### Detailed Automation Approach

Automation approach will be N-1 , meaning scripts development will be 1 sprint behind. The automation scripts will be shared with the acceptance testing team as from reasonable component stability.

The automation scope will be decided upon together with the Client acceptance Qualifications team through a review of the integration test plan for the drop. The below ‘Automation Best Practices’ will allow for easy reusability of the Integration automation script on the client environment.

The automation setup is the same as the one in use with client for legacy systems.

* Data driven framework or Robot framework, implemented with JAVA or PYTHON will be used as test automation framework.
* Different libraries will be applied as an when required
* PYCHARM or ECLIPSE editor for development and execution of test cases
* Jenkins as CI/CD tool
* Git or Bit bucket will be used as code repository
* Keyword or method logic is to be based on different levels
* Single Object Interaction [Select from dropdown list , enter text in text field…]
* Multiple Object Interaction [Fill the address , Customer configuration…]
* Action Level [Next Page , Finish/Submit, Ok , Cancel…..]
* Validations/code robustness should be done at all levels
* No hardcoded environment specific variables or Test Data
* Keywords should be reusable [self-cleaning]
* Automation captures relevant results
* Compare actual and expected results and highlight the discrepancies if any
* Print screen should be available on failure
* Reports analysis of PASS and FAIL
  1. Automation Development and Execution Approach

Automation commences from the first sprint of every drop. N-1 approach is adopted for the automation script development and execution being 1 sprint behind.

Scripts will be executed individually post development and reports will be uploaded to the user stories for test evidences. Test cases developed for the sprints will be added to the regression suite and will be scheduled to run at the end of the sprint in SI testing phase.

For each release, the last sprint test cases along with backlogs will be developed and the whole regression suite will be executed, auto uploading the reports to test management tool.

Client team would be provided with SI repository access to pull automation scrips from Bit bucket or Git.

* 1. Test Automation Process

* Identify the regression suite and perform the automation feasibility
* Prepare the Automation script Estimation along with ETA
* Develop the automation scripts based on the identified regression test cases
* Perform a code and technical review once development is completed
* Create Pull request to deliver a code
* Send an email communication to seek **sign off** for completed sprints along with all details and execution results
* After sign off merge the code to target branch.
  1. Common Test Automation Requirements
     1. Automation Framework capabilities

Automation framework must be designed in generic way and must be able to access all areas of the targeted application in order to write fully end2end test cases.

Considering application uses multiple technology , our framework should be flexible to extends its capabilities to test the applications pertaining to the different technologies.

In addition, our framework must be able to control the COM [Component object model] objects like excel, notepad, other office tools.

* + 1. Automation Best Practices
* Validations/code robustness should be done at all levels
* Test Data must be created in separate files and feed to the test cases
* No hardcoded environment specific variables or test data
* User defined keywords or methods should be reusable [Self-cleaning]
* Actual results should be evaluated against the expected results and highlight the discrepancies if any
* Independent object repository container should be created
* Screenshots of the failed test cases should be captured for detailed verification
* Automated test cases must be able to be executed alone, without requiring manual preparation, intervention and without being dependent of other tests
* All the pre-requisition for the test to start and clean up after execution must be included in the test itself.
  + 1. Version Control

Automated sources must be stored on a central repository and previous versions must be easily retrievable. Bit bucket/ Git code hosting platform is preferred for code housing.

* + 1. OS Portability

Automated test should be able to run on different OS versions mentioned in the system requirements [FRS]

* + 1. Browser Support

Automated tests should run on all supported browsers and should be compatible with different browser versions as mentioned in the system requirement [FRS].

* + 1. Integration with Jenkins/Octane or other tools to upload results

Test Status and test evidences should be uploaded into micro focus test management tools. That tool should be integrated with Jenkins CI/CD to auto upload the run results.

* + 1. Scheduling the Test Execution

Automation tests should be scheduled to execute at specific hours on specific days. This to allow automated regression to be executed outside normal working hours.

Jenkins(CI/CD) tool is preferred to schedule the test remotely.

1. Guidelines for Test Automation
   1. Tests that can be automated   
        
      There are many factors which determines what should be automated first. We Believe the following candidate fall in line first for automation and there must each be assessed to ensure cost vs benefits efficiency.

* Business critical flows where in a single failure would cause a major impact on the system
* Test that needs to run against each release such as Smoke/sanity.
* Frequently used functionalities.
* Test that executes same workflows with different data sets.
* Tests that needs to run against multiple configuration different OS and Browser combinations.
* Test that have easily measured output.
* Test with consistent behavior and regular controls.
  1. Test that cannot be automated
* Test that run Only once
* Test with unstable objects
* Test that involves manual intervention
* New feature without functional validation should not be considered for the automation.
  1. Know the functional flow

Automation testers are always recommended execute test cases for that specific product or functionality in a manual way.

This will allow the testers to learn the product or functionality and will help him to identify the best ways to implement the automation in a more efficient way (for example identifying reusable components).

* 1. Document the test cases

Automated test cases must at least contains a minimal description of what they will do an the primary source for test automation remains in Test Management tool.

* 1. Document the common resources

Common automated resources, functions must be clearly documented to allow an easy maintenance and a potential re-use by other team members and in other products

* 1. Link manual and automated tests

Automated test should be linked to the relevant manual tests in test management tool to get a big picture on automation test coverage.

* 1. Maintainability

One major challenge in automation is maintainability as the system being tested is prone to change often per iteration or release. The automated test cases must therefore be written in way they can easily maintain so that maintenance will not consume a significant part of the test automation effort. To do this, the cases must be early analysable, challengeable and expandable.

1. Automation development cycle

In order to make sure the test automation successful that it can strongly contribute in raising the quality of application against a modest budget. The automation activities should be embedded to a lifecycle comprising 5 phases. This can be applied at test cases level.

* 1. Feasibility Phase

As a part of this phase, we may have to conduct POC on the simple application with automation tool (Selenium) preferred for development of scripts to understand the complexity and document the observation.

* 1. Development Phase

Actual automation development happens in this phase on the basis of manual test cases which are finalized for automation in feasibility phase.

* 1. Deployment Phase

When test automation is compiling for the project, the automated scripts and documentation should be committed to the SI repository in GIT/Bit bucket for future reuse.

* 1. Execution Phase

In this phase, SI automation team to perform the execution of automated test cases in bulk and upload reports into test management tool against each test case. Any defect or bug introduced during execution to be logged in octane with reports.

* 1. Post Sprint Phase

Sprint will be reviewed to identify any lessons learned to improve future automation activities.

1. Framework Solution

The section describes the proposed automation framework solution to support the test automation.

* 1. Tools and Components of the Solution

The automation infrastructure comprises of below tools and components.