Table of content

[Introduction 1](#_Toc13899931)

[Objective 1](#_Toc13899932)

[Test Scope 2](#_Toc13899933)

[Environment Details 2](#_Toc13899934)

[Automation Framework Detail 3](#_Toc13899935)

[Config Files 3](#_Toc13899936)

[Control Structures 3](#_Toc13899937)

[Driver Scripts 4](#_Toc13899938)

[Framework Snapshots 4](#_Toc13899939)

[Generic Functions 4](#_Toc13899940)

[Modules 4](#_Toc13899941)

[Object Repository 4](#_Toc13899942)

[Results 4](#_Toc13899943)

[Framework Flexibility 5](#_Toc13899944)

[Future Framework Enhancement 6](#_Toc13899945)

[Additional Scenarios 6](#_Toc13899946)

**Test Plan**

# Introduction

This document refers the test plan for the “snaptravel.com” application. For now this will include only positive flow for booking but the test automation framework is robust and can easily be enhanced in future to incorporate additional test scenarios as well as additional framework level features.

**NOTE: However in current flow, while trying to book rooms, we are providing dummy bank payment details, which results into failure message in the end, however for reporting purpose, Booking details has been added in the current result as well.**

# Objective

This document will help to understand current testing scenario developed for web application- “snaptravel.com”, along with all possible future enhancement at testing and framework level.

This document **targets management level user** to understand test plan as well as **QA level users** to understand the scenarios and framework in brief along with possible enhancement.

# Test Scope

This document will cover current testing scope/scenarios covered along with additional scenarios that can be covered in the future. This will also provides Framework level features which are present and which can be enhanced in future if required.

For now, **only one Booking negative flow** **- booking failure due to wrong payment detail - has been covered in the script,** along with possible testing scenarios which can be incorporated in future.

**In Scope** – Functional Testing

**Out of Scope** – DB testing, UAT, Integration Testing, Non Functional testing like – Load testing, Performance testing

# Environment Details

Below is the details regarding this framework, need to take care of prerequisite for running this test scripts. It also includes few details like application environment that we have used as part of current scenarios.

* **Pre-requisite :** Testing environment stet up – Need to install UFT software

Note: You can download latest version of UFT from below link:

<https://www.microfocus.com/en-us/products/unified-functional-automated-testing/download>

* **Automation tool used** – UFT Automation tool
* **Application Environment detail** –

For current scenario, below application environment is used, however this value can be parameterized and can be changed as per requirement in future.

URL: <https://www.snaptravel.com/search?encrypted_user_id=5xqebwRCiWusH08KS2yJKA>

* **Required skill set to utilize this automation code:**

No training is required to execute the automation script.

Basic UFT/VB script knowledge is required for automation coding

# Automation Framework Detail

The whole automation framework consists of below folders:

## Config Files

* **EnvironmentVariables.txt**

This includes Environment variables, which are loaded at run time from driver script.

## Control Structures

* **Execution\_Sheet.xls**

This includes all available testing scenarios along with “ExecutionFlag” field, which decides whether user wanted to execute that scenario or skip that scenario.

This also includes additional details like Application Environment, User, Sprint number, Cycle number – all these details for reporting purpose

* **Control\_Sheet.xls**

This includes the flow of the application as per specific test scenario

* **InputData\_Sheet.xls**

This includes input data used as per specific scenario

Note: We can have data input from DB, Access etc.

## Driver Scripts

This is the script from where whole execution will start. From this script user will start execution and actions will be called as per flow present in current script

This includes reading data from Executional File, get test case wise flow details from Control File, get input data from Input File and iterate through Actions as per the test case flow defiled in the script.

## Framework Snapshots

This includes Snapshot taken incase of any failure in automation execution.

## Generic Functions

This includes all reusable functions which can be used later on from Driver or Module wise actions.

This also includes, Recovery Scenario - which help to recover automation script to handle in case of any unpredictable run time error.

## Modules

This includes different reusable Actions, which makes this framework very Robust and easy to maintain and debug for automation tester

## Object Repository

* **Shared\_OR.tsr**

This includes a shared Object Repository, which is used by all actions throughout the framework

## Results

* Result\_TestCaseWise\_<Timestam>.xls

This includes test case wise result – Pass or Fail, very high level execution details just to understand how many total scenarios executed, which are passed and which are failed

* Result\_StepWise\_<Timestam>..xls

This includes step case wise result – Pass or Fail, in detail field level reporting – which data used as input, which validation done in scripting, failure if any etc.

This result sheet will mainly be used by automation tester for debugging purpose.

# Framework Flexibility

* Current framework is very robust and can **easily be enhanced** to accommodate additional testing scenarios in future with minimum efforts.
* User can easily add new framework level features like, **Execution recording**, **Email Trigger after end of Execution etc.** in future if required.
* **All input data is provided from an** **input excel**, which makes it easy for tester to change any booking details if required.
* Current framework has been **divided in different modules to increase its flexibility**.
* **Modular framework** will help testing team to easily add additional scenarios as well as it makes debugging task easy as well.
* Framework has “**Iteration**” – feature, which allows user to run same test scenarios with same test data multiple times. **ex.- generate transaction no using same test data**
* Framework has “**RowIndex**” – feature, which allows user to run same test scenarios with different sets of test data **ex. – Run Login Script with different user name and password, Book ticket for different users.**
* This framework also has “**Recovery Scenario”** – which help to recover automation script, **to handle in case of any unpredictable run time error.**

# Future Framework Enhancement

* **Add mail feature** – At the end of each execution, an Email will be trigger to inform all required people that automation execution completed along with execution result attached in the mail
* **Add execution recording feature - It will record whole execution process and will attach that recording in result excel**
* **Add failure snap shot –**  At the time of Script level, Application Level or run time failure, it will take screen shot and will attach it in result excel
* **Send execution result in HTML format via mail** – We can format result and prepare HTML file for better understanding and user experience
* **Add .Net utility tool (.exe file) to trigger execution –** To make Manual tester work more easy, we can develop a interface which will trigger the automation execution without even need of opening UFT manually by Manual tester
* **Change in result excel format –** Currently we have simple result excel with without any formatting, but we can easily format result excel in future with min efforts.

# Additional Scenarios

Below are few of the scenarios identified on high level, which can be added if required.

|  |  |
| --- | --- |
| **Scenario Name** | **Description** |
| Test\_LaunchURL\_Positive | URL is launched and user is Navigated to Home screen successfully |
| Test\_LaunchURL\_Negative | URL is launched and but user is NOT Navigated to Home screen |
| Test\_Booking\_Negative | Enter all Booking details and due to wrong payment details – Booking failed |
| Test\_Booking\_Positive | Enter all Booking details – Booking is done successfully |
| Test\_UI\_MandatoryFields | – Validate What all fields are mandatory in different screens |
| Test\_UI\_OptionalFields | Validate What all fields are optional in different screens |
| Test\_UI\_Fields\_DBType\_Validation | Validate What all fields wise DB types restriction at UI level |
| Test\_UI\_Fields\_DBSize\_Validation | Validate What all fields wise DB field size restriction at UI level |
| Integration Scenario | Test Integrated system |
| Performance Testing | Test Performance of the Application |
| Load Testing | Test User thresh-hold for current system |
|  |  |