**Introduction:**

Test automation means using a software tool to run repeatable tests against the application to be tested. The automation software can enter test data into the System Under Test, compare expected and actual results and generate detailed test reports.

We will be using Lettuce framework which is Behaviour-driven development (BDD) automation tool which basically uses Python as scripting language.

**Advantages:**

* Frequent regression testing
* Virtually unlimited iterations of test case execution
* Support for Agile and extreme development methodologies
* Disciplined documentation of test cases
* Customized defect reporting
* Finding defects missed by manual testing

**Modules Covered:**

* Automatic Discovery
* Common Resource Learn
* IP View
* Resources
* User Accounts
* Alarms
* Notifications

**Prerequisites:**

* Lettuce Framework should be installed
* Everest should be Functionally Active

**Automation setup procedure:**

* Install Python 2.7
* Install “setuptools-0.6c11.win32-py2.7.exe”
* Execute command “easy\_install lettuce\_webdriver” in the path “C:\Python27\Scripts”
* Execute command “easy\_install nose” in the path “C:\Python27\Scripts”
* Copy IEDriverServer.exe inside Python folder and run it
* Insert Everest.feature and steps.py inside the path C:\Python27\features (create ‘features’ folder)
* Download ant from http://ant.apache.org and copy it in the root of C drive and follow http://ant.apache.org/manual/index.html to set path in environmental variables
* Copy “runtests.bat” inside “Python 2.7”
* Copy “build.xml” inside Python folder and create a folder by name “reports”
* Install ‘jdk’ and set its path in environmental variables
* Run C:\Python27>runtests.bat

**Files used in Automation :**

1. Feature file: It is a file in which the expected behaviour of the test will be mentioned. The statements used are just like normal English.

E.g., Scenario: Factorial of 0

Given I have the number 0

When I compute its factorial

Then I see the number 1

We can make use of *makeFeatures.py* to build feature files for Resource and IP View modules

1. Steps file: It will define the steps of the scenario, so that Lettuce can understand the behaviour description. Create the steps.py file which will contain python code describing the steps.
2. npoller.py: This is the test poller from which Everest will poll the learnt mimic devices.
3. Automation\_Configuration.csv: This file contains input test data to Everest poller. It also contains output data for verification.
4. makefeature.py: This is an automated python file which creates feature files.

**Automation test flow:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl no.** | **Duration** | **Scenarios** |  |
| 1 | First Hour ( Eg., 11:00 AM – 12:00 PM) | Following modules will be tested in this duration 1. Automatic Discovery 2. Common Resource Learn 3. IP View  4.Resources 5. User Accounts 6. Configurations – Add Notifications etc 7. Verification of Thresholds in Statistics page. |  |
| 2 | Second Hour | 1. Check Statistics for all resources 2. Polling should be enabled for all the Resources. 3. Statistics should be updated for  \* Latest Value \* Last Hour |  |
| 3 | Third Hour | \* Alarm testing - Memory Utilization High (Events table and Alarms page) \* Notification |
| 4 | Fourth Hour | \*Alarm testing - Device Down Alarm (Events table and in Alarms page) \*Notification |
| 5 | Fifth Hour | \*Alarm testing - CPU Utilization High Alarm( Events table and in Alarms page)  \*Notification |
| 6 | Sixth Hour | \*Alarm testing - Network Interface Down (Events table and in Alarms page)  \*Notification |
| 7 | Seventh Hour | \*Alarm testing - Disk Utilization High (Events table and in Alarms page)  \*Notification |
| 8 | Eighth Hour | Testing Overview : 1.Alarms Summary  2.Network Summary 3.System Summary |  |

**Scenarios covered:**

|  |  |  |
| --- | --- | --- |
| **Sl No.** | **Modules** | **Scenarios** |
| 1 | Automatic Discovery | \* Verification of AD page (All the fields in it) |
|  |  | \* Discovery of Nodes by Automatic discovery. |
|  |  | \*Verification of discovered node in Resource page. |
|  |  |  |
| 2 | Common Resource Learn | \* Verification of Common Resource Learn page (All the fields in it) |
|  |  | \* Discovery of through Common Resource Learn. |
|  |  | \*Verification of learnt node in Resource page. |
|  |  |  |
| 3 | IP View | \* Verification of IP view page. |
|  |  | \* Verification of Discovered node in IP View page, with all the columns in it. |
|  |  | \*Searching node by different filter options, like IP address, Host name, Operating system, Device name etc |
|  |  | \* Verification of Pagination bar. |
|  |  |  |
| 4 | Resources | \* Verification of Resource page. |
|  |  | \* Verification of Discovered node in Resource page |
|  |  | \*Verification of Statistics under Resources |
|  |  | \* Verification of Configuration and Events tables in Resource page. |
|  |  | \* Verification of Thresholds. |
|  |  |  |
| 5 | User Accounts | \* Verification of User Accounts page. |
|  |  | \*Validation of User accounts page. |
|  |  | \* Verification of Add, Edit, Delete, Enable and Disable functionalities |
|  |  | \*Verification of Logging in with added, deleted and disabled user account types |
|  |  | \*Verification of User privileges. |
|  |  |  |
| 6 | Alarms | \* Verification of Alarms page. |
|  |  | \*Verification of Alarms in alarms page as well as in Events table of Resource page. |
|  |  | \* Navigation from Resource in Alarms page to that of in Resource page. |
|  |  | \*Verification of Timescale. |
|  |  |  |
| 7 | Notifications | \* Verification of Notifier Alerts and Methods page. |
|  |  | \*Verify Add, Edit and Delete Notifier Alert and Notifier Method. |
|  |  | \* Verify Monitor message for a Notification sent on behalf of an alarm |

**Limitations:**

* Everest Reports validation
* Polling by Test poller (Static data)

**Automation Scope – Phase 2:**

* Integration between Sample.csv and makefeature.py file
* Automation scripts to generate feature files for Automatic discovery, CRL and User accounts
* Creating link between the screenshot to its scenario in Test report
* Highlighting Scenario name in Test reports
* Scheduling of Triggering Automation scripts for different modules.
* Verification of Overview page