URL | https://acme-test.uipath.com |
| AcmeWorkItemsURL | https://acme-test.uipath.com/work-items |
| AcmeDownloadMonthlyReportsURL | https://acme-test.uipath.com/reports/download |
| AcmeUploadYearlyReportURL | https://acme-test.uipath.com/reports/upload |
| AcmeUpdateURL | https://acme-test.uipath.com/work-items/update |
| Status | Completed |

2. In the Constants sheet, make sure the MaxRetryNumber to 0.
3. In Orchestrator, make sure the Credential type Asset for ACME System 1 with the name System1_Credential is available.
4. Save and close the Configuration file

## 3.3. Applications Used: open/close/kill

### *3.3.1. Edit the InitiAllApplications.xaml workflow*

1. Add a Log Message activity to mark the start of the workflow.
2. Invoke the System1\System1_Login.xaml workflow.
    a. Click Import Arguments and make the following changes:

| Name | Direction | Type | Value |
|---|---|---|---|
| in_System1URL | In | String | in_Config("System1_URL").ToString |
| in_System1Credential | In | String | in_Config("System1_Credential").ToString |

3. Add a Log Message activity to mark the end of the workflow.

### *3.3.2. Edit the Framework/CloseAllApplication.xaml workflow*

1. Add a Log Message activity to mark the start of the workflow.
2. Invoke the System1\System1_Close.xaml workflow.
3. Add a Log Message activity to mark the end of the workflow.

### *3.3.3. Edit the Framework/KillAllProcesses.xaml workflow*

1. Add a Log Message activity to mark the start of the workflow.
2. Add a Kill Process activity.
    a. For the ProcessName property, enter the process name for your browser (for example, “msedge” for Edge).
3. Add a Log Message activity to mark the end of the workflow.

## 3.4. Business Process: Transaction Data and Process

### *3.4.1. Edit the GetTransactionData.xaml workflow*

No changes are needed in this workflow.

### *3.4.2. Edit the Process.xaml workflow*

1. Add an Assign activity:
    a. To: Create a new variable of type String called WIID
    b. Value: Enter in_TransactionItem.SpecificContent("WIID").ToString
2. Add an Info level Log Message activity with the Message: “Processing “ + WIID.
3. Invoke the System1\System1_NavigateTo.xaml workflow.
    a. Click Import Arguments and make the following changes:

| **Name** | **Direction** | **Type** | **Value** |
|---|---|---|---|
| in_URL | In | String | in_Config("AcmeWorkItemsURL").ToString + "/" + WIID |

4. Create a new variable of type String called TaxID
5. Invoke the System1\System1_GetClientDetails.xaml workflow.
    a. Click Import arguments and make the following changes:

| **Name** | **Direction** | **Type** | **Value** |
|---|---|---|---|
| out_TaxID | Out | String | TaxID |

6. Invoke the System1\System1_NavigateTo.xaml workflow.
    a. Click Import Arguments and make the following changes:

| **Name** | **Direction** | **Type** | **Value** |
|---|---|---|---|
| in_URL | In | String | in_Config("AcmeDownloadMonthlyReportsURL").ToString |

7. Invoke the System1\ System1_DownloadMonthlyReports.xaml workflow.
    a. Click Import Arguments and make the following changes:

| **Name** | **Direction** | **Type** | **Value** |
|---|---|---|---|
| in_TaxID | In | String | TaxID |
| in_Year | In | String | “2022” |
| in_ReportDirPath | In | String | in_Config("ReportDirectory").ToString |

8. Create a new variable of string type called YearlyReportPath.

9. Invoke the System1\System1_MergeMonthlyReports.xaml workflow.
    a. Click Import Arguments and make the following changes:

| Name | Direction | Type | Value |
|---|---|---|---|
| in_TaxID | In | String | TaxID |
| in_Year | In | String | “2022” |
| in_ReportDir | In | String | in_Config("ReportDirectory").ToString |
| out_YearlyReportPath | Out | String | YearlyReportPath |

10. Invoke the System1\System1_NavigateTo.xaml workflow.
    a. Click Import Arguments and make the following changes:

| Name | Direction | Type | Value |
|---|---|---|---|
| in_URL | In | String | in_Config("AcmeUploadYearlyReportURL").ToString |

11. Create a new variable of String type called Confirmation.
12. Invoke the System1\System1_UploadYearlyReport.xaml workflow.
    a. Click Import Arguments and make the following changes:

| Name | Direction | Type | Value |
|---|---|---|---|
| in_TaxID | In | String | TaxID |
| in_Year | In | String | “2022” |
| in_YearlyReportPath | In | String | YearlyReportPath |
| out_YearlyReportPath | Out | String | Confirmation |

13. Invoke the System1\System1_NavigateTo.xaml workflow.
    a. Click Import Arguments and make the following changes:

| Name | Direction | Type | Value |
|---|---|---|---|
| in_URL | In | String | in_Config("AcmeUpdateURL").ToString + "/" + WIID |

14. Invoke the System1\System1_UpdateWorkItems.xaml workflow.
    a. Click Import Arguments and make the following changes:

| Name | Direction | Type | Value |
|---|---|---|---|
| in_Comment | In | String | Confirmation |
| in_Status | In | String | in_Config("Status").ToString |

15. Invoke the System1\System1_NavigateTo.xaml workflow.
    a. Click Import Arguments and make the following changes:

| Name | Direction | Type | Value |
|---|---|---|---|
| in_URL | In | String | in_Config("AcmeWorkItemsURL").ToString |