Development Specifications Document (DSD)

*Process Name: YouTube Scraper*

Table of Contents

[I. Document Overview 1](#_Toc114006549)

[II. Automated Master Project Details 1](#_Toc114006550)

[III. Runtime Guide 2](#_Toc114006551)

[1. Runtime Diagram 2](#_Toc114006552)

[2. List of Packages 3](#_Toc114006553)

[3. Master Project Runtime Details 3](#_Toc114006554)

[IV. Project Details 5](#_Toc114006555)

[1. Project Name: YouTube Scraper 5](#_Toc114006556)

[1.1 Workflow(s) specific to the Project 7](#_Toc114006557)

[V. Compliance Considerations and Reporting Requirements 9](#_Toc114006558)

[VII. Other Details 9](#_Toc114006559)

[1. Future Improvements 9](#_Toc114006560)

[2. Debugging Tips 9](#_Toc114006561)

[3. Other Remarks 9](#_Toc114006562)

[VIII. Post UAT Specifications 9](#_Toc114006563)

[IX. Glossary 10](#_Toc114006564)

Version Control

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Date | Version | Role | Name | Organization Department | Function | Comments |
|  | 1.0 | Author | Jesse Peterson | Udacity | Student |  |

# 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.

# Automated 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 | YouTube Scraper |
| 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 |

# Runtime Guide

## Runtime Diagram

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

3. Automation Loads YouTube and Applies Users Inputs in Search

2. Automation writes Startup Form input to Log File

1. User fills out Startup Form



6. Log File is Upload to UiPath Orchestrator Queue

5. Automation writes YouTube Results to Log File

4. Automation Scrapes each record in the YouTube Search Results



8. Automation sends email via Gmail with Start Form Log File and YouTube Results Log File

7. Automation pulls Log File Transaction from UiPath Orchestrator Queue

## 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 | UiPath.Excel.Activities 2.11.4 | The package for managing activities that read and write from Microsoft Excel compatible files |
| 2 | UiPath.Form.Activities 1.9.1 | The package for generating UI forms that allow users to enter input data captured as variables or arguments. |
| 3 | UiPath.Mail.Activities 1.12.2 | The package for sending and receiving emails |
| 4 | UiPath.System.Activities 21.10.2 | The package for interacting with local file systems |
| 5 | UiPath.UIAutomation.Activities 21.10.3 | The package for implementing UI and Desktop automation activities. |

\*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 | UiPath Version 2022.4.3 |
| 2 | Prerequisites to run | Any text editor capable of reading csv files as well as a gmail account |
| 3 | Input Data | A UiPath form that captures user input |
| 4 | Expected Output (output data) | Two log files sent via email; One log file to capture the users input and another log file to capture the result-set from the search performed by the automation. |
| 5 | How to start the automated process? | A user clicks on the ‘Run’ or ‘Debug’ button. |
| 6 | Resuming the process from a particular step | A users selects the ‘Run from this Activity’ button. |
| 7 | Reporting  queues reporting, Kibana or another platform | There are two reports used by this automation:   * One local ‘csv’ logfile capturing the users input into the automation. * One ‘csv’ logfile capturing the transactions from the UiPath Orchestrator queue.   Both of these reports are delivered via email upon the successful completion of the automation. |
| 8 | Manual Error Handling  roll back or manually complete failed transactions. Procedures to reset the item. Ex “set status as investigating” | This automation requires users to manually roll back the automation when errors are generated. |
| 1. How to resume the process in case of error | The Process should be resume either from the beginning if there was a exception thrown OR the users should manually hit the ‘retry’ button if a selector was not able to be identified. |
| 1. How to manually fix transactions with error | When errors occur, the user should stop the automation, manually investigate the error, fix the issue, then rerun the automation from the beginning OR use the ‘ignore’ feature to bypass the current error. |
| 9 | Use of Orchestrator | This automation uses a single UiPath Orchestrator Queue for pushing and pulling transactions. |
| 1. Password Policies   specific compliance requests? | N/A |
| 1. Stored Credentials   Never hard code credentials in the workflow | N/A |
| 1. List of Asset Names | N/A |
| 1. List of Queues Name | “YouTube\_Scraper\_Queue” |
| 1. Schedule Details | N/A |
| 10 | Recommended Resolution | 1920x1080 |

# 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: YouTube Scraper

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 | UiPath Studio Version 2022.4.3 |
| 2 | Environment prerequisites  OS details, libraries, required apps | OS Details: Windows 10 Enterprise, v20H2  Libraries:   * UiPath.Excel.Activities 2.11.4 * UiPath.Form.Activities 1.9.1 * UiPath.Mail.Activities 1.12.2 * UiPath.System.Activities 21.10.2 * UiPath.UIAutomation.Activities 21.10.3   Require Apps:   * Microsoft Excel or any text editor * Gmail Account |
| 3 | Logging level | ‘Info’ for debugging  ‘Error’ for error handling |
| 4 | Details about automation  if the apps were automated using UI Automation, Image & Text | This automation uses the ‘Legacy’ version of UiPath and the ‘UiPath.System.Activities ‘ library to prompt the user to enter data into a UI form using the ‘UiPath.Form.Activities’ library, performs a YouTube search and scrapes the results using the ‘UiPath.UIAutomation.Activities’ library, saves the results to a local file using the 'UiPath.Excel.Activities’ library, then sends a summary report through email using the ‘UiPath.Mail.Activities ‘ library. |
| 5 | In case of attended bot, can the user operate the computer while the robot is running? | The user cannot operate the computer while the robot is running. |
| 6 | Repository for project  where the developed project is stored | GitHub link: [Jrpete/Udacity\_YouTube\_Scraper\_Automation](https://github.com/jrpete/Udacity_YouTube_Scraper_Automation) |
| 7 | List of reused components | There are no reused components in this automation |
| 8 | Custom logs defined in the workflows  where Throw Activity was used or custom log message was defined | Main.xaml   * Catches generic exceptions and generates a log message if the user is unable to initiate the automation.   Enter\_Form\_Process.xaml   * Catches an InvalidDataException if the users enters data that doesn’t match the requirements of the form. * Catches a WriterException if the automation is unable to write the users input into a CSV Log File.   Load\_YouTube\_Process.xaml   * Catches UiElementHasNoItemsException if there are no selectors in the YouTube Search Results.   Scrape\_Results\_Process.xaml   * Catches a SelectorOperationException if there are selectors in the YouTube Search Results but the automation is unable to extract the text.   Post\_Data\_To\_Orchestrator.xaml   * Catches a OrchestratorHttpException if the Automation is unable to connect, push, or pull Transactions to UiPath Orchestrator   Send\_Status\_Report.xaml   * Catches a WriterException if the automation is unable to write the YouTube Search Results into a CSV Log File. * Catches a MailException if the automation is unable to log into Gmail or send an email. |
| 9 | Frequent errors found in the development phase | The most frequent errors occurred when the automation is unable to identify a selector in the YouTube filters or search criteria. Additional error handling was implemented to account for these errors. |
| 10 | Workarounds used in the automation phase | More specific CSS selectors had to be used to find a reliable way of selecting the YouTube filters and IF-statements were used to validate the outputs to ensure that the correct attributes were read. |
| 11 | Configuration method  assets, excel file, Json file | There are two assets:   * A .csv logfile generated from the users input * A .csv Status Report generated from the YouTube Search Results |
| 12 | Configuration details  path for input files, configuration Orchestrator assets used | All output files are stored in the ‘Logs’ and ‘Status’ report sub-folders within the automation. |
| 13 | Workflow File Export List  Use the project mapping tool | Main.xaml  Enter\_Form\_Process.xaml  Load\_YouTube\_Process.xaml  Post\_Data\_To\_Orchestrator.xaml  Scrape\_Results\_Process.xaml  Send\_Status\_Report.xaml |

### 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 | Main.xaml | The Entry Point workflow that executes each subworkflow | None | Requires a Workflows, Logs, and Status\_Reports folder 1 level below the root level |
| 2 | Enter\_Form\_Process.xaml | Prompts user to fill out a form to be used as the Search Input criteria for YouTube and creates a logfile | out\_Features, String; out\_Duration, String; out\_SortBy, String; out\_UploadDate, String;  out\_Type, String; out\_SearchTerms, String; out\_SearchCount, String; out\_LogFilePath, String. | Saves Logfile to the root of the Project |
| 3 | Load\_YouTube\_Process.xaml | Takes the users inputs from Enter\_Form\_Process.xaml and searches YouTube using the arguments as filters and validates the filters. | in\_URL, String; in\_SearchTerm, String; in\_UploadDate, String; in\_Type, String; in\_Duration, String; in\_Features, String; in\_SortBy, String. | Users are allowed to choose a 'None' filter option, all filters are validated prior to applying |
| 4 | Scrape\_Results\_Process.xaml | Scrapes the result-set from YouTube after the filters have been applied and adds them to a DataTables | in\_SearchCount, Int32;  out\_Title, String; out\_Creator, String; out\_VerifiedStatus, String; out\_WatchCount, String; out\_Description, String;  out\_TagList, String; out\_ResultsDT, DataTable, in\_Feature, String | Added validation to some selectors to account for filters returning different css selectors. |
| 5 | Post\_Data\_To\_Orchestrator.xaml | Posts DataTable of YouTube results to UiPath Orchestrator Queue, then retrieves results from Queue to confirm that they were successfully posted. | in\_ResultsDT, DataTable, out\_SuccessTransactionCount, String, out\_FailQueueItemsCount, String; out\_InProgressQueueItemsCount, String; out\_DeleteQueueItemsCount, String. | Adding a List or Array to a UiPath Orchestrator Queue requires you to first Serialize it as a JSON object. |
| 6 | Send\_Status\_Report.xaml | Creates log file of YouTube Search Results, Emails Search Results and Input Log File along with UiPath Orchestrator Summary Statistics | in\_ResultDT, DataTable; in\_LogPath, String; in\_FailQueueItemsCount, String; in\_DeleteQueueItemsCount, String; in\_InProgressQueueItemsCount, String; in\_SuccessTransactionCount, String | Users will be prompted to log into their Gmail account |

# Compliance Considerations and Reporting Requirements

1. There are no existing compliance considerations or reporting requirements.

# Other Details

## Future Improvements

* Allowing users to execute multi-threaded searches using the same input data.
* Passing filter choices are URL Parameters
* Adding regex to filter out character names in the YouTube Video descriptions
* Performing a bulk scrape of all results using the default ‘Data Scraping’ activity then filtering the results within UiPath according to the users inputs.

## Debugging Tips

* If one of the selectors in the automation fails during runtime, it is possible that the YouTube developers have pushed out some new changes that will require the user to update the selectors.
* Sometimes when there is a bad network connection, the automation will throw an error stating that the selector cannot be identified. Hitting the ‘retry’ button after a few second will often resolve the issue as the selectors is then loaded into the automations frame.
* Posting bulk data to UiPath Orchestrator requires every column in the data table to be of type ‘String’. Serializing objects into Strings is necessary to prevent errors while attempting to Bulk Upload to Orchestrator.
* Sometimes it is necessary to validate the output file that is sent via email as it contains non-English characters that are not easily captured.

## Other Remarks

* Users will need a Gmail account before running the automation and will be prompted to login while executing the automation.

# Post UAT Specifications

* Average duration per transaction (varies depending on the Test environment):
  + 15 seconds per YouTube Video
* Recommended number of robots for the specified volumes:
  + 1 robot / every 1000 videos searched
* Specified schedule:
  + Run Once Per day

# 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.