Robotic Process Automation

Development Specifications Document (DSD)

*Process Name: BYO Recipe Scraper*

Table of Contents

[**Document Overview**](#_fmc2ik42b62t) **1**

[Automated Master Project Details](#_soa72miybokv) **1**

[Runtime Guide](#_e5eh7vtp3elw) **1**

[Runtime Diagram](#_2pt89uzbsm6q) 1

[List of Packages](#_8uc76jjm25ud) 2

[Master Project Runtime Details](#_33q1drg667x0) 2

[**Project Details**](#_q7qb3l8qz84p) **3**

[Project Name: <project name>](#_vrc3lxjwb5na) 3

[Workflow(s) specific to the Project](#_in5ehl2op8tm) 4

[Project Name: <project name>](#_eddmfv9qxvte) 4

[Workflow(s) specific to the Project](#_cebp8lzhrwzt) 5

[**Compliance Considerations and Reporting Requirements**](#_azdzmbnsrqr4) **5**

[**Other Details**](#_gmvdjkbe065o) **6**

[Future Improvements](#_3e7irmfl1h6l) 6

[Debugging Tips](#_qtg3tsjmu03s) 6

[Other Remarks](#_zgfonke2bma) 6

[**Post UAT Specifications**](#_qba241jo7cu2) **6**

[**Glossary**](#_go2cr78yd0pl) **6**

Version Control

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Date | Version | Role | Name | Organization Department | Function | Comments |
| 08-Feb-2021 | 1.0 | Author | Jes Hunsballe | N/A | N/A |  |
| 13-Feb-2021 | 1.1 | Author | Jes Hunsballe | N/A | N/A | Revised runtime diagram |
| 14-Feb-2021 | 1.2 | Author | Jes Hunsballe | N/A | N/A | Changes to List of Packages, Master Project Runtime Details, Project Details |



# Document Overview

The Development Specifications Document (DSD) is created for every business process automated using RPA. The DSD needs to be reviewed and updated for every change requested and applied to the automated process. This document provides a technical snapshot and must always reflect the latest design and key features of the automated workflow.

The document naming convention will follow the naming convention and the version of the automated process. This can be “business process name version” or it can be defined, case by case, as part of the larger RPA project design.

This document is completed by the RPA Solution architect and RPA developer who automates the business process. It is reviewed by the business process owner, application owner, and CoE design authority.

This document is meant to assist the RPA COE, IT operations and process owners by providing a snapshot of the automated process details and components. It can also serve developers to have a quick glance at the setup, before diving into the code, to troubleshoot or update changes. The purpose of the document is to record the outcome specific to the automated master project and its subcomponents: projects, workflows, sequences etc.

# Master Project Details

Details filled in by the developer reflect the actual information for the master project released for production.

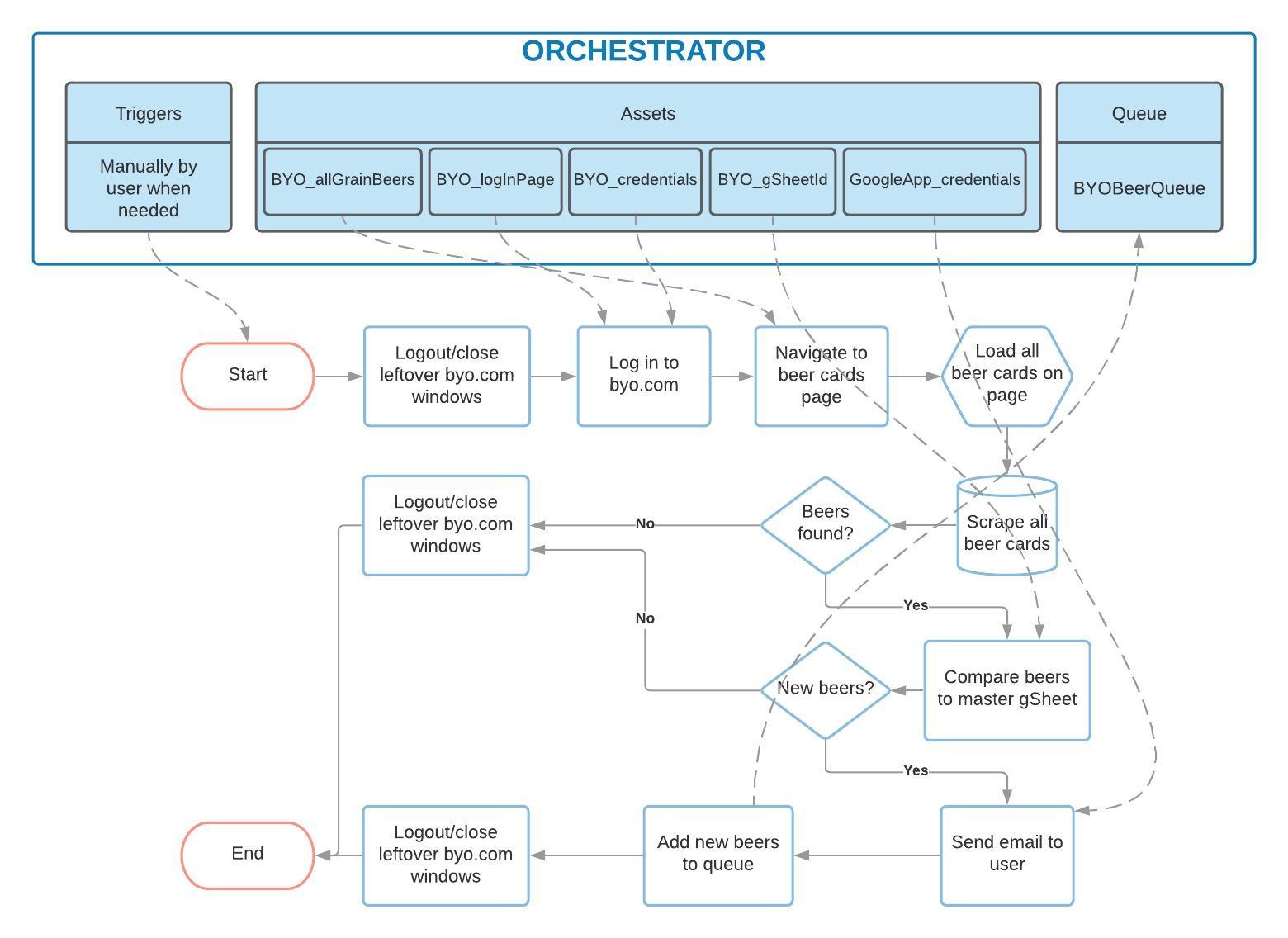
|  |  |  |
| --- | --- | --- |
| # | Item | Details  Fill in with free text. If not applicable, mark the filed as "N/A". No empty fields. |
| 1 | Master Project Name and Version | BYO Recipe Scraper v1.1 |
| 2 | Robot Type (attended/unattended/mix) | Attended |
| 3 | Is Orchestrator used? (Yes/No) | Yes |
| 4 | Scalable? (Yes/No)  Can the process be run by multiple robots in parallel? | Yes – the Performer part can be run by multiple robots |

# Runtime Guide

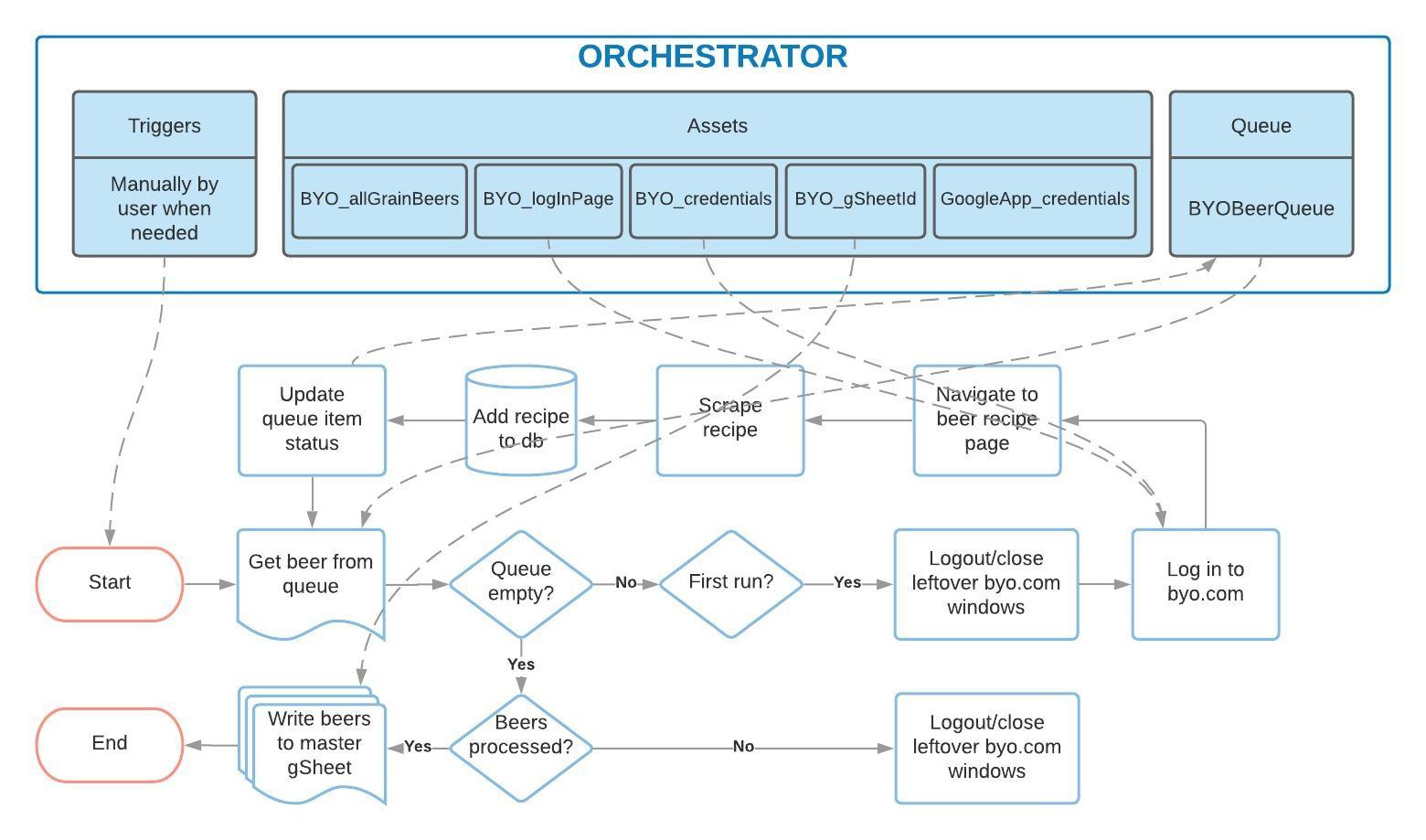
## Runtime Diagram

**Architectural Structure of the Master Project** Display the interaction between components (package / robots, Orchestrator queues, and running order).

***Dispatcher:***



***Performer:***



## List of Packages

Include **the list of packages and the high level description** for each of them, to explain each one's purpose:

|  |  |  |
| --- | --- | --- |
| # | Package Name | High-Level Description |
| 1 | Dispatcher.xaml | Dispatcher scrapes beer details from byo.com, compares them with a master gSheet and adds only new beers to an Orchestrator queue for subsequent processing. |
| 2 | Performer.xaml | Performer gets Orchestrator queue items to use for scraping full beer recipes from byo.com, then adds them to a master gSheet. |

\*Add more rows to the table to include all the project names and versions. No fields should be left empty. Use “N/A” for the items that don't apply to your project.

## Master Project Runtime Details

Details of the automated process:

|  |  |  |
| --- | --- | --- |
| # | Item | Details  (Fill in with free text. If the section does not apply to your automation, mark the field as “N/A”. No empty fields. ) |
| 1 | Production Environment Details | Machine Name: laptop-at64rvek\jeshu  Location: C:\Users\jeshu\OneDrive\Documents\Code\RPA nanodegree\BYORecipeScraper  OS: MS Windows 10  CPU: Intel Core i7  GPU: N/A  RAM: 8 GB  Browser: Chrome  UiPath: UiPath Studio |
| 2 | Prerequisites to run | Byo.com account  UiPath Web Automation plugin for Chrome  Chrome browser  UiPath Studio  Google account |
| 3 | Input Data | Master Google sheet with known beer details.  Beer card details from byo.com web page.  Beer recipes from byo.com beer web page.  Mail template html files in \Mail folder. |
| 4 | Expected Output (output data) | Beer details and recipe in a beer master Google Sheet (one beer per row).  Beer details in the output are: Beer name, beer url, beer description, beer style(s), beer recipe.  Email sent to user with new beers identified. |
| 5 | How to start the automated process? | Manually by user from user’s Windows PC |
| 6 | Resuming the process from a particular step | N/A |
| 7 | Reporting  queues reporting, Kibana or another platform | N/A |
| 8 | Manual Error Handling  roll back or manually complete failed transactions. Procedures to reset the item. Ex “set status as investigating” | Failed transactions are left unprocessed.  Status of queue item is set to ‘Failed’. |
| 1. How to resume the process in case of error | N/A |
| 1. How to manually fix transactions with error | Run the dispatcher again and it will identify unscraped beers from byo.com and add them to the queue again. |
| 9 | Use of Orchestrator |  |
| 1. Password Policies   specific compliance requests? | N/A |
| 1. Stored Credentials   Never hard code credentials in the workflow | 1. Byo.com credentials stored in asset ‘BYO\_credentials’. 2. Gmail app password and user name stored in asset ‘GoogleApp\_credentials’. 3. Access to gSheet with an API service account json key stored externally on local user drive. gSheet must be shared to the Service Account e-mail. |
| 1. List of Asset Names | 1. ‘BYO\_logInPage’ – url to login page 2. ‘BYO\_allGrainBeers’ – url to beer cards page 3. ‘BYO\_credentials’ – user credentials to byo.com 4. ‘BYO\_gSheetId’ – beer master gSheet ID 5. ‘GoogleApp\_credentials’ – app password for logging in to Gmail without 2-step verification |
| 1. List of Queues Name | 1. ‘BYOBeerQueue’ – new beers identified on byo.com |
| 1. Schedule Details | Whenever you feel like brewing something new 😊 |
| 10 | Recommended Resolution | 1920 x 1080 |

# Project Details

In this section describe all the projects that compose the automated process.

For each project, describe the workflow(s) in the logical order that they are called in.

If the workflow is a flowchart, also include the exported image from Studio.

If the automated process is composed of multiple projects, copy paste and fill in the table below for each project with its specific details (there are 2 here already, assuming a dispatcher and performer project)

## Project Name: Dispatcher

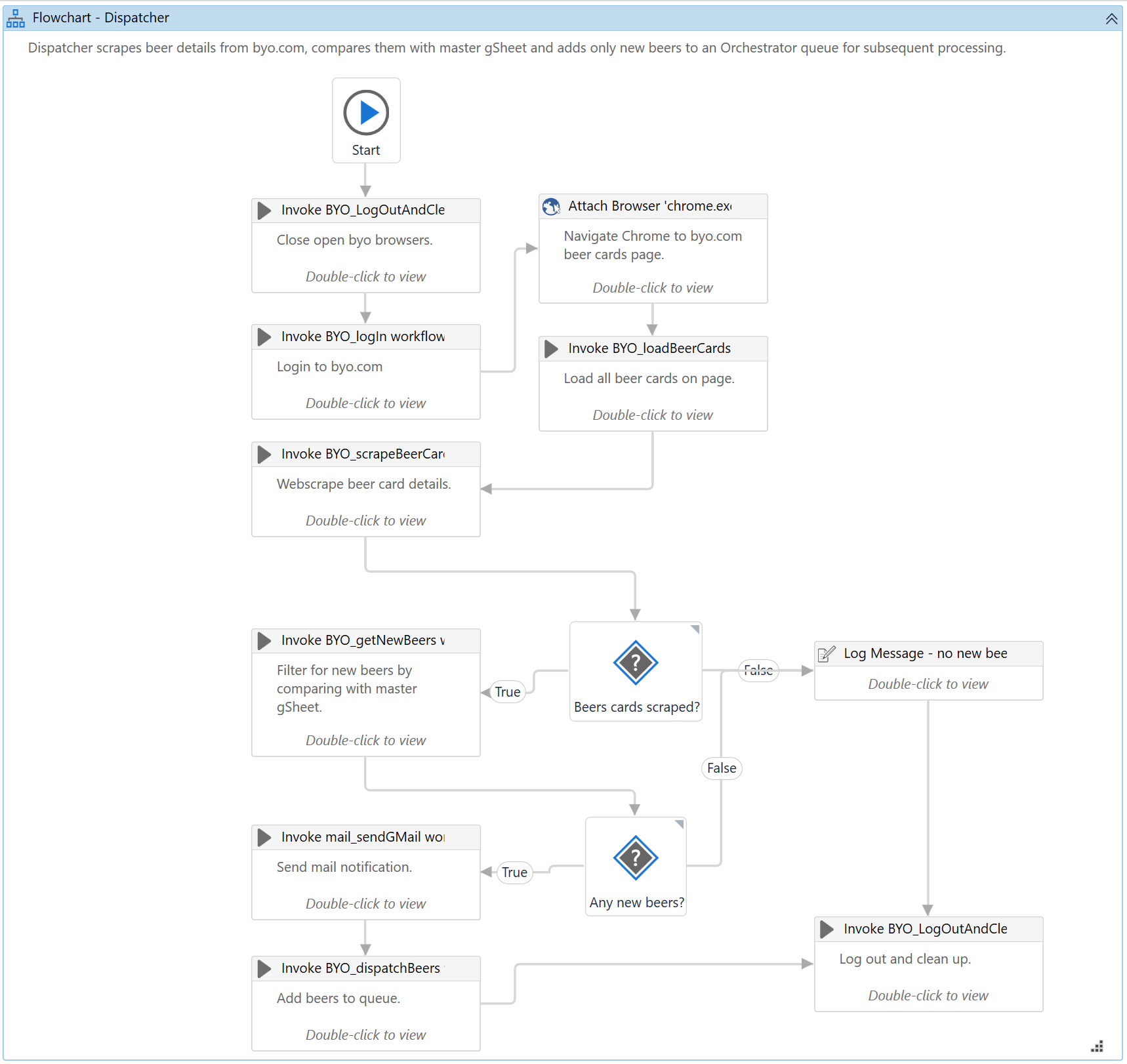
General information about the process selected for RPA prior to automation.

|  |  |  |
| --- | --- | --- |
| # | Item Name | Details  Fill in with free text. If not applicable, mark the field as “N/A". No empty fields. |
| 1 | Environment used for development  name, location, configuration details etc | Machine Name: laptop-at64rvek\jeshu  Location: C:\Users\jeshu\OneDrive\Documents\Code\RPA nanodegree\BYORecipeScraper  OS: MS Windows 10  CPU: Intel Core i7  GPU: N/A  RAM: 8 GB  Browser: Chrome  UiPath: UiPath Studio |
| 2 | Environment prerequisites  OS details, libraries, required apps | Byo.com account  UiPath Web Automation plugin for Chrome  Chrome browser  UiPath Studio  Google account |
| 3 | Logging level | Info: Custom log when starting/ending workflows and processing data  Warn: Beer recipe scrape failed  Error: Failed login to byo.com |
| 4 | Details about automation  if the apps were automated using UI Automation, Image & Text | UI automation via Chrome |
| 5 | In case of attended bot, can the user operate the computer while the robot is running? | No |
| 6 | Repository for project  where the developed project is stored | N/A |
| 7 | List of reused components | N/A |
| 8 | Custom logs defined in the workflows  where Throw Activity was used or custom log message was defined | **BYO\_login.xaml**:   * Info level message on start – *“Logging in to byo.com...”* andon completion – *“Login completed.”* * Error level message on page failing to login with BR exception – *“Login was unsuccessful. Check credentials and url.”*   **BYO\_loadBeerCards.xaml**:   * Info level message on start – *“Started webpage loading process...”* andon completion *– “Completed loading all <n> beers.”* * Info level message when loading of beer cards loop begins – *“Starting to load a total of #<n> beers...”*   **BYO\_scrapeBeerCards.xaml**:   * Info level message on start – *“Starting scraping of beers from byo.com...”* andon completion *– “Scrape finished. Found <n> beer cards.”*   **BYO\_getNewBeers.xaml**:   * Info level message on start – *“Starting to sort new beers...”* andon completion *– “Completed identification of <m> new beers of a total of <n> scraped beers.”*   **mail\_sendGMail.xaml**:   * Info level message on start – *“Sending mail with new beers...”* andon completion *– “Mail sent.”*   **BYO\_dispatchBeers.xaml**:   * Info level message on start – *“Starting upload of new beers to Orchestrator queue..”* andon completion *– “Finished queue upload.”*   **BYO\_LogOutAndCleanUp.xaml**:   * Info level log message when logging off/closing browsers begins – *“Logging off and closing open BYO windows...”* and finishes – *“Finished logging out and closing BYO windows.”* |
| 9 | Frequent errors found in the development phase | BYO\_loadBeerCards.xaml faied to load all beer cards on page because ajax autoloader did not respond consistently to hovering over. |
| 10 | Workarounds used in the automation phase | Added a pageUp + pageDown activity on the web page to trigger the ajax autoloader more reliably. |
| 11 | Configuration method  assets, excel file, Json file | Orchestrator assets and a local Json file |
| 12 | Configuration details  path for input files, configuration Orchestrator assets used | Assets:   * ‘BYO\_logInPage’ – url to login page * ‘BYO\_allGrainBeers’ – url to beer cards page * ‘BYO\_credentials’ – user credentials to byo.com * ‘BYO\_gSheetId’ – beer master gSheet ID * ‘GoogleApp\_credentials’ – app password for Gmail * ‘BYOBeerQueue’ – new beers identified on byo.com   Json file:   * Google API service account json key in local file outside of project folder |

### Workflow(s) specific to the Project

Define below all the workflow files (.xaml files) used in the project, with the Input and Output data.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Workflow File Name | Description | Arguments | Comments |
| 1 | Dispatcher.xaml | Main workflow | N/A | Flowchart |
| 2 | BYO\_login.xaml | Logs in to byo.com | N/A | Sequence |
| 3 | BYO\_loadBeerCards.xaml | Ensure all recipe cards load on screen | N/A | Sequence |
| 4 | BYO\_scrapeBeerCards.xaml | Scrape details from recipe cards once all loaded | out\_dt\_beerCardDetails - datatable | Sequence |
| 5 | BYO\_getNewBeers.xaml | Check scraped beer urls against master beer recipe collection gSheet and only add beers to Orchestrator queue if they were not already in the gSheet. | in\_dt\_scrapedBeers –  datatable  out\_dt\_newBeers - datatable" | Sequence |
| 6 | mail\_sendGMail.xaml | Send new recipe names, styles and urls from datatable by gmail to myself | in\_dt\_newBeers - datatable | Sequence |
| 7 | BYO\_dispatchBeers.xaml | Add scraped beer card details from beer cards page on byo.com to an Orchestrator queue | in\_dt\_newBeers - datatable | Sequence |
| 8 | BYO\_LogOutAndCleanUp.xaml | Log out and clean up left over browsers | N/A | Sequence |



## Project Name: Performer

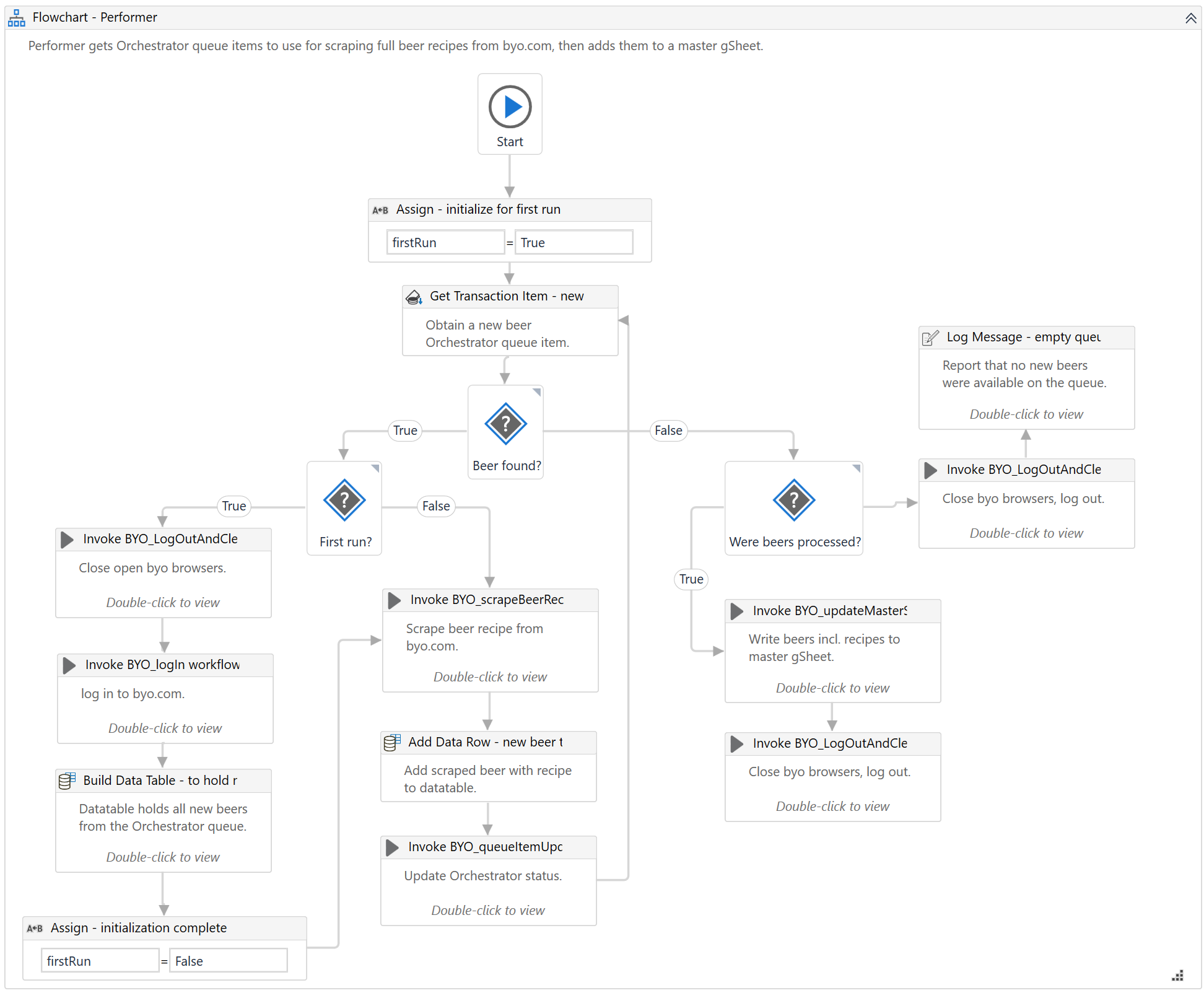
General information about the process selected for RPA prior to automation.

|  |  |  |
| --- | --- | --- |
| # | Item Name | Details  Fill in with free text. If not applicable, mark the field as “N/A". No empty fields. |
| 1 | Environment used for development  name, location, configuration details etc | Machine Name: laptop-at64rvek\jeshu  Location: C:\Users\jeshu\OneDrive\Documents\Code\RPA nanodegree\BYORecipeScraper  OS: MS Windows 10  CPU: Intel Core i7  GPU: N/A  RAM: 8 GB  Browser: Chrome  UiPath: UiPath Studio |
| 2 | Environment prerequisites  OS details, libraries, required apps | Byo.com account  UiPath Web Automation plugin for Chrome  Chrome browser  UiPath Studio  Google account |
| 3 | Logging level | Info: Custom log when starting/ending workflows and processing data  Error: Failed login to byo.com |
| 4 | Details about automation  if the apps were automated using UI Automation, Image & Text | UI automation via Chrome |
| 5 | In case of attended bot, can the user operate the computer while the robot is running? | No |
| 6 | Repository for project  where the developed project is stored | N/A |
| 7 | List of reused components | N/A |
| 8 | Custom logs defined in the workflows  where Throw Activity was used or custom log message was defined | **BYO\_login.xaml**:   * Info level message on start – *“Logging in to byo.com...”* andon completion – *“Login completed.”* * Error level message on page failing to login with BR exception – *“Login was unsuccessful. Check credentials and url.”*   **BYO\_scrapeBeerRecipe.xaml**:   * Info level message on start – *“Scraping of beer recipe started...”* andon successful completion *– “Finished scraping recipe for: <beer>”* * Warn level message if scraping of beer recipe fails – *“Failed to scrape recipe for <beer>. Exception message: <exception.message> thrown by source: <exception.source>.”*   **BYO\_queueItemUpdate.xaml**:   * Info level message on completion *– “Queue item status was updated.”*   **BYO\_updateMasterSheet.xaml**:   * Info level message on start – *“Starting to add new beers...”* andon completion *– “Finished adding <n> new beers to master gSheet.”*   **BYO\_LogOutAndCleanUp.xaml**:   * Info level log message when logging off/closing browsers begins – *“Logging off and closing open BYO windows...”* and finishes – *“Finished logging out and closing BYO windows.”* |
| 9 | Frequent errors found in the development phase | N/A |
| 10 | Workarounds used in the automation phase | N/A |
| 11 | Configuration method  assets, excel file, Json file | Orchestrator assets and a local Json file |
| 12 | Configuration details  path for input files, configuration Orchestrator assets used | Assets:   * ‘BYO\_logInPage’ – url to login page * ‘BYO\_credentials’ – user credentials to byo.com * ‘BYO\_gSheetId’ – beer master gSheet ID * ‘BYOBeerQueue’ – new beers identified on byo.com   Json file:   * Google API service account json key in local file outside of project folder |

### Workflow(s) specific to the Project

Define below all the workflow files (.xaml files) used in the project, with the Input and Output data.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Workflow File Name | Description | Arguments | Comments |
| 1 | Performer.xaml | Main workflow | N/A | Flowchart |
| 2 | BYO\_login.xaml | Logs in to byo.com | N/A | Sequence |
| 3 | BYO\_scrapeBeerRecipe.xaml | Get beer url from queue item, navigate to beer recipe page and scrape full recipe. | in\_transactionItem\_newBeer - queueItem  out\_beerRow - array  out\_status - string  out\_failReason – string | Sequence |
| 4 | BYO\_queueItemUpdate.xaml | Update queue item with relevant status depending on recipe scraping success or failure. | in\_transactionItem\_newBeer - queueItem  in\_status - string  in\_failReason - string | Sequence |
| 5 | BYO\_updateMasterSheet.xaml | Append recipes from datatable to master recipe collection gSheet | in\_dt\_newBeers - datatable | Sequence |
| 8 | BYO\_LogOutAndCleanUp.xaml | Log out and clean up left over browsers | N/A | Sequence |



# Compliance Considerations and Reporting Requirements

* Passwords/credentials
* Google API service account json key must be kept in local file outside of project folder

# Other Details

## Future Improvements

* Move Performer to REFramework for better allround exception handling and logging
* Local config file to store Orchestrator asset and queue names
* With BYO\_scrapeBeerRecipe.xaml merge beer url and name into one in the out\_beerRow array (=HYPERLINK(<beer url>,<beer name>))
* Further divide a scraped beer recipe into parts of ‘Stats’, ‘Ingredients’, ‘Step by Step’ and ‘Other Instructions’ using regex or code
* Add code to clean up a scraped beer recipe if it contains partial mash instructions – should keep all-grain instructions

## Debugging Tips

* BYO\_login.xaml waits default 30 secs when user is not logged in because it waits to find the “My account” element. Hard-code this shorter to avoid this wait when testing.
* BYO\_loadBeerCards.xaml hovers over auto-load element which does not always show up and so loading freezes. This seems to have been resolved by sending a PgUp+Down hotkey command to "shake" the page a bit
* Had a temporary test.xaml workflow and tried smaller parts out during development to identify errors in the initial workflow parts, e.g. experimented with Browser instances to enable correct catching and closing of BYO browsers in a While loop instead of hard-coding a delay to avoid re-catching an already closed window
* Stored credentials in Orchestrator from the start to avoid hard-coding them in and finding and replacing those afterwards
* Added temporary logging in the workflow files to state when certain places completed successfully

## Other Remarks

* Errors occur when loading the project saying *Failed to load assembly:*
  + \.nuget\packages\microsoft.aspnetcore.connections.abstractions/3.1.6\lib/netstandard2.0/Microsoft.AspNetCore.Connections.Abstractions.dll
  + \.nuget\packages\microsoft.aspnetcore.http.connections.client/3.1.6\lib/netstandard2.0/Microsoft.AspNetCore.Http.Connections.Client.dll
  + \.nuget\packages\system.text.json/4.7.2\lib/net461/System.Text.Json.dll
* Numerous errors occur when using the Analyze File feature in UiPath Studio
* Possible cause of above errors: Related to incompatible activity packages (i.e. higher package revision than what is compatible with UiPath Studio 2019.10.5)?

# Post UAT Specifications

* Average duration per transaction (varies depending on the Test environment): Dispatcher takes ~6 minutes to load and scan ~1100 beer cards where majority of time is spent loading beer cards on the page. Performer takes 2-5 seconds to scrape one beer recipe which amounts to about 1½ hours to scrape all current ~1100 beer recipes on byo.com.
* Recommended number of robots for the specified volumes: 1
* Specified schedule: Whenever you feel like brewing something new 😊

# Glossary

* **Master project** - the overall output of the development, containing one or multiple projects that together cover the scope of the robotic process automation.
* **Project** - a UiPath Studio project containing one or multiple workflow files. A project can be converted to a package and run independently, covering a particular scope within the master project. The project is used when defining the development and support phase of the automation.
* **Package** - the output of compiling a project. A package can be deployed on the robot machine and be executed by the robot service. Only one package can be executed at a given time by a robot. The package is used when defining the running phase of the automation
* **Workflow** - a component of the package, the workflow encapsulates a part of the project logic. The workflow can be of type: sequence, flowchart or state machine. a workflow is saved as an .xaml file inside the project folder. A workflow file can be invoked from another workflow and by default there is an initial workflow file that will run when executing the package.
* **Activity** - an action that the robot executes.
* **Sequence** - a workflow where activities are executed one after another, in a sequential order
* **Flowchart** - a workflow where activities are connected by arrows and the logic of the workflow can be easily followed in a visual manner. The flowchart can also be exported as an image from UiPath studio
* **State machine** - a more advanced way of organizing a workflow, similar to a flowchart.
* **BOR** - Back office robot
* **FOR** – Front office robot
* **Orchestrator** – Enterprise architecture server platform supporting: release management, centralized logging, reporting, auditing and monitoring tools, remote control, centralized scheduling, queue/robot workload management, assets management.