|  |  |  |
| --- | --- | --- |
|  |  | SMART TRAVEL  GUIDE |
|  |  |  |
| SOFTWARE  TESTING |  | PROJECT REPORT |

SMART TRAVEL GUIDE

**P R O J E C T R E P O R T**

**COURSE CODE: SWE2005**

**COURSE TITLE: SOFTWARE TESTING SLOT: C2+TC2**

|  |  |
| --- | --- |
| **NAME** | **REGNO** |
| G. SAI VARUN | 15MIS0385 |

**PROF. IYAPPARAJA.M**

Table of Contents

1. [PURPOSE 4](#_bookmark0)
2. [APPLICATION OVERVIEW 4](#_bookmark1)
3. [TESTING SCOPE 4](#_bookmark2)
   1. [IN SCOPE 4](#_bookmark3)
   2. [OUT SCOPE 4](#_bookmark4)
   3. [ITEMS NOT TESTED 5](#_bookmark5)
4. [METRICS 5](#_bookmark6)
5. [Types of testing performed: 7](#_bookmark7)
6. [DECISION TABLE TESTING 7](#_bookmark8)
7. [BOUNDARY VALUE ANALYSIS(BVA) 7](#_bookmark9)
8. [STATE TRANSITION TESTING 8](#_bookmark10)
9. [TEST ENVIRONMENT AND TOOLS 8](#_bookmark11)
10. [TEST CASES 8](#_bookmark12)
11. [SCREENSHOTS 11](#_bookmark13)
12. [TESTING TOOLS 12](#_bookmark14)
13. [SELENIUM 12](#_bookmark15)
14. [GRAPHICAL USER INTERFACE TESTING 16](#_bookmark16)
15. [INTERFACE 18](#_bookmark17)
16. [EXIT CRITERIA 21](#_bookmark18)
17. [CONCLUSION 21](#_bookmark19)

# PURPOSE:

In this project we have developed and executed the test cases for various modules using black box testing to the “**SMART TRAVEL GUIDE FOR VITIANS”.** The test cases have been executed using the selenium test case tool.

# APPLICATION OVERVIEW:

“**SMART TRAVEL GUIDE FOR VITIANS”** is a web based booking application. Tickets for various trips can be booked using the online facilities. Real time passenger information is received from a ‘Central repository system’, which will be referred before booking is confirmed. There are several modules like Login, Booking, Add user and Add category which are integrated to fulfill the purpose.

# TESTING SCOPE:

## IN SCOPE:

Functional testing for the following modules are in scope of testing

* + - Login
    - Booking
    - Add user

## OUT SCOPE:

Performance Testing was not done for this application.

## ITEMS NOT TESTED:

Verification of connectivity with the third party system ‘Central repository system’ was not tested, as the connectivity could not be established due to some technical limitations. This can be verified during UAT (User Acceptance Testing) where the connectivity is available or can be established.

# METRICS:

* 1. **No of test cases planned vs executed**
  2. **No of test cases passed/failed**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test cases** | **Test cases** | **TCs** | **Tcs** |
| **planned** | **executed** | **Pass** | **Failed** |
| 46 | 46 | 44 | 02 |

**TEST CASES PASS VS FAIL**

FAIL 2%

PASS

FAIL

PASS

98%

* 1. **No of defects identified and their status & severity**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Critical** | **Major** | **Medium** | **Total** |
| **Closed** | 0 | 0 | 0 | 0 |
| **Open** | 0 | 2 | 0 | 5 |
|  | | | | 05 |

DEFECTS SEVERITY AND STATUS

2.5

2

1.5

1

0.5

0

DEFECTS SEVERITY & STATUS

DEFECTS SEVERITY & STATUS

0

2

0

MINOR

MAJOR

CRTICAL

* 1. **Defects distribution – module wise**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Login** | **Booking** | **Add user** | **Total** |
| **Critical** | 0 | 0 | 0 | **0** |
| **Major** | 0 | 2 | 0 | **2** |
| **Medium** | 0 | 0 | 0 | **0** |
| **Total-->** | **0** | **2** | **0** | **2** |

**DEFECTS DISTRIBUTION MODULE - WISE**

LOGIN BOOKING ADD USER

# Types of testing performed:

## DECISION TABLE TESTING

Decision Table Testing is a good way to deal with a combination of inputs, which produce different results. It helps reduce test effort in verifying each and every combinations of test data, at the same time ensuring complete coverage

## BOUNDARY VALUE ANALYSIS (BVA)

Boundary value analysis is another black box test design technique and it is used to find the errors at boundaries of input domain rather than finding those errors in the center of input.

Equivalence Partitioning and Boundary value analysis are linked to each other and can be used together at all levels of testing. Based on the edges of the equivalence classes, test cases can then be derived.

## STATE TRANSITION TESTING

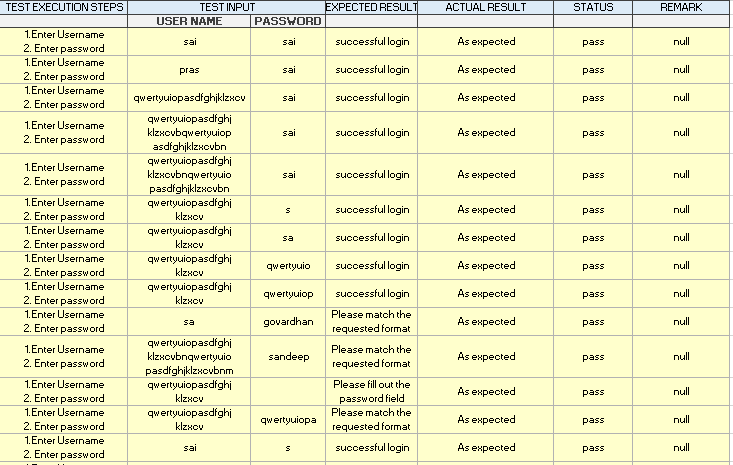
State Transition testing, a black box testing technique, in which outputs are triggered by changes to the input conditions or changes to 'state' of the system. In other words, tests are designed to execute valid and invalid state transitions

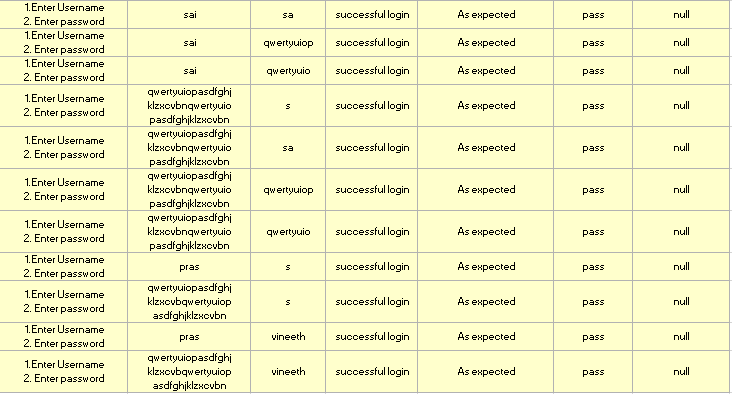
# TEST ENVIRONMENT AND TOOLS

|  |  |
| --- | --- |
| APPLICATION URL | localhost/Travel, localhost/Travel/Admin |
| APPS SERVER | **XAMPP** |
| DATABASE | **MY SQL,APACHE** |

# TEST CASES:

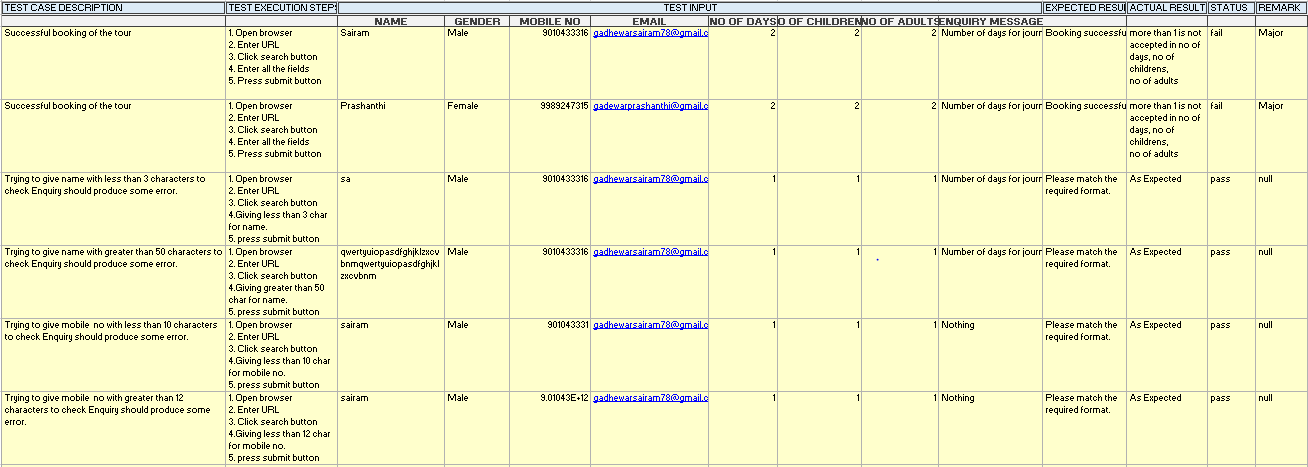
**FOR LOGIN MODULE:**

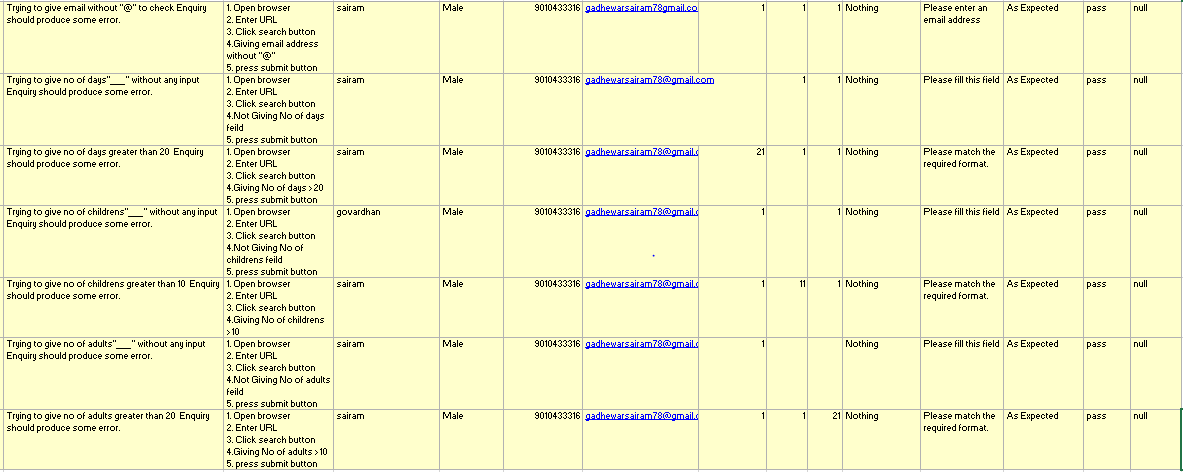




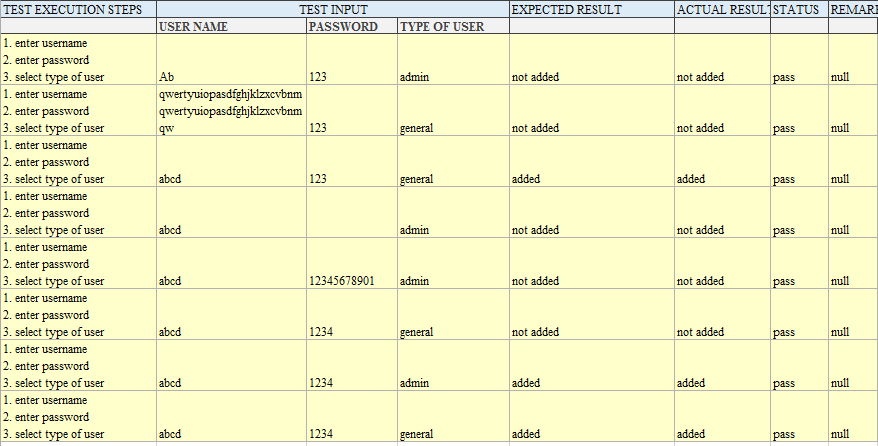


**FOR BOOKING MODULE:**





**FOR ADD USER MODULE:**



# SCREENSHOTS:

* [info] Playing test case Untitled
* [info] Test case passed
* [info] Playing test case Untitled 2
* [info] Executing: |open | /Travel/detail.php?pid=1 | |
* [info] Executing: |clickAndWait | link=Category | |
* [info] Executing: |waitForPopUp | \_self | 30000 |
* [info] Executing: |clickAndWait | link=Family Tours | |
* [info] Executing: |clickAndWait | link=Family Tours | |
* [info] Executing: |clickAndWait | css=font | |
* [info] Executing: |clickAndWait | css=font | |
* [info] Executing: |clickAndWait | name=sbmt | |
* [info] Executing: |type | name=t1 | ram |
* [info] Executing: |type | name=t2 | 2333210 |
* [info] Executing: |type | name=t3 | [raohaojn@jxhja.com](mailto:raohaojn@jxhja.com) |
* [info] Executing: |type | name=t4 | 1 |
* [info] Executing: |click | name=sbmt | |
* [info] Executing: |type | name=t2 | 2333210000 |
* [info] Executing: |type | name=t4 | 2 |
* [info] Executing: |type | name=t4 | 9 |
* [info] Executing: |type | name=t4 | 1 |
* [info] Executing: |type | name=t5 | 5 |
* [info] Executing: |type | name=t6 | 1 |
* [info] Executing: |type | name=t7 | asd |
* [info] Executing: |type | name=t4 | 6 |
* [info] Executing: |click | name=sbmt | |
* [info] Executing: |type | name=t4 | 1 |
* [info] Executing: |click | name=sbmt | |
* [info] Executing: |type | name=t5 | 2 |
* [info] Executing: |click | name=sbmt | |
* [info] Executing: |type | name=t5 | 1 |
* [info] Executing: |clickAndWait | name=sbmt | |
* [error] interrupted
* [info] Test case failed
* [info] Test suite completed: 2 played, 1 failed

# TESTING TOOLS

1. SELENIUM

INTRODUCTION:

* + Selenium is an automation test framework or an API that works with Java, C#, Perl, Ruby, Python and Groovy programming languages to automate the browser activities. Selenium is a portable software-testing framework for web applications.
  + Selenium provides a record/playback tool for authoring tests without learning a test scripting language (Selenium IDE).
  + Selenium integrated development environment, acronym, as Selenium IDE is an automated testing tool that is released as a Firefox plug-in.
  + The tool is laid on a record and playback fundamental and allows editing of the recorded scripts.
  + Selenium is a free (open source) automated testing suite for web applications across different browsers and platforms.
  + The name Selenium comes from a joke made by Huggins in an email, mocking a competitor named Mercury[,](https://en.wikipedia.org/wiki/Mercury_Interactive) saying that you can cure mercury poisoning by taking selenium supplements. The others that received the email took the name and ran with it.

**SELENIUM TEST PACKAGE:**

Selenium is a package of various test components, which consists of the following three major tools. Each one has a specific role in aiding the development of test automation for a Web application.

1. Selenium **IDE** – A Firefox extension to record test cases and suites.
2. Selenium **RC** – Used to run tests on different browsers and systems.
3. Selenium **Grid** – Runs multiple instances of Selenium RC at once.
4. **Qualitia and Tellurium** – A wrapper for the Selenium engine.
5. **Ranorex IDE-** An user interface tester used to test the GUI of the website.

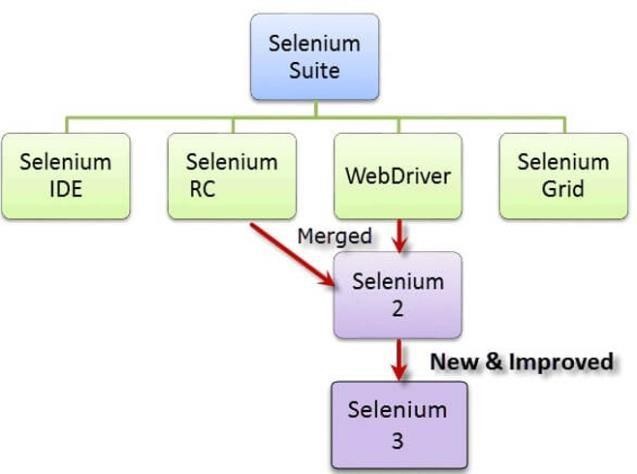


Fig1. Various versions of selenium

**SELENIUM IDE:**

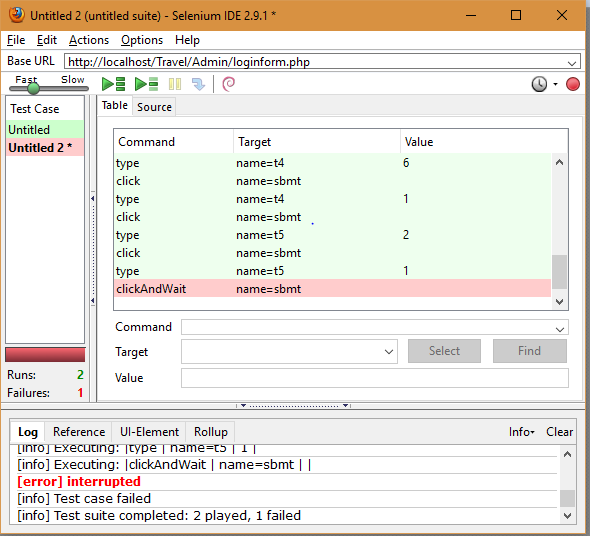
Selenium is not an automation tool like HP-UFT (QTP) as it doesn’t give you any UI to perform some automation activity. While, selenium has its first version called selenium IDE which has a user Interface to record and play an activity without knowing any programming or scripting language. It used to work as a Firefox browser plugin but that has very limited scope of achieving test automation objective.

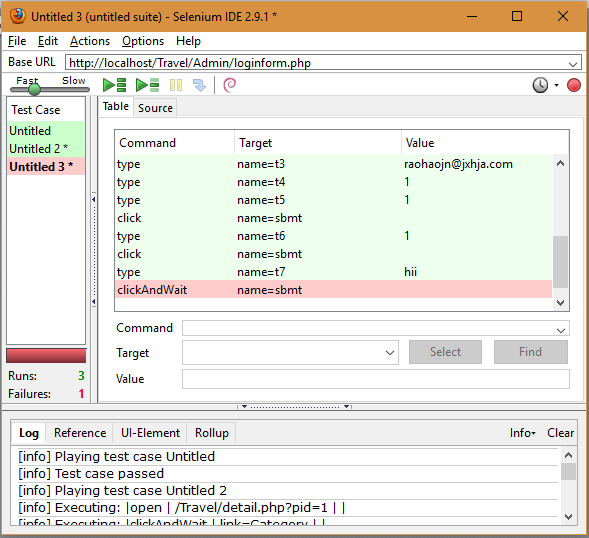
**SELENIUM RC:**

Later, Selenium came with Selenium Remote Control (RC) or selenium 1.0 that allows tester to write automation script for web application UI tests in any programming language against any HTTP website using any mainstream JavaScript-enabled browser. In order to work with selenium RC selenium provided a server and it was required to run the server to start the execution of automation scripts.

**SELENIUM WEBDRIVER:**

Selenium provided the more advanced version and called it selenium 2.0, Selenium WebDriver, or WebDriver. It has more clearly and organized APIs, improved, and added functionality, which make it more popular choice as automation tool for web applications. Therefore, Web driver is the only selenium (not IDE and RC) which is running in the market of automation.





2. GRAPHICAL USER INTERFACE TESTING **(for Ranorex website): Introduction of the tool:**

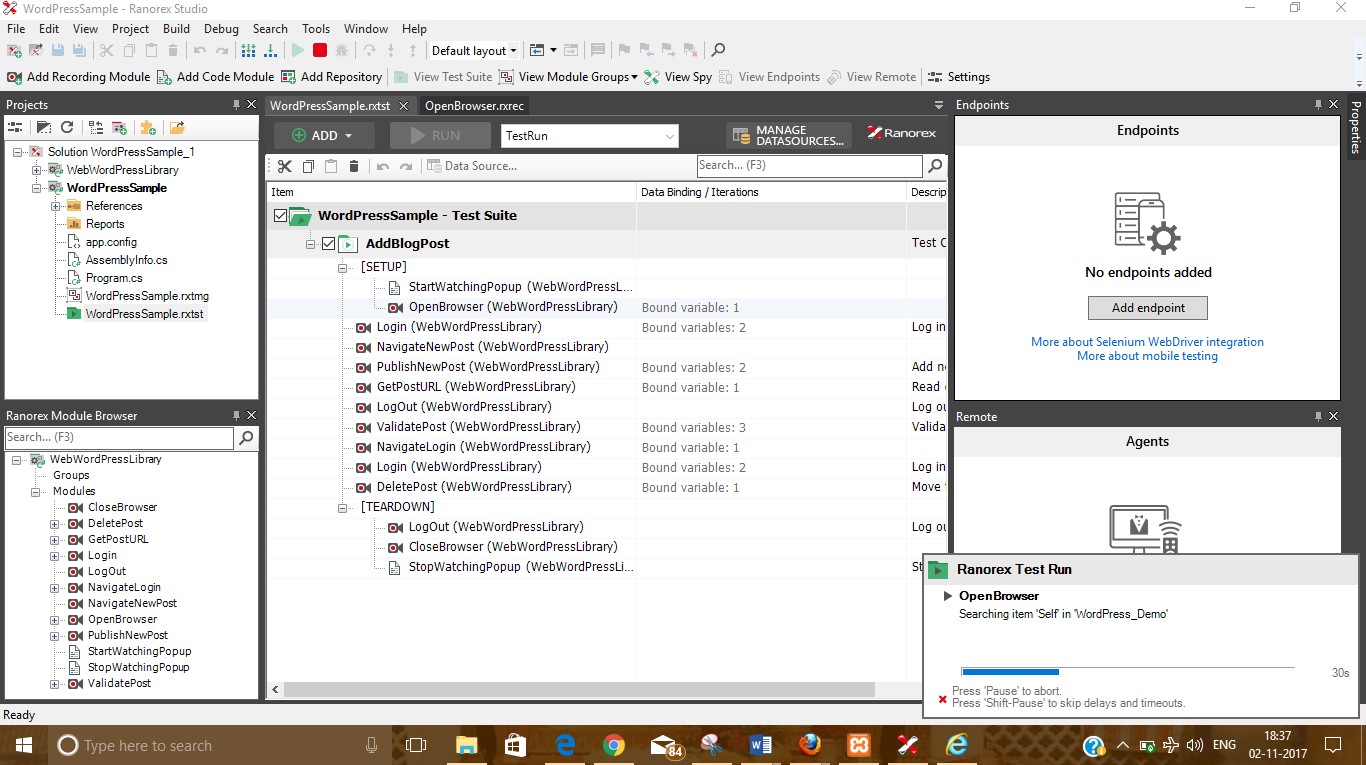
Ranorex is a GUI test automation framework for testing of desktop, web-based and mobile applications. Ranorex GmbH, a software development company, provides Ranorex. Ranorex does not have a scripting language of its own, instead using standard programming languages such as C# and VB.NET as a base.

### MAIN FEATURES:

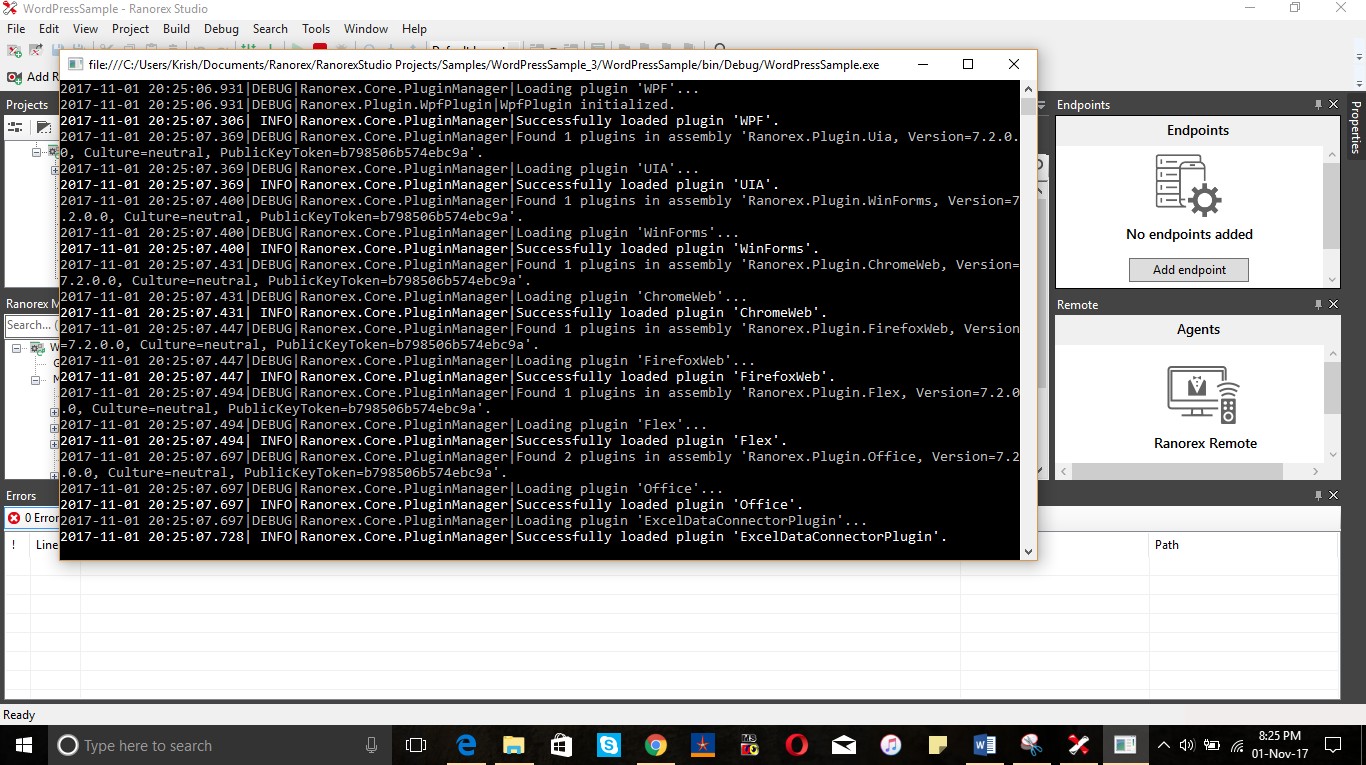
**GUI Object Recognition**: RanoreXPath for identification/filtering of all types of GUI elements. Ranorex Spy provides the mapping information of [GUI](https://en.wikipedia.org/wiki/GUI) elements to their RanoreXPath expression

**Object-based Capture/Replay functionality:** (called Ranorex Recorder), which provides maintainable recordings via an actions table editor transforms recorded actions into C# and VB.NET code and generates detailed report files for quick error detection.

**Flexible Test Automation Interface:** Test Automation within specific test environments is possibly.]Test suites with Ranorex results in .EXE files for simple integration into existing environments such as test management tools, continuous integration processes or batch execution scenarios. They can be easily run by launching the .EXE file from the command line.

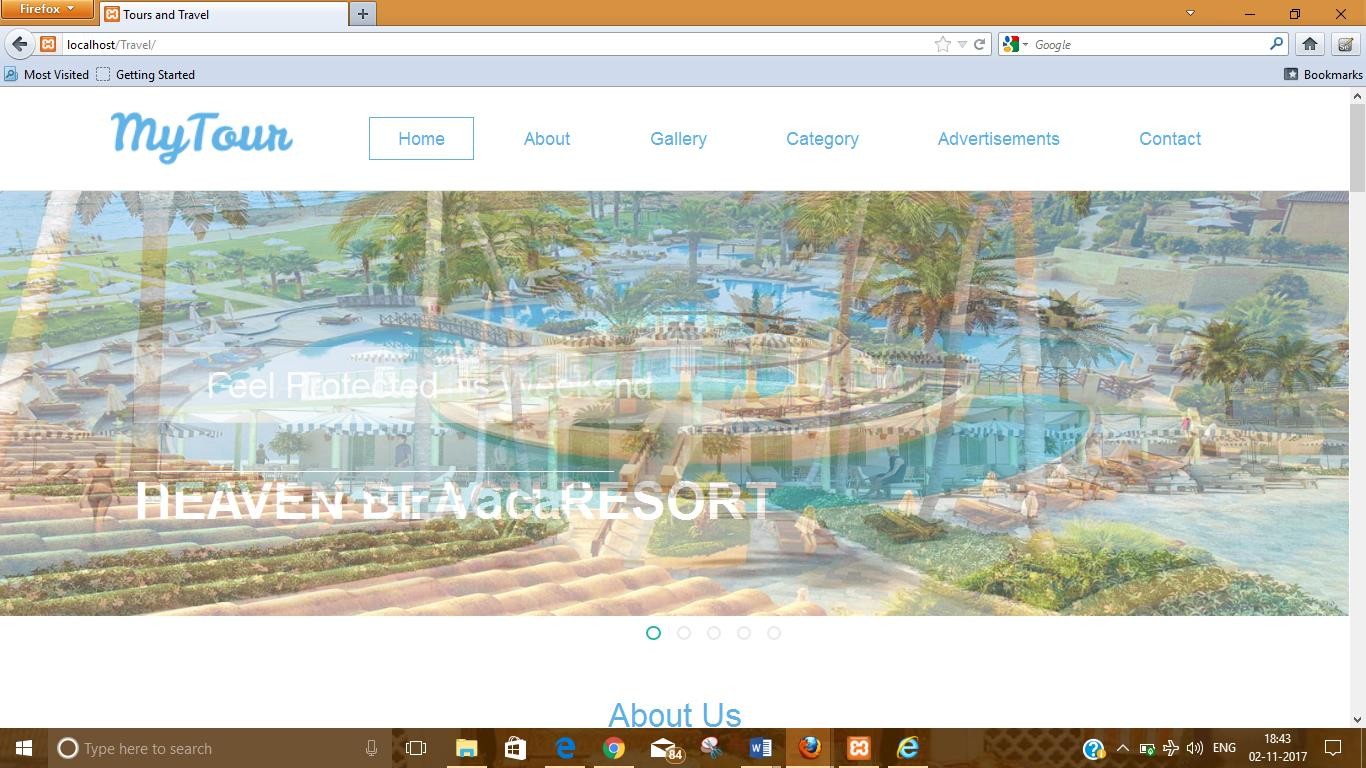
* Test project management
* Intuitive code editor and syntax highlighting
* Code completion
* Debugging and watch monitor
* Project files are compatible with Visual Studio
* Full integration of all Ranorex test automation tools
  + Ranorex Test Recorder
  + Ranorex Object Repository Ranorex Spy

**Running of website:**



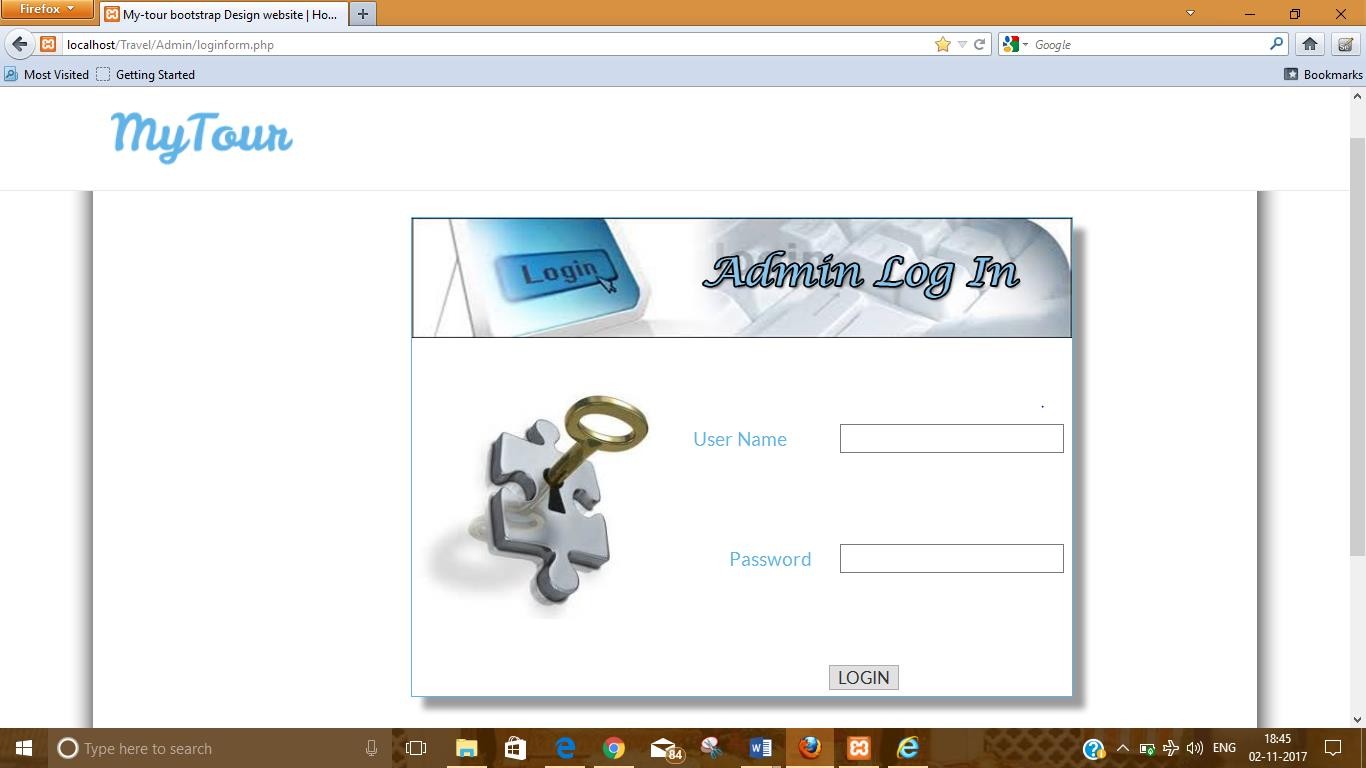
# INTERFACE

**HOME PAGE FOR END USERS:**

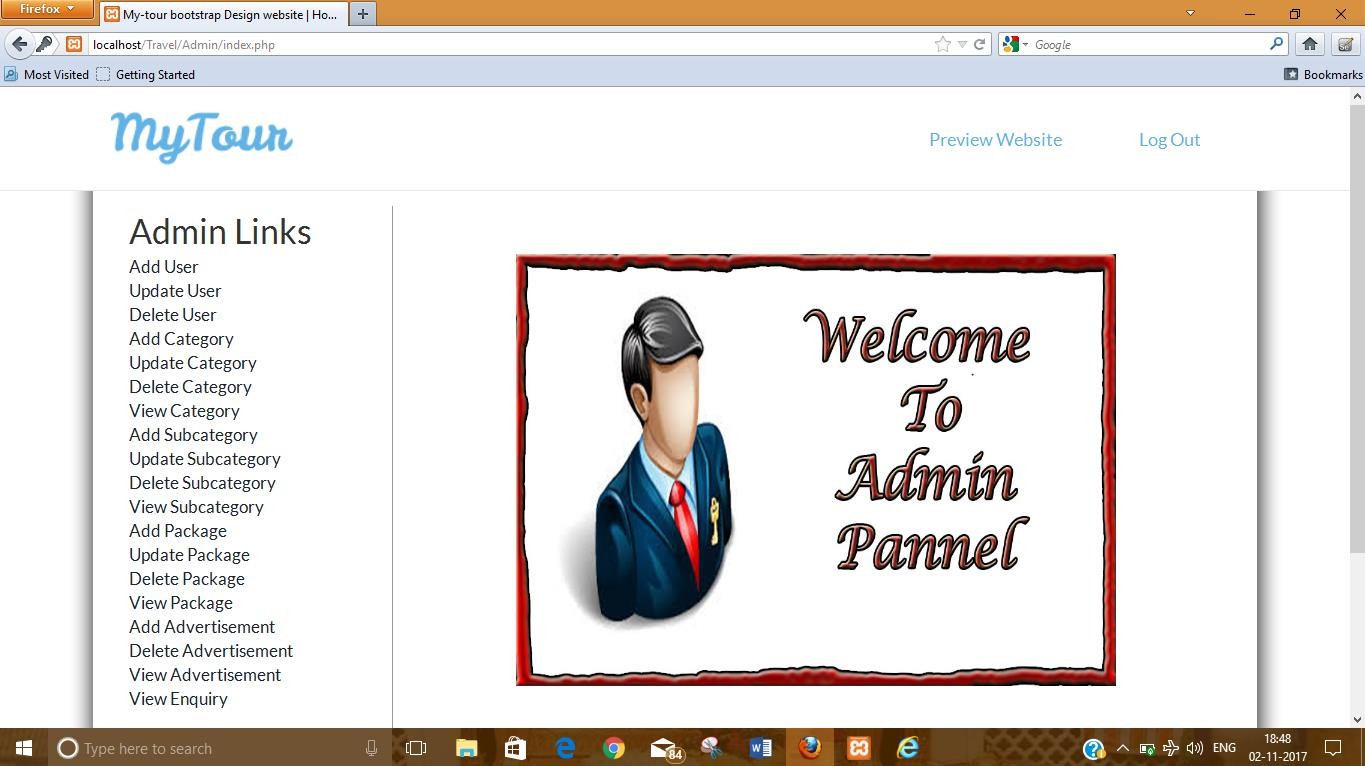


**HOMEPAGE FOR ADMINS**

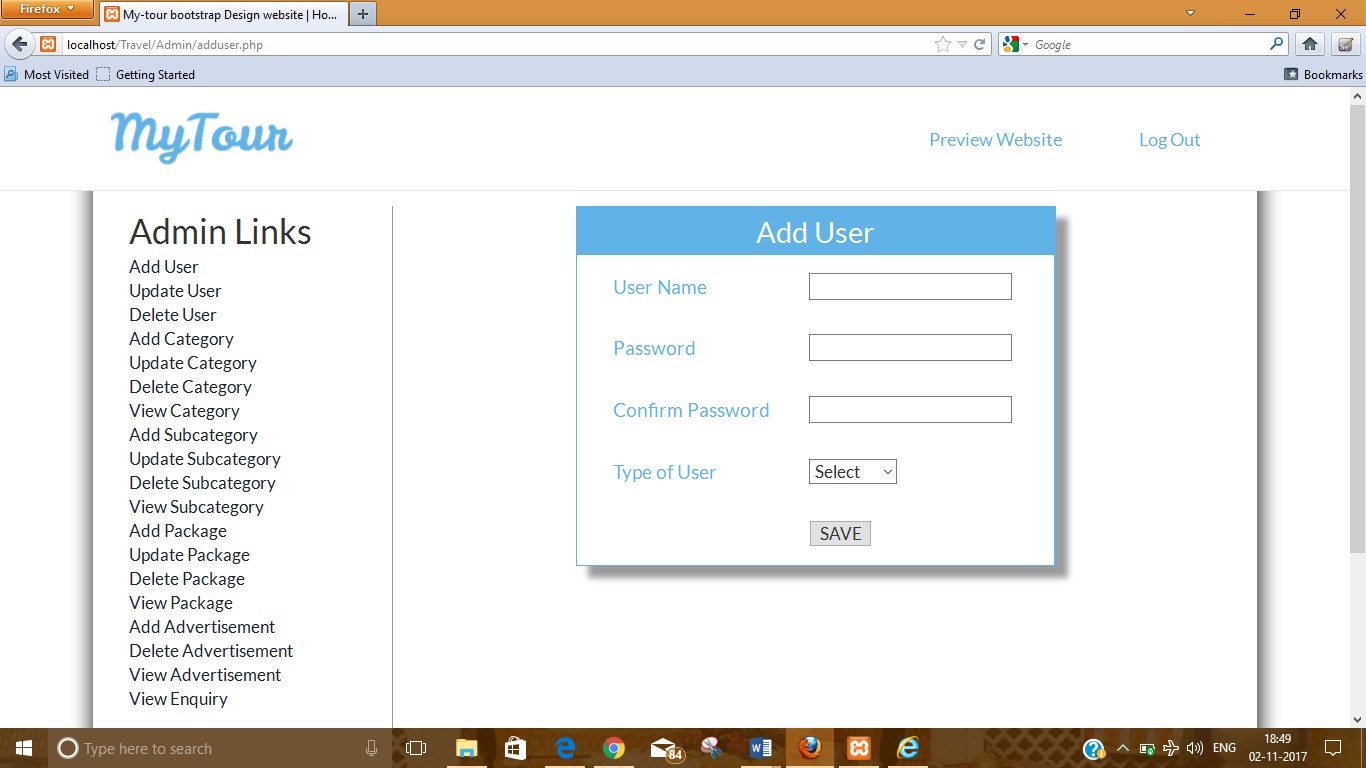
1. **LOGIN PAGE:**



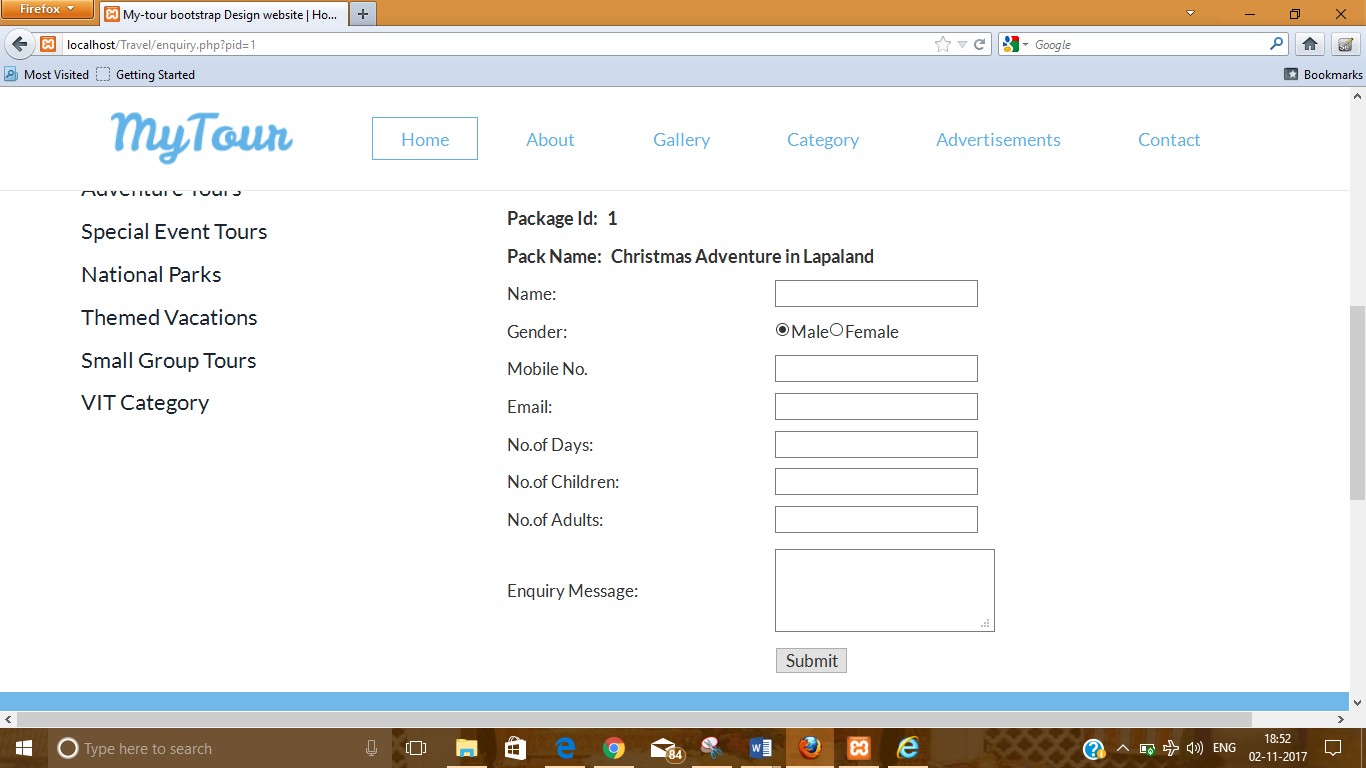
1. **HOME PAGE FOR ADMIN:**



**ADDING USER:**



**BOOKING:**



# EXIT CRITERIA:

* All test cases should be executed – **Yes**
* All defects in Critical, Major, Medium severity should be verified and closed – **No**
* Any open defects in trivial severity – **Action plan prepared with expected dates of closure.**

# CONCLUSION:

As the Exit criteria was met and satisfied as mentioned above, this application is suggested to ‘Go Live’ by the Testing team. Appropriate User/Business acceptance testing should be performed before ‘Go Live’.