Nagarro Autothon Event

28th September 2019

**Test Strategy Document**

*Movie Rental Application*

Prepared by:   
Testing- SQUAD

Nagarro Autothon

Table of Revisions

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Revised Date** | **Revised By** | **Revisions/Reason for Changes** |
| 0.1 | September 28, 2019 | Shrey Aggarwal | Initial Draft |
|  |  |  |  |

[**Introduction** 2](#_Toc20569748)

[Purpose 2](#_Toc20569749)

[Testing Scope 3](#_Toc20569750)

[In Scope 3](#_Toc20569751)

[**Testing Strategy** 5](#_Toc20569752)

[Testing Tools 6](#_Toc20569753)

[Automation configuration/Strategy 6](#_Toc20569754)

[**Test Execution Plan** 7](#_Toc20569755)

[Test Data 7](#_Toc20569756)

[Entry and Exit Criteria 7](#_Toc20569757)

[**Testing Deliverables** 8](#_Toc20569758)

# **Introduction**

## Purpose

This Movie Rental Application Test Report provides a summary of the approach used for automation test performed as outlined within this document.

The main purpose of maintaining the test Strategy document is:

* Provide a central artifact to govern the strategic approach of the test effort. Also, it defines the general approach to be employed when testing the software and when evaluating the results of that testing.
* Also, it gives a clear idea to the team how the testing process will be conducted in the project to ensure all the requirements defined in software requirements specification document have been met and the system functions are working as expected.

## Testing Scope

The scope of testing defines the tasks for the QA to know exact paths that are required to cover while performing testing activities.

### In Scope

The following testing is currently in scope for each solution set and describes the different testing types planned for the project

|  |  |  |
| --- | --- | --- |
| **Test Types** | **Method** | **Definition** |
| System | Automation | * The purpose of this test is to evaluate the system’s compliance with the specified requirements. * The purpose of this test is to ensure that new or newly modified components of the system do not affect functionality of non-modified components |
| API Testing |  | * Mainly concentrates on the business logic layer of the software architecture. * Use software to send calls to the API, get output, and note down the system's response. |

# **Automation Testing Strategy**

## Testing Framework

**TestNG**

TestNG is a testing framework inspired by JUnit and NUnit but introducing many new functionalities that make it more powerful and easier to use, such as [annotations](http://javarevisited.blogspot.sg/2012/06/junit4-annotations-test-examples-and.html#axzz56lq0jrxn), running your tests in arbitrarily big thread pools with various policies available (all methods in their own thread, one thread per test class, etc).

**MAVEN**

**Maven** is a powerful project management tool that is based on POM (project object model). It is **used** for projects build, dependency and documentation. In short terms we can tell **maven** is a tool that can be **used** for building and managing any Java-based project.

**Selenium 2 i.e. WebDriver –**

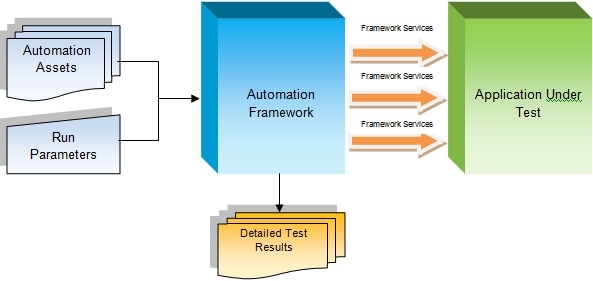
Selenium is probably the most popular tool for Java UI testing, which allows you to test your [JSP pages](http://www.java67.com/2018/02/5-free-servlet-jsp-and-jdbc-online-courses-for-java-developers.html) without launching them in a browser.

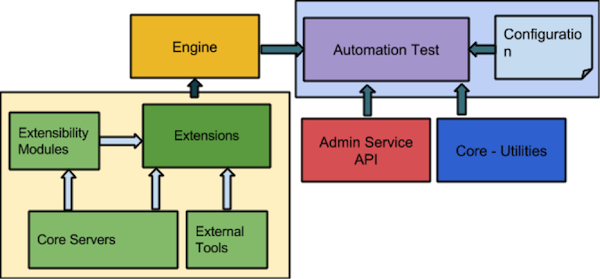
**POM** –

A Project Object Model or **POM** is the fundamental unit of work in Maven. It is an XML file that contains information about the project and configuration details used by Maven to build the project. It contains default values for most projects.

**Rest Assured –**

**REST Assured** is a Java library that provides a domain-specific language (DSL) for writing powerful, maintainable tests for RESTful APIs. In the following sections, I'll show you how to set up and configure **REST Assured**, write and run **REST Assured** tests, and apply some of its most powerful features.





## Testing Tools

|  |  |
| --- | --- |
| **Testing Type** | **Tool** |
| API testing | Rest Assured |
| Front end testing | Cypress or Selenium-Protractor |
| Test Case management | Testrail |
| Performance | Locust |

## Automation configuration/Strategy

This section describes which test should be automated and the ones which are not to be automated

|  |  |
| --- | --- |
| **Automated Test** | **Non Automated Test** |
| * Business critical paths – the features or user flows that if they fail, cause considerable damage to the business | * User experience tests for usability (tests that require a user to respond as to how easy the app is to use) |
| * Tests that need to be run against every build/release of the application, such as smoke test, sanity test and regression test | * Tests that need to be run ASAP. Usually, a new feature which is developed requires a quick feedback so testing it manually at first |
| * Tests that execute the same workflow but use different data for its inputs for each test run e.g. data-driven | * Tests that require ad hoc/random testing based on domain knowledge/expertise – Exploratory Testing. |
| * Tests that involve inputting large volumes of data, such as filling up very long forms | * Tests that require visual confirmation |
| * Tests that can be used for performance testing, like stress and load tests |  |
| * Tests that take a long time to perform and may need to be run during breaks or overnight. |  |

# **Test Execution Plan**

The following explains the strategy to execute the test plan:

## Test Data

Test data is basically is a specific set of data along with expected results for a particular test objective. Following list the types of test data that may be required in order to perform the testing activities.

|  |  |
| --- | --- |
| **Task** | **Type/Format** |
| API Testing | JSON |
| Front end testing | Text |
| Set up database conditions for test scenarios | TBD |

## Entry and Exit Criteria

For each test level in the development environment, testing team will ensure that all entry and exit criteria are met.

|  |  |  |  |
| --- | --- | --- | --- |
| Test Stage | Entry Criteria | Exit Criteria | Frequency |
| System Testing | * System test environment has been updated with a new build deployment * Integration of the module has been done successfully and passed the exit criteria of Integration testing * All the priority bugs have been fixed and closed * Test cases/ scripts are ready | * All the test cases have been executed * Bugs are recorded and make sure the priority bugs are fixed and closed * System is meeting all sorts of business and functional requirements | Every Release and every Sprit |
| API Test | * Successfully passing the exit criteria of the system testing * UAT environment has been set up | * Essential business process has been covered * No critical defect is left out * User Acceptance Testing close off meeting held * Bugs are recorded and make sure the priority bugs are fixed and closed | Every Release |

# **Testing Deliverables**

|  |  |
| --- | --- |
| **Deliverable** | **Description** |
| Test Strategy | A document which outlines the high level scope and strategy |
| Test Cases | The management of the test scenarios and test cases will be done using Testrail tool |
| Test Results | Documentation of executed test cases with corresponding results. |
| Defect Logging | Defects will be logged in testrail |
| Test  Execution Reports/Summary | Reporting will be provided on a per sprint basis to report the number of defects logged, severity and test progress |
| Test Scripts/ Data sets | The set of data/scripts used for testing the application |

## 