­­­­­

|  |  |
| --- | --- |
|  |  |



Solution Design

Document

Automated Product Price Monitoring

Check product prices on Amazon and Flipkart. Compares prices, generates a reports with the lowest price option, and sends an email notification.

Revision History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Rev. # | Date | Section/Page# | Revision Summary | Author |
| 1 |  |  |  |  |

Table of Contents

[I. Purpose 4](#_Toc166143838)

[2 Automated process details 5](#_Toc166143841)

[2.1 Summary 5](#_Toc166143842)

[2.2 Developer(s) 5](#_Toc166143843)

[3 Runtime guide 6](#_Toc166143844)

[3.1 Architectural structure of the Master Project 6](#_Toc166143845)

[3.2 Master Project Runtime Details 6](#_Toc166143846)

[3.3 Automated Product Price Monitoring Project 6](#_Toc166143847)

[3.4 Project(s) workflows 7](#_Toc166143848)

[3.5 Packages 7](#_Toc166143849)

[3.6 Compliance Considerations 8](#_Toc166143850)

[4 Other Details 9](#_Toc166143851)

[Remarks 9](#_Toc166143852)

[5 Glossary 10](#_Toc166143853)

# Purpose



Outlines the major components of the Master Project (the overall output of the development, containing one or multiple projects that together cover the scope of the robotic process automation) taking into account all the business restrictions (scheduling, peaks, future increases in volume etc.). The focus of the Solution Architect will be on:

* Robustness;
* Scalability;
* Efficiency;
* Reliability;
* Reusability of component

The information herein is targeted primarily at the developers that will initially implement the solution and subsequently at the support developers in case of change requests.

# Automated process details

|  |  |
| --- | --- |
| Item | Description |
| Master Project Name | Automated Product Price Monitoring |
| Robot Type | BOR |
| Orchestrator used? | Yes |
| Scalable | No |
| UiPath version used | 2024.2.0 |

## Summary

This process automates product comparisons between Amazon and Flipkart. It searches for products on both platforms and gathers details like description, price, and discounts. Products are matched based on description similarity and key features.

* Search: Products are searched on Amazon and Flipkart.
* Data Extraction: Product details like URL, description, original price, discount price, and discount percentage are collected.
* Matching & Analysis: Products are matched across platforms. Price difference and percentage change are calculated.
* Reporting: Matched products with lower prices are reported in Excel sheets (one sheet per product). Reports include details from both platforms and the price comparison.
* PDF Generation: PDF reports of the lowest-priced products across all searches.
* Email Delivery: Final Excel files and individual product PDF reports are sent to the user via email.

This process helps users find the best deals by comparing product offerings and prices across Amazon and Flipkart.

## Developer(s)

* Raj Lad

# Runtime guide

## Architectural structure of the Master Project

Display the interaction between components (package / robots, Orchestrator queues, and running order) in a diagram

## Master Project Runtime Details

|  |  |
| --- | --- |
| ITEM NAME | DESCRIPTION |
| Production environment details | Running on Local Machine. |
| Prerequisites to run | Have queue items with in orchestrator queue  Having Microsoft Excel on the machine  Having Chrome Browser on the machine |
| Input Data | List of Product Name |
| Expected output | Email the user with:   * Excel reports with product details and price comparisons (one sheet per product). * PDF reports for the lowest-priced products across all searches. |
| How to start the automated process | The process will be started from orchestrator server (demo.uipath.com) |
| Reporting | Orchestrator logs and jobs dashboards. |
| How is Orchestrator used? | Orchestrator used for process running, and assets. |
| Password policies | Email credentials with the app password which is not expiring |
| Stored credentials | Stored in Orchestrator Assets |
| List of queues names | APPM\_ProductQueue |
| Schedule Details | N/A |
| Multiple Resolutions Supported? | N/A |
| Recommended Resolution | N/A |

## Automated Product Price Monitoring Project

|  |  |
| --- | --- |
| ITEM NAME | DESCRIPTION |
| Environment used for development | Development Machine, UiPath Studio, UiPath Assistance, UiPath Orchestrator |
| Environment prerequisites | Windows 10 or above, UiPath Studio (Community or Enterprise Edition), Microsoft Excel, Chrome Browser |
| Repository for project | N/A |
| Configuration method | Assets, Config.xlsx File |
| List of reused components | N/A |
|
| List of new reusable components | N/A |

## Project(s) workflows

### Dispatcher

|  |  |  |
| --- | --- | --- |
| Workflow Name | Arguments | Description |
| AddProductItemToQueue | In\_FilePath | **Input:** Read the data from the excel file based on the input path in the tabular form**.**  **Bulk Insert Data to Queue:** Insert all the data to the orchestrator queue  **Logging:** Logging messages are generated for:   * The start of the dispatcher process. * The successful insertion of data to queue. |

### Performer

|  |  |  |
| --- | --- | --- |
| Workflow Name | Arguments | Description |
| ExtractProductListFromAmazonFlipkart | In\_TransactionItem | **Input:** It retrieves a product name from the Transaction Item of a queue**.**  **Product Search & Data Extraction:** It searches for the product on both Amazon and Flipkart.  **Data Presentation:** The extracted relevant data (product url, title, original price, discount price, discount percentage, ram (in case of flipkart)) is presented in separate data tables for each platform.  **Error Handling:** The activities for opening browsers, searching, and data extraction are encapsulated within a Try-Catch block. This ensures the process continues even if:   * The server becomes unavailable. * There are unexpected changes in the UI elements of the websites. * No product found based on search.   **Logging:** Logging messages are generated for:   * The start of the search process. * The completion of data extraction. * Any exceptions encountered during execution. |
| FilterExtractedDataTable | io\_dt\_ProductDataTable,  in\_ProductName | The general process for filtering and manipulating the data tables extracted from both Amazon and Flipkart.  An additional column named "NewTitle" is added to data table. This column will store the manipulated product title.  **Data Row Processing:**  For each row in data table:   * **URL Shortening:** The URL in the row is shortened to meet specific requirements (up to the ‘?’ as after that parameters are not required in URL). * **RAM Column Check:** The data table is checked for a column named "RAM".   + **If RAM Column Exists:**      - The product title is converted to lowercase and the RAM value (converted to lowercase) are concatenated and stored in the "NewTitle" column.   + If RAM Column Does Not Exist:     - The product title is converted to lowercase and stored in the "NewTitle" column. * The "NewTitle" is compared against the original product name. * **Product Name Mismatch:** If the "NewTitle" doesn't contain the original product name, the row is marked for removal and stored in List of type DataRow.   **Row Removal:**   * For each item in the list of rows marked for removal will be deleted from its respective data table   **Exception:**  Throw the Business Exceptionin case if it get no product based on search.  **Logging:** Logging messages are generated for:   * The start of filtering of table * For no searched product item found on any of the platform * No matched product found |
| FindProductMatch | in\_dt\_FlipkartData,  in\_dt\_AmazonData,  out\_dt\_MatchRecords, | The process of identifying similar products between Amazon and Flipkart data tables.  **Output Table Creation:**  An output table will be created to store the comparison results. The table will include the following columns:   * Amazon Product Url * Amazon Product Title * Amazon MRP * Amazon SP * Amazon Discount * Flipkart Product Url * Flipkart Product Title * Flipkart MRP * Flipkart SP * Flipkart Discount * Price Difference * Discount Percentage   **Matching Logic:**   * For each Flipkart row: Extract the Color, RAM, and Storage values from the corresponding "NewTitle" column using string manipulation techniques.   + For each row in the Amazon data table :     - Perform a case-insensitive comparison to check if the "NewTitle" column of the current Amazon row contains the extracted Color, RAM, and Storage values from the corresponding Flipkart row.     - If a match is found between an Amazon and Flipkart row based on the extracted features: process it for further comparisons.   **Logging:** Logging messages are generated for:   * The start of process of matching the product items of both platform. |
| IfProductMatched | in\_CurrentAmazonRow,  in\_CurrentFlipkartRow,  io\_dt\_MatchedRecords, | This process describes the actions taken when a match is found between a product on Amazon and Flipkart based on the previously defined criteria.  **Price Check:** Verify if the Selling Price (SP) for either the Amazon or Flipkart product is not null or empty.  **Price Difference Calculation (if applicable):** The price difference is calculated by subtracting the lower price from the higher price.  **Percentage Difference Calculation (if applicable):**  If a price difference exists (greater than 0):   * The percentage difference is calculated based on the product with the higher price. Here's a possible formula:   Percentage Difference = (Higher Price - Lower Price) / Higher Price \* 100  **Data Table Update:** The extracted data for the matched products, including the price difference and percentage difference (if applicable), is added to the "Matched Records" data table.  **Lowest Price Product Report:** Create an Excel file for product with a separate sheet for each product item, containing details of the lower-priced product found during the comparison between Amazon and Flipkart.  **Logging:** Logging messages are generated for:   * The start of process for the further action for the matched product. * In case the price difference is zero |
| GenrateLowestPriceProductExcelReport | in\_ProductURL,  in\_ProductTitle,  in\_ProductMRP,  in\_ProductSP,  in\_ProductDiscount | The process for generating an Excel report containing details of the lowest-priced product(s) identified during the search.  **Data Extraction:** Navigate to the product URL and extract additional data required for the report:   * Bank Offers * Partner Offers * Product Image (Download it)   **Excel File Preparation:** Create a new Excel file or locate an existing one with a filename as “ProductName.xlsx” that reflects the searched product  **Sheet Creation:** Create a new sheet within the Excel file and name it based on the product title.  **Data Population:**  Populate the newly created sheet with the following data in separate columns:   * Product Image (insert image) * Application * Product URL * Product Title * Product MRP (Maximum Retail Price) * Product SP (Selling Price) * Product Discount * Bank Offers * Partner Offers   **Error Handling:** The activities for opening browsers, searching, and data extraction are encapsulated within a Try-Catch block. This ensures the process continues even if:   * The server becomes unavailable. * There are unexpected changes in the UI elements of the websites.   **Logging:** Logging messages are generated for:   * The start of the search process for lowest price product report data extraction. * Any exceptions encountered during execution. |
| ArchivePreviousReports |  | This process archives the previously generated Excel and PDF reports.  **Compression:** The reports generated in previous process run are compressed into a single archive file.  **Filename:** The archive filename follows a specific format that includes the date of current process is running as Reports\_[YYYY-MM-DD].zip. This ensures easy identification and versioning of archived reports.  **Logging:** Logging messages are generated for:   * The start of the process for archiving previous report. * If no files are available to archive. |
| GenerateExcelReport | in\_dt\_MatchedRecords | The process generate an Excel report containing data for products identified as matches during the comparison between Amazon and Flipkart.  **Report Creation:** A new Excel file is created to store the report data.  **Sheet Creation and Population:** For product:   * A new sheet is created within the Excel file. * The sheet is named based on the product name. * The following data is then written to the corresponding sheet for each matched product:   + Amazon Product Url   + Amazon Product Title   + Amazon MRP (Maximum Retail Price)   + Amazon SP (Selling Price)   + Amazon Discount   + Flipkart Product Url   + Flipkart Product Title   + Flipkart MRP   + Flipkart SP   + Flipkart Discount   + Price Difference   + Discount Percentage   **Logging:** Logging messages are generated for:   * The start of the search process for inserting matched records data to excel sheet. * For No data to insert. |
| GeneratePDFReport |  | The process of converting individual sheets from the generated Excel report (containing lowest-priced product data) into separate PDFs and then merging them into a single, consolidated PDF.  **Input Excel File:** Retrieves the Excel file that was previously created and updated with details of the lowest-priced products for each search.  **PDF Conversion:**  Save the Excel file as PDF  **Clean-up:** Delete the excel file once the PDF file is generated.  **Logging:** Logging messages are generated for:   * The start of the search process for converting the lowest price product excel report to pdf. * For No file to convert. |
| SendMail | in\_Subject,  in\_Body,  in\_Attachments | The process of sending email notifications to users throughout the execution of a process.  **Process Start:** An email to be sent to the user upon initiating the automation process.  **Process Completion:** Another email to be sent upon successful completion of the process with all the reports generated.  **Process Exception:** If an exception occurs during execution, an email notification will be triggered. This email might include details about the error and potentially a screenshot where the error occurred (if applicable).  **Logging:** Logging messages are generated for:   * The sending the mail for start of process * The sending the mail for completion of process * The sending the mail for any exception in the process |

## Packages

|  |  |
| --- | --- |
| Package Name | Description |
| UiPath.Excel.Activities (Version: [2.22.3]) | This is the core UiPath package for interacting with Excel files. It allows you to perform various actions such as reading data, writing data, formatting cells, manipulating worksheets, and more. |
| UiPath.PDF.Activities (Version: [3.19.0]) | This package provides activities for working with PDF documents. You can use it to extract text, manipulate pages, merge PDFs, convert PDFs to other formats, and more. |
| UiPath.System.Activities (Version: [24.3.1]) | This core UiPath package offers essential activities for general automation tasks. It includes functionalities like working with variables, conditional statements, loops, date and time manipulation, file operations, and more. It forms the foundation for building robust automations. |
| UiPath.UIAutomation.Activities (Version: [23.10.11]) | This core UiPath package offers a wide range of activities for interacting with user interfaces (UI) of desktop applications and web browsers. It allows you to simulate user actions like clicking buttons, entering text, navigating menus, and more. It's essential for automating tasks within various software applications. |
| BalaReva.Excel.Activities (Version: [2021.1.0]) | This is a third-party package that might offer additional functionalities for working with Excel data beyond the core UiPath.Excel.Activities package. This offers Advanced data manipulation, Interacting with specific Excel features, Compatibility with older or less common Excel formats and many more. |

## Compliance Considerations

|  |  |
| --- | --- |
| Item | Specific Considerations |
| Sarbanes Oxley (SOX) | N/A |
| HIPAA | N/A |
| FERC Standards of Conduct | N/A |
| Personally Identifiable Information (ensure process follows PoLP) | Yes |

# Other Details

### Remarks

* The process will be triggered by queues. Specifically, it will run whenever there are at least three items in the queue.
* Each queue item should contain only details of smartphones.
* The product names within the queue items must exactly match the corresponding product names on both Amazon and Flipkart.
* The process will focus on searching for the top versions of models mentioned in the product names.

# Glossary

The main terms used in the Solution Architecture Document are defined below:

**Master project** - the overall output of the development, containing one or multiple projects that together cover the scope of the robotic process automation. There is a 1 to 1 connection between the Master Project and the Process to be automated (As presented in the PDD).

**Project** - an 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. Or multiple projects can be converted into one package depending on the aims and restrictions of the automation. The project is used when defining the development and support phase of the automation.

**Package** - the output of compiling one or multiple projects. 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.