Software Test Description

SSE 657

Erin Cargin Dylan DeVries
Jarod Miller Joel Seepersaud

1. Overview

This document describes a detailed breakdown of every test case that is used to verify the functionality of our application. Every test case verifies specific design requirements and has input parameters and expected output. The overall strategy of our testing is defined in our Software Test Plan. Because the unit under test is a graphical user interface, most test cases are run manually by a user by inputting information. These test cases are for regression testing whenever new code is added to the baseline and also for qualifying a release build.

2. Test Cases

The test cases are split into groups based on the graphical user interface window or component under test. Every component has a test matrix where the user is required to input a specific input and verify an expected output. The results for the test cases are found in the Software Test Report.

Login Page:

Test Case	Email Input	Password Input	Expected Output	Requirements Verified
1	None	None	Login Screen is shown	APPLICATION_POWER_UP
2	None	None	Upon login button click, the application displays “invalid params” error	VERIFY_LOGIN_DATA_FILLED
3	testuser@mail.com	None	Upon login button click, the application displays “invalid params” error	VERIFY_LOGIN_DATA_FILLED
4	None	password	Upon login button click, the application displays “invalid params” error	VERIFY_LOGIN_DATA_FILLED
5	testuser@mail.com	wrongpas	Upon login button click, the application displays “invalid login” error	SEND_VALIDATION_REQUEST, LOGIN_FAILURE
6	testuser@mail.com	password	Upon login button click, the application displays the home screen	SEND_VALIDATION_REQUEST, LOGIN_SUCCESS, R01: LOGIN
7	testuser@mail.com	password	Upon login button click, capture the database query message in wireshark. Verify that no plaintext password can be seen.	ENCRYPTED_PASSWORD

Reservation Page:

Test Case	Input	Expected Output	Requirements Verified
1	None	Upon reservation page load, All user current reservations are viewable	DISPLAY_UPCOMING_RESERVATIONS, R02: VIEW_CURRENT_RESERVATIONS

New Reservation Page:

Test Case	Date Range	Passengers	Type	Locations	Expected Output	Requirements Verified
1	12/31-1/15	10	1	1, 4	No reservations or price displayed	NEW_RESERVATION_FIELDS, DISPLAY_PRICE
2	12/25-12/31	1	2	1, 1	Upon see availability button click, valid rental options will be displayed with the correct price.	DISPLAY_PRICE, R03: CREATE_NEW_RESERVATIONS
3	12/25-12/31	1	2	4, 4	Upon selecting the book button, after the availability button, an error message is displayed because no cars can be booked at this location currently	SEND_BOOKING_VALIDATION, BOOKING_RESPONSE_FAILURE
4	12/25-12/31	1	2	1, 1	Upon selecting the book button, after the availability button, the user is informed and booking is displayed in upcoming bookings	SEND_BOOKING_VALIDATION, BOOKING_RESPONSE_SUCCESS, R04: BOOK_NEW_RESERVATIONS

Database:

Test Case	Input	Expected Output	Requirements Verified
1	None	After manually verify database type. The database should be PostgreSQL	DATABASE_TYPE
2	“Select * from Cars;”	The database should return results within 5 seconds	RESPONSE_TIMING
3	“Select * from Customers;”	All password information should be hashed and non human readable	STORE_ENCRYPTED_PASSWORD
4	“Insert into Vehicles (make, model, year, color, mileage, price, priceclass, available, location) VALUES *” *input random values	Upon user inspection of the database, the inputted vehicle should be seen	R05: ADD_NEW_VEHICLE_INFORMATION
5	“Update Vehicles set *“ *update random field	Upon user inspection of the database, the update should be seen to the inputted car	R07: UPDATE_CURRENT_VEHICLE_INFORMATION
6	“Delete from Vehicles where vehicleId = *“ *input created id	Upon user inspection of the database, the inputted car should be removed	R06: REMOVE_CURRENT_VEHICLE_INFORMATION