Test Execution Document

For

Online Cab Booking System

GoSafe

Version 1.1 approved

**Prepared by: Jagan Mohan Reddy Guda**

**Kodamasimham Aakash**

**Karanam Rakesh**

**Koushik Andhavarapu**

**Keshaowar Shivraj**

**NIIT University**

Table of Contents

|  |  |
| --- | --- |
| **Name** | **Page number** |
| **1.Introduction** | **3** |
| * 1. **Purpose** | **3** |
| * 1. **Testing Process** | **3** |
| **2 Unit Testing** | **3** |
| **2.1 Introduction** | **3** |
| **2.2 Register** | **4** |
| **2.3 Login** | **8** |
| **2.4 validuser** | **11** |
| **2.5 loginUser** | **12** |
| **2.6 dashboard** | **13** |
| **2.7 getCars** | **17** |
| **2.8 shared** | **18** |
| **2.9 getSharedCars** | **22** |
| **2.10 getCarnumber** | **23** |
| **2.11 booking** | **24** |
| **2.12 bookAvaiblecar** | **27** |

# Introduction

## Purpose

**Text Execution Document** is prepared for an Online Cab Booking System taken under software engineering as a project work is the solution for the difficulties of people getting cabs at the time of going home or any occasions and vacations being named **Gosafe.** This document provides complete information of test cases applied on the prepared project and the actions it showed on applying. Testing of the web application has been done in two different ways.

* Unit Testing
* Integration Testing

## Testing Process

Unit Testing: The whole project will be of many units. Like for each page view there will be a coding unit. All the units are separately tested. Each unit test cases are divided into black box test cases and white box test cases. Black box and White box are just types of deriving test cases under Unit Testing. Each unit upon white and black box cases are explained below.

Integration Testing: Complete project divided into integral parts which are all the sum of units. Hence before integration testing the unit testing should have been completed. And each integral part again undergoes black box and white box test cases which are only for deriving test cases. Theses test cases are also explained below.

# Unit Testing

## Introduction

Whole web application code is divided into small units which can be tested under blocks. All the test cases are derived into black and white box testing methods. Each unit will have two sections namely black and white box testing. Each section will have number of test cases and be explained clearly.

## Unit 1 – Register

**Section – 1 Black Box Test Cases**

|  |  |  |
| --- | --- | --- |
| Test Case | Function | Type |
| Test Case 1 | Register | Appropriate Inputs(All fields) |
| Test Case 2 | Register | Invalid Inputs (Discrete fields) |
| Test Case 3 | Register | Invalid Inputs (All fields) |
| Test Case 4 | Register | Special/ Un allowed characters. (All fields) |

**Test Case – 1**

**Description:** The user has to register for the first time in order to obtain the details from the user for the sake of validation and for providing the services without any discrepancies. For also keeping up the application away from intruders. The following details are taken from the user and stored in database: full name, mail id, password, phone number. The user once registered can’t register another time with the same details.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly.

**Test Inputs:** As valid arguments for register unit are given as inputs in this first test case which are respectively: Expected Inputs

Full name: alphabets a-z, A-Z with no special characters.

Mail id: a-z, 0-9, @

Password: all alphabets and special characters are allowed

Phone number: 0-9

**Test Results:**

Test inputs all have **passed** for correct inputs applied. All the functions are working properly on correct inputs. No anomalies.

**Test Case – 2**

**Description:** The user has to register for the first time in order to obtain the details from the user for the sake of validation and for providing the services without any discrepancies. For also keeping up the application away from intruders. The following details are taken from the user and stored in database: full name, mail id, password, phone number. The user once registered can’t register another time with the same details.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly.

**Test Inputs:** As valid arguments for register unit are given as inputs in this first test case which are respectively: Expected Inputs are

Full name: alphabets a-z, A-Z with no special characters.

Mail id: a-z, 0-9, @

Password: all alphabets and special characters are allowed

Phone number: 0-9

\*For this test case all invalid inputs are used for discrete fields that is in either of the fields in register function

For Full name: if the user tries to input other than expected inputs it doesn’t allow to surpass that page unless appropriate inputs are given.

Mail id must contain @ sign in the first version. In later versions the mail verification will also be implied.

Password has no restrictions so this works fine.

Phone number will not intake anything other than numbers which is compulsory.

**Test Results:**

Test inputs all will not be accepted orderly can’t register which makes the test case **pass**. Respective fields doesn’t work with the wrong inputs which is desired and can be stated working properly.

**Test Case – 3**

**Description:** The user has to register for the first time in order to obtain the details from the user for the sake of validation and for providing the services without any discrepancies. For also keeping up the application away from intruders. The following details are taken from the user and stored in database: full name, mail id, password, phone number. The user once registered can’t register another time with the same details.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly.

**Test Inputs:** As valid arguments for register unit are given as inputs in this first test case which are respectively: Expected Inputs

Full name: alphabets a-z, A-Z with no special characters.

Mail id: a-z, 0-9, @

Password: all alphabets and special characters are allowed

Phone number: 0-9

\*In this test case all invalid inputs are used for all fields in register function.

For Full name: if the user tries to input other than expected inputs it doesn’t allow to surpass that page unless appropriate inputs are given.

Mail id must contain @ sign in the first version. In later versions the mail verification will also be implied.

Password has no restrictions so this works fine.

Phone number will not intake anything other than numbers which is compulsory

**Test Results:**

Test inputs all in all fields at least one anomaly will not be accepted which ensures **passed** according to the inputs applied. All the functions are working properly on correct inputs. No anomalies

**Test Case – 4**

**Description:** The user has to register for the first time in order to obtain the details from the user for the sake of validation and for providing the services without any discrepancies. For also keeping up the application away from intruders. The following details are taken from the user and stored in database: full name, mail id, password, phone number. The user once registered can’t register another time with the same details.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly.

**Test Inputs:** As valid arguments for register unit are given as inputs in this first test case which are respectively: Expected Inputs

Full name: alphabets a-z, A-Z with no special characters.

Mail id: a-z, 0-9, @

Password: all alphabets and special characters are allowed

Phone number: 0-9

Inputs other than above mentioned are used then the fields should restrict the user from applying them and can’t register himself.

**Test Results:**

Test inputs all have **passed** for correct inputs applied. All the functions are working properly on correct inputs. No anomalies.

**Section 2 – White Box Test Cases**

**Test Case:**

**Description:** The user has to register for the first time in order to obtain the details from the user for the sake of validation and for providing the services without any discrepancies. For also keeping up the application away from intruders. The following details are taken from the user and stored in database: full name, mail id, password, phone number. The user once registered can’t register another time with the same details.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly.

**Test Inputs:** As valid arguments for register unit are given as inputs in this first test case which are respectively: Expected Inputs

Full name: alphabets a-z, A-Z with no special characters.

Mail id: a-z, 0-9, @

Password: all alphabets and special characters are allowed

Phone number: 0-9

**Resolved BUG:**

If input type is ‘text’ all the characters will be accepted but for mail id field the input type should be ‘email’ otherwise email restrictions will not function. This anomaly has been rectified by using appropriate input types.

**Test Results:**

After the bug implementation the test case is **passed** for correct inputs applied. All the functions are working properly on correct inputs. No anomalies.

## Unit 2 – Log in

**Section – 1 Black Box Test Cases**

|  |  |  |
| --- | --- | --- |
| Test Case | Function | Type |
| Test Case 1 | Register | Appropriate Inputs(All fields) |
| Test Case 2 | Register | Invalid Inputs (Discrete fields) |
| Test Case 3 | Register | Invalid Inputs (All fields) |
| Test Case 4 | Register | Special/ Un allowed characters. (All fields) |

**Test Case – 1**

**Description:** The user can Log in only if he was registered before. For logging in the user gives the inputs mail id and password set while registering. The inputs if matched with the stored details in database the user will be accessed to log in.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those browsers all the input fields are to be tested accordingly.

**Test Inputs:** As valid arguments for register unit are given as inputs in this first test case which are respectively: Expected Inputs

Mail id: a-z, 0-9, @

Password: all alphabets and special characters are allowed

**Test Results:**

Test inputs all have **passed** for correct inputs applied. All the functions are working properly on correct inputs. No anomalies.

**Test Case – 2**

**Description:** The user can Log in only if he was registered before. For logging in the user gives the inputs mail id and password set while registering. The inputs if matched with the stored details in database the user will be accessed to log in.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those browsers all the input fields are to be tested accordingly.

**Test Inputs:** As valid arguments for register unit are given as inputs in this first test case which are respectively: Expected Inputs

Mail id: a-z, 0-9, @

Password: all alphabets and special characters are allowed

**Test Results:**

In any of the both fields the inputs for the user are wrong he will receive a popup entertaining him that the user credentials are wrongly given and he has to log in with true ones.

Test inputs all have **passed** for correct inputs applied. All the functions are working properly on correct inputs. No anomalies.

**Test Case – 3**

**Description:** The user can Log in only if he was registered before. For logging in the user gives the inputs mail id and password set while registering. The inputs if matched with the stored details in database the user will be accessed to log in.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those browsers all the input fields are to be tested accordingly.

**Test Inputs:** As valid arguments for register unit are given as inputs in this first test case which are respectively: Expected Inputs

Mail id: a-z, 0-9, @

Password: all alphabets and special characters are allowed

**Test Results:**

In any of the both fields the inputs for the user are wrong he will receive a popup entertaining him that the user credentials are wrongly given and he has to log in with true ones.

Test inputs all in all fields at least one anomaly will not be accepted which ensures **passed** according to the inputs applied. All the functions are working properly on correct inputs. No anomalies

**Test Case – 4**

**Description:** The user can Log in only if he was registered before. For logging in the user gives the inputs mail id and password set while registering. The inputs if matched with the stored details in database the user will be accessed to log in.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those browsers all the input fields are to be tested accordingly.

**Test Inputs:** As valid arguments for register unit are given as inputs in this first test case which are respectively: Expected Inputs

Mail id: a-z, 0-9, @

Password: all alphabets and special characters are allowed

**Test Results:**

In any of the both fields the inputs for the user are wrong he will receive a popup entertaining him that the user credentials are wrongly given and he has to log in with true ones.

Test inputs all have **passed** for correct inputs applied. All the functions are working properly on correct inputs. No anomalies.

**Section 2 – White Box Test Cases**

**Test Case:**

**Description:** The user can Log in only if he was registered before. For logging in the user gives the inputs mail id and password set while registering. The inputs if matched with the stored details in database the user will be accessed to log in.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly.

**Test Inputs:** As valid arguments for register unit are given as inputs in this first test case which are respectively: Expected Inputs.

Mail id: a-z, 0-9, @

Password: all alphabets and special characters are allowed

**Resolved BUG:**

If input type is ‘text’ all the characters will be accepted but for mail id field the input type should be ‘email’ otherwise email restrictions will not function. This anomaly has been rectified by using appropriate input types.

**Test Results:**

After the bug implementation the test case is **passed** for correct inputs applied. All the functions are working properly on correct inputs. No anomalies

## Unit 3 – Valid user (register once)

**Section – 1 Black Box Test Cases**

**Test case:**

**Description:** The user once register should not be registered again under any circumstances except the management allows the user. The primary key which will be a unique field over everywhere which is email id in our web application cannot be used more than once to register. This is a server side page used for verifying the user whether he is unique or not.

**Test Approach**: In black box what matters is about the input given and the output received. Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly.

**Test inputs:** As this is server side script the inputs will be sent and received from this script as objects. The inputs will be given to the register page where the inputs are verified and echo will be received that successfully registered if they are valid. Otherwise the user is asked to give the inputs correctly or if the user is already registered then popup will be received saying user is already registered and asks to log in.

**Test Results:** No bugs. Running successfully. **Passed** the test.

**Section 2 – White Box Test Cases**

**Test case:**

**Description:** The user once register should not be registered again under any circumstances except the management allows the user. The primary key which will be a unique field over everywhere which is email id in our web application cannot be used more than once to register. This is a server side page used for verifying the user whether he is unique or not.

**Test Approach**: In black box what matters is about the input given and the output received. Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly.

**Test inputs:** As this is server side script the inputs will be sent and received from this script as objects. The inputs will be given to the register page where the inputs are verified and echo will be received that successfully registered if they are valid. Otherwise the user is asked to give the inputs correctly or if the user is already registered then popup will be received saying user is already registered and asks to log in..

**Test Results:**

The test case is **passed** for correct inputs applied. All the functions are working properly on correct inputs. No anomalies found.

## Unit 4 – loginUser

**Section – 1 Black Box Test Cases**

**Test case – 1**

**Description:** The user once registered can login to our web application giving appropriate mail id and password. The primary key which will be a unique field over everywhere which is email id in our web application is used for logging in which will be same to the mail given while registering. This is a server side script used for verifying the user whether he is unique or not.

**Test Approach**: In black box what matters is about the input given and the output received. Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly.

**Test inputs:** As this is server side script the inputs will be sent and received from this script as objects. The inputs will be given in index page which has log in credentials thereafter the inputs are verified and will be directed to dashboard if they are valid. Otherwise the user is asked to give the inputs correctly and asks to log in using correct details.

**Test Results:** No bugs. The test case is **passed** for correct inputs applied. All the functions are working properly on correct inputs. No anomalies found.

**Section 2 – White Box Test Cases**

**Description:** The user once registered can login to our web application giving appropriate mail id and password. The primary key which will be a unique field over everywhere which is email id in our web application is used for logging in which will be same to the mail given while registering. This is a server side script used for verifying the user whether he is unique or not.

**Test Approach**: In black box what matters is about the input given and the output received. Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly.

White Box test cases included, where for a given unit of code path coverage, branch coverage, statement coverage must have been achieved by the derived set of test cases. By the above test cases which were covered (Black box test cases) there is no need of extra test cases to complete/ achieve the above mentioned coverage’s. Now, we can conclude that the white box test cases have been executed as above.

## Unit 5 – dashboard

**Section – 1 Black Box Test Cases**

|  |  |  |
| --- | --- | --- |
| Test Case | Function | Type |
| Test Case 1 | Search | Appropriate Inputs(All fields) |
| Test Case 2 | Search | Invalid Inputs (Discrete fields) |
| Test Case 3 | Search | Invalid Inputs (All fields) |
| Test Case 4 | Search | Special/ Un allowed characters. (All fields) |

**Test Case – 1**

**Description:** The dashboard is the first page after logging in. From dashboard the user can search for booking cars, get in contact with the travel agents and concerned fellows, and can go to search for shared cars as well. The user has to give inputs from and to locations, from and to dates and on clicking search button the available cars which can be booked will be showed with some details about the cab like number of seats, car name.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly. Testing will be done using different inputs in all the fields available.

**Test Inputs:** As valid arguments for register unit are given as inputs in this first test case which are respectively: Expected Inputs

From Location: A drop down box will be given with the allowed cities or places among which one place will be selected.

To Location: A drop down box will be given with the allowed cities or places among which one place will be selected.

From date: Date selector is made ready on the selection part which can be selected and will be above the present day date.

To date: Date selector is made ready on the selection part which can be selected and will be above the present day date.

**Test Results:**

Test inputs all have **passed** for correct inputs applied. All the functions are working properly on correct inputs. No anomalies.

**Test Case – 2**

**Description:** The dashboard is the first page after logging in. From dashboard the user can search for booking cars, get in contact with the travel agents and concerned fellows, and can go to search for shared cars as well. The user has to give inputs from and to locations, from and to dates and on clicking search button the available cars which can be booked will be showed with some details about the cab like number of seats, car name.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly. Testing will be done using different inputs in all the fields available.

**Test Inputs:** As valid arguments for register unit are given as inputs in this first test case which are respectively: Expected Inputs

From Location: A drop down box will be given with the allowed cities or places among which one place will be selected.

To Location: A drop down box will be given with the allowed cities or places among which one place will be selected.

From date: Date selector is made ready on the selection part which can be selected and will be above the present day date.

To date: Date selector is made ready on the selection part which can be selected and will be above the present day date.

**Test Results:**

On the inputs which are valid the available cars will get from database. Invalid inputs doesn’t show cars and provide popup explaining the inputs mistakes.

Test inputs all have **passed** for correct inputs applied. All the functions are working properly on correct inputs. No anomalies.

**Test Case – 3**

**Description:** The dashboard is the first page after logging in. From dashboard the user can search for booking cars, get in contact with the travel agents and concerned fellows, and can go to search for shared cars as well. The user has to give inputs from and to locations, from and to dates and on clicking search button the available cars which can be booked will be showed with some details about the cab like number of seats, car name.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly. Testing will be done using different inputs in all the fields available.

**Test Inputs:** As valid arguments for register unit are given as inputs in this first test case which are respectively: Expected Inputs

From Location: A drop down box will be given with the allowed cities or places among which one place will be selected.

To Location: A drop down box will be given with the allowed cities or places among which one place will be selected.

From date: Date selector is made ready on the selection part which can be selected and will be above the present day date.

To date: Date selector is made ready on the selection part which can be selected and will be above the present day date.

**Test Results:**

In dates the user can give past dates which shows there are no cars available in those dates. Test case is **passed** for correct inputs applied. All the functions are working properly on correct inputs. No anomalies.

**Test Case – 4**

**Description:** The dashboard is the first page after logging in. From dashboard the user can search for booking cars, get in contact with the travel agents and concerned fellows, and can go to search for shared cars as well. The user has to give inputs from and to locations, from and to dates and on clicking search button the available cars which can be booked will be showed with some details about the cab like number of seats, car name.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly. Testing will be done using different inputs in all the fields available.

**Test Inputs:** As valid arguments for register unit are given as inputs in this first test case which are respectively: Expected Inputs

From Location: A drop down box will be given with the allowed cities or places among which one place will be selected.

To Location: A drop down box will be given with the allowed cities or places among which one place will be selected.

From date: Date selector is made ready on the selection part which can be selected and will be above the present day date.

To date: Date selector is made ready on the selection part which can be selected and will be above the present day date.

**Test Results:**

Test inputs all have **passed** for correct inputs applied. All the functions are working properly on correct inputs. No anomalies.

**Section 2 – White Box Test Cases**

**Description:** The dashboard is the first page after logging in. From dashboard the user can search for booking cars, get in contact with the travel agents and concerned fellows, and can go to search for shared cars as well. The user has to give inputs from and to locations, from and to dates and on clicking search button the available cars which can be booked will be showed with some details about the cab like number of seats, car name.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly. Testing will be done using different inputs in all the fields available.

**Test Results:** White Box test cases included, where for a given unit of code path coverage, branch coverage, statement coverage must have been achieved by the derived set of test cases. By the above test cases which were covered (Black box test cases) there is no need of extra test cases to complete/ achieve the above mentioned coverage’s. Now, we can conclude that the white box test cases have been executed as above.

## Unit 6 – getcars:

**Section – 1 Black Box Test Cases**

**Description:** The dashboard is the first page after logging in. From dashboard the user can search for booking cars, get in contact with the travel agents and concerned fellows, and can go to search for shared cars as well. The user has to give inputs from and to locations, from and to dates and on clicking search button the available cars which can be booked will be showed with some details about the cab like number of seats, car name.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly. Testing will be done using different inputs in all the fields available.

**Test inputs:** The user can’t see the presence of this page as this is server side script. Only the required functions are done through this function. Whichever cars are not booked, ready to ride at any time are shown getting the data from database through query. On trying with valid inputs the cars are shown and for invalid the dashboard asked for appropriate inputs.

**Test Result:** This case is working perfectly without any bugs or errors. Hence this case **passed** the requirement part.

**Section 2 – White Box Test Cases**

**Description:** The dashboard is the first page after logging in. From dashboard the user can search for booking cars, get in contact with the travel agents and concerned fellows, and can go to search for shared cars as well. The user has to give inputs from and to locations, from and to dates and on clicking search button the available cars which can be booked will be showed with some details about the cab like number of seats, car name.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly. Testing will be done using different inputs in all the fields available.

**Test inputs:** The user can’t see the presence of this page as this is server side script. Only the required functions are done through this function. Whichever cars are not booked, ready to ride at any time are shown getting the data from database through query. On trying with valid inputs the cars are shown and for invalid the dashboard asked for appropriate inputs.

**Test Result:** The query written is correct and working well. If have any doubts can be tried in any of the sql or database working software for expected output. This case is working perfectly without any bugs or errors. Hence this case **passed** the requirement part.

## Unit 7 – shared

|  |  |  |
| --- | --- | --- |
| Test Case | Function | Type |
| Test Case 1 | Shared | Appropriate Inputs(All fields) |
| Test Case 2 | Shared | Invalid Inputs (Discrete fields) |
| Test Case 3 | Shared | Invalid Inputs (All fields) |
| Test Case 4 | Shared | Special/ Un allowed characters. (All fields) |

**Test Case – 1:**

**Description:** The shared form page will have spaces for date and locations details. From shared page the user has access to search for booking cars, get in contact with the travel agents and concerned fellows, and can go to search for shared cars as well. The user has to give inputs from and to locations, from and to dates and on clicking search button the available cars which can be booked will be showed with some details about the cab like number of seats, car name.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly. Testing will be done using different inputs in all the fields available.

**Test Inputs:** As valid arguments for register unit are given as inputs in this first test case which are respectively: Expected Inputs

From Location: A drop down box will be given with the allowed cities or places among which one place will be selected.

To Location: A drop down box will be given with the allowed cities or places among which one place will be selected.

From date: Date selector is made ready on the selection part which can be selected and will be above the present day date.

To date: Date selector is made ready on the selection part which can be selected and will be above the present day date.

**Test Result:** The query written is correct and working well. If have any doubts can be tried in any of the sql or database working software for expected output. This case is working perfectly without any bugs or errors. Hence this case **passed** the requirement part.

**Test Case – 2:**

**Description:** The shared form page will have spaces for date and locations details. From shared page the user has access to search for booking cars, get in contact with the travel agents and concerned fellows, and can go to search for shared cars as well. The user has to give inputs from and to locations, from and to dates and on clicking search button the available cars which can be booked will be showed with some details about the cab like number of seats, car name.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly. Testing will be done using different inputs in all the fields available.

**Test Inputs:** As valid arguments for register unit are given as inputs in this first test case which are respectively: Expected Inputs

From Location: A drop down box will be given with the allowed cities or places among which one place will be selected.

To Location: A drop down box will be given with the allowed cities or places among which one place will be selected.

From date: Date selector is made ready on the selection part which can be selected and will be above the present day date.

To date: Date selector is made ready on the selection part which can be selected and will be above the present day date.

**Test Result:** The query written is correct and working well. If have any doubts can be tried in any of the sql or database working software for expected output. This case is working perfectly without any bugs or errors. Hence this case **passed** the requirement part.

**Test Case – 3:**

**Description:** The shared form page will have spaces for date and locations details. From shared page the user has access to search for booking cars, get in contact with the travel agents and concerned fellows, and can go to search for shared cars as well. The user has to give inputs from and to locations, from and to dates and on clicking search button the available cars which can be booked will be showed with some details about the cab like number of seats, car name.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly. Testing will be done using different inputs in all the fields available.

**Test Inputs:** As valid arguments for register unit are given as inputs in this first test case which are respectively: Expected Inputs

From Location: A drop down box will be given with the allowed cities or places among which one place will be selected.

To Location: A drop down box will be given with the allowed cities or places among which one place will be selected.

From date: Date selector is made ready on the selection part which can be selected and will be above the present day date.

To date: Date selector is made ready on the selection part which can be selected and will be above the present day date.

**Test Result:** The query written is correct and working well. If have any doubts can be tried in any of the sql or database working software for expected output. This case is working perfectly without any bugs or errors. Hence this case **passed** the requirement part.

**Test Case – 4:**

**Description:** The shared form page will have spaces for date and locations details. From shared page the user has access to search for booking cars, get in contact with the travel agents and concerned fellows, and can go to search for shared cars as well. The user has to give inputs from and to locations, from and to dates and on clicking search button the available cars which can be booked will be showed with some details about the cab like number of seats, car name.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly. Testing will be done using different inputs in all the fields available.

**Test Inputs:** As valid arguments for register unit are given as inputs in this first test case which are respectively: Expected Inputs

From Location: A drop down box will be given with the allowed cities or places among which one place will be selected.

To Location: A drop down box will be given with the allowed cities or places among which one place will be selected.

From date: Date selector is made ready on the selection part which can be selected and will be above the present day date.

To date: Date selector is made ready on the selection part which can be selected and will be above the present day date.

**Test Result:** This case is working perfectly without any bugs or errors. Hence this case **passed** the requirement part for special characters are not allowed in these fields.

**Section 2 – White Box Test Cases**

**Description:** The shared form page will have spaces for date and locations details. From shared page the user has access to search for booking cars, get in contact with the travel agents and concerned fellows, and can go to search for shared cars as well. The user has to give inputs from and to locations, from and to dates and on clicking search button the available cars which can be booked will be showed with some details about the cab like number of seats, car name.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly. Testing will be done using different inputs in all the fields available.

**Test Results:** White Box test cases included, where for a given unit of code path coverage, branch coverage, statement coverage must have been achieved by the derived set of test cases. By the above test cases which were covered (Black box test cases) there is no need of extra test cases to complete/ achieve the above mentioned coverage’s. Now, we can conclude that the white box test cases have been executed as above.

## Unit 8 – getSharedCars

**Section – 1 Black Box Test Cases**

**Description:** The shared form page will have spaces for date and locations details. From shared page the user has access to search for booking cars, get in contact with the travel agents and concerned fellows, and can go to search for shared cars as well. The user has to give inputs from and to locations, from and to dates and on clicking search button the available cars which can be booked will be showed with some details about the cab like number of seats, car name.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly. Testing will be done using different inputs in all the fields available.

**Test inputs:** The user can’t see the presence of this page as this is server side script. Only the required functions are done through this function. Whichever cars are booked already for ride and are shared by the booked user are shown through query. On trying with valid inputs the cars are shown and for invalid then the user is asked for appropriate inputs.

**Test Result:** This case is working perfectly without any bugs or errors. Hence this case **passed** the requirement part.

**Section 2 – White Box Test Cases**

**Description:** The shared form page will have spaces for date and locations details. From shared page the user has access to search for booking cars, get in contact with the travel agents and concerned fellows, and can go to search for shared cars as well. The user has to give inputs from and to locations, from and to dates and on clicking search button the available cars which can be booked will be showed with some details about the cab like number of seats, car name.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly. Testing will be done using different inputs in all the fields available.

**Test inputs:** The user can’t see the presence of this page as this is server side script. Only the required functions are done through this function. Whichever cars are not booked, ready to ride at any time are shown getting the data from database through query. On trying with valid inputs the cars are shown and for invalid the dashboard asked for appropriate inputs.

**Test Result:** The query written is correct and working well. If have any doubts can be tried in any of the sql or database working software for expected output. This case is working perfectly without any bugs or errors. Hence this case **passed** the requirement part.

## Unit 9 – getCarnumber

**Section – 1 Black Box Test Cases**

**Description:** The shared form page will have spaces for date and locations details. From shared page the user has access to search for booking cars, get in contact with the travel agents and concerned fellows, and can go to search for shared cars as well. The user has to give inputs from and to locations, from and to dates and on clicking search button the available cars which can be booked will be showed with some details about the cab like number of seats, car name.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly. Testing will be done using different inputs in all the fields available.

**Test inputs:** The user can’t see the presence of this page as this is server side script. Only the required functions are done through this function. Whichever cars are booked already for ride and are shared by the booked user are shown through query. On trying with valid inputs the cars are shown and for invalid then the user is asked for appropriate inputs.

**Test Result:** This case is working perfectly without any bugs or errors. Hence this case **passed** the requirement part.

**Section 2 – White Box Test Cases**

**Description:** The shared form page will have spaces for date and locations details. From shared page the user has access to search for booking cars, get in contact with the travel agents and concerned fellows, and can go to search for shared cars as well. The user has to give inputs from and to locations, from and to dates and on clicking search button the available cars which can be booked will be showed with some details about the cab like number of seats, car name.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly. Testing will be done using different inputs in all the fields available.

**Result:**

White Box test cases included, where for a given unit of code path coverage, branch coverage, statement coverage must have been achieved by the derived set of test cases. By the above test cases which were covered (Black box test cases) there is no need of extra test cases to complete/ achieve the above mentioned coverage’s. Now, we can conclude that the white box test cases have been executed as above.

## Unit 10 – booking

**Section – 1 Black Box Test Cases**

|  |  |  |
| --- | --- | --- |
| Test Case | Function | Type |
| Test Case 1 | Book | Appropriate Inputs(All fields) |
| Test Case 2 | Book | Invalid Inputs (Discrete fields) |
| Test Case 3 | Book | Invalid Inputs (All fields) |
| Test Case 4 | Book | Special/ Un allowed characters. (All fields) |

**Test Case – 1:**

**Description:** The booking form page arises when user clicks book now on any of the given cars for the user for booking. Places for date and locations details are to be filed along with number of seats required to be booked, want to share or not, from and to dates and time the cab should be ready.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly. Testing will be done using different inputs in all the above fields available.

**Test inputs:** In first case all fields are filled with appropriate complete good details.

Wanted time: (24 hours clock) When to start the journey

Wanted place: Where to start the journey a-z

From Location: Starting place

To Location: Final destination

Number of seats: less than or equal to 7 (as 7 seater is the largest car provided), 0-9.

Want to share or not? : Yes or No (selection)

**Test Result:** This case is working perfectly without any bugs or errors. Hence this case **passed** the requirement part.

**Test Case – 2:**

**Description:** The booking form page arises when user clicks book now on any of the given cars for the user for booking. Places for date and locations details are to be filed along with number of seats required to be booked, want to share or not, from and to dates and time the cab should be ready.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly. Testing will be done using different inputs in all the above fields available.

**Test inputs:** In first case all fields are filled with appropriate complete good details.

Wanted time: (24 hours clock) When to start the journey

Wanted place: Where to start the journey a-z

From Location: Starting place

To Location: Final destination

Number of seats: less than or equal to 7 (as 7 seater is the largest car provided), 0-9.

Want to share or not? : Yes or No (selection)

Some wrong inputs are tried to fill the fields and they should not be accepted.

**Test Result:** This case is working perfectly without any bugs or errors. Hence this case **passed** the requirement part.

**Test Case – 3:**

**Description:** The booking form page arises when user clicks book now on any of the given cars for the user for booking. Places for date and locations details are to be filed along with number of seats required to be booked, want to share or not, from and to dates and time the cab should be ready.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly. Testing will be done using different inputs in all the above fields available.

**Test inputs:** In first case all fields are filled with appropriate complete good details.

Wanted time: (24 hours clock) When to start the journey

Wanted place: Where to start the journey a-z

From Location: Starting place

To Location: Final destination

Number of seats: less than or equal to 7 (as 7 seater is the largest car provided), 0-9.

Want to share or not? : Yes or No (selection)

Wrong inputs are given in every field available which should not be allowed.

**Test Result:** Popup arises saying that the car is not booked and asks to try again using correct information.This case is working perfectly without any bugs or errors. Hence this case **passed** the requirement part.

**Test Case – 4:**

**Description:** The booking form page arises when user clicks book now on any of the given cars for the user for booking. Places for date and locations details are to be filed along with number of seats required to be booked, want to share or not, from and to dates and time the cab should be ready.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly. Testing will be done using different inputs in all the above fields available.

**Test inputs:** In first case all fields are filled with appropriate complete good details.

Wanted time: (24 hours clock) When to start the journey

Wanted place: Where to start the journey a-z

From Location: Starting place

To Location: Final destination

Number of seats: less than or equal to 7 (as 7 seater is the largest car provided), 0-9.

Want to share or not? : Yes or No (selection)

**Test Result:** Similar to the above case, special characters are not at all allowed and so there are no bugs or errors. Hence this case **passed** the requirement part.

**Section 2 – White Box Test Cases**

White Box test cases included, where for a given unit of code path coverage, branch coverage, statement coverage must have been achieved by the derived set of test cases. By the above test cases which were covered (Black box test cases) there is no need of extra test cases to complete/ achieve the above mentioned coverage’s. Now, we can conclude that the white box test cases have been executed as above.

## Unit 11 – bookAvaibleCar

**Section – 1 Black Box Test Cases**

**Description:** The booking form page arises when user clicks book now on any of the given cars for the user for booking. Places for date and locations details are to be filed along with number of seats required to be booked, want to share or not, from and to dates and time the cab should be ready.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly. Testing will be done using different inputs in all the above fields available.

**Test inputs:** All the data provided by the user are sent to the database for saving the action and generate auto mail to the user as well as cab driver. The data should be saved in the database for further actions.

**Test Result:** This case is working perfectly without any bugs or errors. Hence this case **passed** the requirement part.

**Section 2 – White Box Test Cases**

**Description:** The booking form page arises when user clicks book now on any of the given cars for the user for booking. Places for date and locations details are to be filed along with number of seats required to be booked, want to share or not, from and to dates and time the cab should be ready.

**Test** **Approach:** Testing should be done on almost all available web browsers as the project work is web application which will be opened and used in browsers for example chrome, firefox, uc browser, vivaldi, on pc’s and mobile interfaces as well. In all those situations all the input fields are to be tested accordingly. Testing will be done using different inputs in all the above fields available.

**Result:**

White Box test cases included, where for a given unit of code path coverage, branch coverage, statement coverage must have been achieved by the derived set of test cases. By the above test cases which were covered (Black box test cases) there is no need of extra test cases to complete/ achieve the above mentioned coverage’s. Now, we can conclude that the white box test cases have been executed as above.

# Integration Testing

## Introduction

## After unit testing is completed, now we do the integration testing, where we integrate and make the product work with exchange of data happening with other functions. Test cases can be drawn in two different ways. Black box test cases, White box cases. Each section will be having number of test cases and will be described appropriately.

## Integrate Part 1 🡪 Pushing Data from Content to Scripts

**Section 1 – Black Box Test Cases**

|  |  |  |
| --- | --- | --- |
| Test Case | Type | Function |
| Test Case 1 | Appropriate Data Sent Over | Ajax |
| Test Case 2 | Invalid Data Sent Over | Ajax |

**Test Case 1**

**Test Description:** JavaScript takes the value from the user given space and makes all the data variables as an object and this is sent to server script side. Now we check how many variables reach the JavaScript and we analyze the behavior.

**Test Approach:** The testing should be performed on different web platforms. After the user opens the website the server should be able to respond to it. The user login and how the database reacts to that, only the authenticated user can login or they have to register and booking is done properly or not with no hassle. Canceling with no faulty reasons. Also the interface with each browser is working properly or not.

**Test Inputs:** The input data variable are sent to respect blocks where they are sent and received to valid JavaScript function.

**Test Results:**

The result of the above is “Passed”, and no bugs has been received.

**Test Case 2**

**Test Description:** JavaScript takes the value from the user given space and makes all the data variables as an object and this is sent to server script side. Now we check how many variables reach the JavaScript and we analyze the behavior.

**Test Approach:** The testing should be performed on different web platforms. After the user opens the website the server should be able to respond to it. The user login and how the database reacts to that, only the authenticated user can login or they have to register and booking is done properly or not with no hassle. Canceling with no faulty reasons. Also the interface with each browser is working properly or not.

**Test Inputs:** The input data variable are sent to respect blocks where they are sent and received to valid JavaScript function.

**Test Results:**

The result of the above is “Passed”, and no bugs has been received.

**Section 1 – White Box Test Cases**

|  |  |  |
| --- | --- | --- |
| Test Case | Type | Function |
| Test Case 1 | Data interchanged/ manipulated while sent over | Ajax |

**Test Case 1**

**Test Description:** JavaScript takes the value from the user given space and makes all the data variables as an object and this is sent to server script side. Now we check how many variables reach the JavaScript and we analyze the behavior.

**Test Approach:** The testing should be performed on different web platforms. After the user opens the website the server should be able to respond to it. The user login and how the database reacts to that, only the authenticated user can login or they have to register and booking is done properly or not with no hassle. Canceling with no faulty reasons. Also the interface with each browser is working properly or not.

**Test Inputs:** Whether the taken data object is sent/received to the appropriate function as the data cannot verify itself that the value for its functionality.

**Test Results:**

The result of the above is “Passed”, and no bugs has been received.

## Integrate Part 2 🡪 Posting data through AJAX to scripts

**Section 1 – Black Box Test Cases**

|  |  |  |
| --- | --- | --- |
| Test Case | Type | Function |
| Test Case 1 | Valid Data object Sent | Ajax |
| Test Case 2 | Data object being manipulated over sending | Ajax |
| Test Case 3 | Data Lost Over Server or in between | Ajax |

**Test Case 1**

**Test Description:** Ajax is used to move or post the data to Php scripts on the server to continue the future process to get or change the data from the database. Here we analyze the combined form over different type of objects.

**Test Approach:** The testing should be performed on different web platforms. After the user opens the website the server should be able to respond to it. The user login and how the database reacts to that, only the authenticated user can login or they have to register and booking is done properly or not with no hassle. Canceling with no faulty reasons. Also the interface with each browser is working properly or not.

**Test Inputs:** This case deals with the block where the data object that is being sent or sent over POST is valid to the appropriate functions.

**Test Results:**

The result of the above is “Passed”, and no bugs has been received.

**Test Case 2**

**Test Description:** Ajax is used to move or post the data to Php scripts on the server to continue the future process to get or change the data from the database. Here we analyze the combined form over different type of objects.

**Test Approach:** The testing should be performed on different web platforms. After the user opens the website the server should be able to respond to it. The user login and how the database reacts to that, only the authenticated user can login or they have to register and booking is done properly or not with no hassle. Canceling with no faulty reasons. Also the interface with each browser is working properly or not.

**Test Inputs:** This case deals with the block where the data object that is being sent or sent over POST is manipulated before reaching the appropriate function that is to be executed.

**Test Results:**

The result of the above is “Passed”, and no bugs has been received.

**Test Case 3**

**Test Description:** Ajax is used to move or post the data to Php scripts on the server to continue the future process to get or change the data from the database. Here we analyze the combined form over different type of objects.

**Test Approach:** The testing should be performed on different web platforms. After the user opens the website the server should be able to respond to it. The user login and how the database reacts to that, only the authenticated user can login or they have to register and booking is done properly or not with no hassle. Canceling with no faulty reasons. Also the interface with each browser is working properly or not.

**Test Inputs:** This case deals with the block where the data object that is being sent or sent over POST is completely lost or there is a connection failure over sending the data.

**Test Results:**

The result of the above is “Passed”, and no bugs has been received.

**Section 2 – White Box Test Cases**

**Note:**

This white box test cases, which include the path coverage, branch coverage, statement coverage have been achieved by the above set of cases. The above set of cases we are telling that Black box test cases has been completed. Now we conclude that white box test cases are being executed as above cases.

## Integrate Part 3 🡪 Database Connection via Script

**Section 1 – Black Box Test Cases**

|  |  |  |
| --- | --- | --- |
| Test Case | Type | Function |
| Test Case 1 | Valid Credentials to server | PHP |
| Test Case 2 | Invalid Credentials to server | PHP |

**Test Case 1**

**Test Description:** Credentials are passed through scripts to connect to the database, to get or change the values in the required tables. Connection depends on the credentials that have been shared to the server. We have analyzed on different inputs to the credentials.

**Test Approach:** The testing should be performed on different web platforms. After the user opens the website the server should be able to respond to it. The user login and how the database reacts to that, only the authenticated user can login or they have to register and booking is done properly or not with no hassle. Canceling with no faulty reasons. Also the interface with each browser is working properly or not.

**Test Inputs:** This case deals with the block where valid credentials are sent over the server.

**Test Results:**

The result of the above is “Passed”, and no bugs has been received.

**Test Case 2**

**Test Description:** Credentials are passed through scripts to connect to the database, to get or change the values in the required tables. Connection depends on the credentials that have been shared to the server. We have analyzed on different inputs to the credentials.

**Test Approach:** The testing should be performed on different web platforms. After the user opens the website the server should be able to respond to it. The user login and how the database reacts to that, only the authenticated user can login or they have to register and booking is done properly or not with no hassle. Canceling with no faulty reasons. Also the interface with each browser is working properly or not.

**Test Inputs:** This case deals with the block where Invalid credentials are sent over the server.

**Test Results:**

The result of the above is “Passed”, and no bugs has been received.

**Section 2 – White Box Test Cases**

**Note:**

This white box test cases, which include the path coverage, branch coverage, statement coverage have been achieved by the above set of cases. The above set of cases we are telling that Black box test cases has been completed. Now we conclude that white box test cases are being executed as above cases.

|  |  |  |
| --- | --- | --- |
| Test Case | Type | Function |
| Test Case 1 | Connection Fail over network | PHP |

**Test Case 1**

**Test Description:** Credentials are passed through scripts to connect to the database, to get or change the values in the required tables. Connection depends on the credentials that have been shared to the server. We have analyzed on different inputs to the credentials..

**Test Approach:** The testing should be performed on different web platforms. After the user opens the website the server should be able to respond to it. The user login and how the database reacts to that, only the authenticated user can login or they have to register and booking is done properly or not with no hassle. Canceling with no faulty reasons. Also the interface with each browser is working properly or not.

**Test Inputs:** If the network fails, what are the credentials were given to the server will behavior and the function is appropriate or not.

**Test Results:**

The result of the above is “Passed”, and no bugs has been received.

## Integrate Part 4 🡪 Posting data through AJAX to scripts

**Section 1 – Black Box Test Cases**

|  |  |  |
| --- | --- | --- |
| Test Case | Type | Function |
| Test Case 1 | Valid Data object Sent from Script | Ajax |
| Test Case 2 | Ripped data object received. | Ajax |
| Test Case 3 | Incomplete data received. | Ajax |

**Test Case 1**

**Test Description:** Ajax needs to gets data from PHP scripts that is in server to continue the process. We analyze the combined functions over different objects received.

**Test Approach:** The testing should be performed on different web platforms. After the user opens the website the server should be able to respond to it. The user login and how the database reacts to that, only the authenticated user can login or they have to register and booking is done properly or not with no hassle. Canceling with no faulty reasons. Also the interface with each browser is working properly or not.

**Test Inputs:** This case deals with the block where the data object that is being received is valid and have all the data that is required to complete the function.

**Test Results:**

The result of the above is “Passed”, and no bugs has been received.

**Test Case 2**

**Test Description:** Ajax needs to gets data from PHP scripts that is in server to continue the process. We analyze the combined functions over different objects received.

**Test Approach:** The testing should be performed on different web platforms. After the user opens the website the server should be able to respond to it. The user login and how the database reacts to that, only the authenticated user can login or they have to register and booking is done properly or not with no hassle. Canceling with no faulty reasons. Also the interface with each browser is working properly or not.

**Test Inputs:** This case deals with the block where the data object that is being received is ripped off and not compatible to the committed query.

**Test Results:**

The result of the above is “Passed”, and no bugs has been received

**Test Case 3**

**Test Description:** Ajax needs to gets data from PHP scripts that is in server to continue the process. We analyze the combined functions over different objects received.

**Test Approach:** The testing should be performed on different web platforms. After the user opens the website the server should be able to respond to it. The user login and how the database reacts to that, only the authenticated user can login or they have to register and booking is done properly or not with no hassle. Canceling with no faulty reasons. Also the interface with each browser is working properly or not.

.

**Test Inputs:** This case deals with the block where the data object that is being received is incomplete and the data is not sufficient to the function to complete the query.

**Test Results:**

The result of the above is “Passed”, and no bugs has been received.

**Section 2 – White Box Test Cases**

**Note:**

This white box test cases, which include the path coverage, branch coverage, statement coverage have been achieved by the above set of cases. The above set of cases we are telling that Black box test cases has been completed. Now we conclude that white box test cases are being executed as above cases.

## Integrate Part 5 🡪 Appending the Data to Content

**Section 1 – Black Box Test Cases**

|  |  |  |
| --- | --- | --- |
| Test Case | Type | Function |
| Test Case 1 | Valid Data Supplied | JQuery |
| Test Case 2 | Invalid Data Supplied | JQuery |

**Test Case 1**

**Test Description:** We get the necessary data from the server, now we append the contents in the appropriate section so that it is visible to the user what is going on the user-interface. This case is to analyze that the whether it shows the appropriate data to the user.

**Test Approach:** The testing should be performed on different web platforms. After the user opens the website the server should be able to respond to it. The user login and how the database reacts to that, only the authenticated user can login or they have to register and booking is done properly or not with no hassle. Canceling with no faulty reasons. Also the interface with each browser is working properly or not.

**Test Inputs:** This case deals with the block where valid data and valid section id is received. This is to be appended to the content and verified.

**Test Results:**

The result of the above is “Passed”, and no bugs has been received.

**Test Case 2**

**Test Description:** We get the necessary data from the server, now we append the contents in the appropriate section so that it is visible to the user what is going on the user-interface. This case is to analyze that the whether it shows the appropriate data to the user.

**Test Approach:** The testing should be performed on different web platforms. After the user opens the website the server should be able to respond to it. The user login and how the database reacts to that, only the authenticated user can login or they have to register and booking is done properly or not with no hassle. Canceling with no faulty reasons. Also the interface with each browser is working properly or not.

**Test Inputs:** This case deals with the block where valid data and valid section id might not be received. Upon this the appending behavior is to be analyzed.

**Test Results:**

The result of the above is “Passed”, and no bugs has been received

**Section 2 – White Box Test Cases**

**Note:**

This white box test cases, which include the path coverage, branch coverage, statement coverage have been achieved by the above set of cases. The above set of cases we are telling that Black box test cases has been completed. Now we conclude that white box test cases are being executed as above cases.

|  |  |  |
| --- | --- | --- |
| Test Case | Type | Function |
| Test Case 1 | Error/ Insufficient Data | JQuery |

**Test Case 1**

**Test Description** We get the necessary data from the server, now we append the contents in the appropriate section so that it is visible to the user what is going on the user-interface. This case is to analyze that the whether it shows the appropriate data to the user.

**Test Approach:** The testing should be performed on different web platforms. After the user opens the website the server should be able to respond to it. The user login and how the database reacts to that, only the authenticated user can login or they have to register and booking is done properly or not with no hassle. Canceling with no faulty reasons. Also the interface with each browser is working properly or not.

**Test Inputs:** Data has been undergone so many phase before appending. There is a probability that after so many restrictions may be error data of insufficient data might be received to the content. This behavior is to be analyzed.

**Test Results:**

The result of the above is “Passed”, and no bugs has been received

# Bugs

## Bug #1

**Bug Description:** While filling the date (dd-mm-yyyy), it was first made in text form, ie. It would take character, symbols as input also still show output. So this has been resolved by changing the data type of it to char.

**Bug Status:** RESOLVED