ADM : 20/03930

UNIT : FINAL YEAR PROJECT I

COURSE : BSD

UNIT CODE : BSD 3107

Test Plan Introduction

This section provides an overview of the entire test document, outlining both the test plan and the procedure for the Bus reservation system.

Goals and objectives

The overall goals and objectives of the test process are as follows:

- To ensure the Kimathi Bus Reservation system functions accurately, efficiently and reliably.
- To verify that the system meets the specified requirements and provide a seamless and satisfactory experience for travelers.
- To identify and rectify any defects to improve the overall system performance and user experience.

Statement of Scope

The scope of software testing for the Kimathi Bus reservation system includes the following functionality:

1. Reservation and booking management- testing the ability of travelers to make bookings to their preferred destinations, travelling class & no of seats reserved.

2. Contact details management- testing the system's ability to process filled in contact details, full names, phone contact/ e-mail addresses, and gender for a particular traveler.

3. Billing and payment management- testing the ability of the system to process confirmed mode of payments done, together with the transaction codes and amount payable for a given destination.

Exclusions:

- The testing will not cover third-party integration with external systems.
- Hardware-specific testing, such as compatibility with specific devices, will not be included in this test plan.

Major constraints

The following major constraints will impact the manner in which the software is tested:

- Time constraint: the testing must be completed within the specified project timeline to meet the planned system deploy time.
- Resource limitation: the availability of testing resources, including testing environments and personnel may affect the depth and scope of testing.
- Operational impact: testing activities should not disrupt the normal bus booking operations or negatively affect traveler's experience during the testing process.

Software to be tested

The software to be tested is the Kimathi Bus Reservation System. Exclusions are noted explicitly, which includes any third-party integration.

Testing strategy

The overall testing strategy for the Kimathi Bus Reservation System is as follows:

A) Unit testing

The strategy for unit testing is as follows:

- Identify software components to undergo unit testing based on their critically and complexity.
- Unit testing will focus on individual units of code to ensure they work as expected in isolation.
- The criteria for selecting components for unit tests will be based on their importance in critical functions and areas prone to errors.

B) Integration testing

The integration testing strategy is as follows:

- Integration testing will verify the interactions between different software functions and components.
- Integration testing will be performed based on the order of integration by software function, ensuring that key functions are integrated and tested first.

C) Validation testing

The validation testing strategy is as follows:

- Validation testing will ensure that the entire system works correctly and meets the specified requirements.

- The order of validation will be conducted by software function, ensuring that each function is validated and its integration with other functions is tested.

D) **High- Order testing**

The high order testing strategy is as follows:

- High-order testing will cover various aspects such as recovery testing, security testing, stress testing, performance testing, alpha/beta testing.

- Responsibility for each type of high-order test will be assigned to specialized testing resources or teams .

Testing resources and staffing

The testing resources and staffing for the Kimathi Bus

Reservation system are as follows:

- Specialized testing resources, including testers with expertise in different types of testing (unit testing, integration testing) will be assigned.

- Testing team members will be responsible for different testing phases, ensuring efficient and effective testing.

Test work products

The work products produced as a consequence of the testing strategy include: -

- Test plans for each testing phase, including unit testing, integration testing, validation testing and high-order testing.

- Test cases and test scripts for each test phase.

- Test reports documenting the test results, defects found and corrective actions taken.

Test Record Keeping

Mechanisms for storing and evaluating test results will be as follows:

- A centralized test repository will be used to store test cases, test scripts and test reports.

- Test logos will be maintained to track the progress and results of each testing phase.

Test Metrics

The following test metrics will be used during the testing activity:

- Test coverage metrics to measure the percentage of code tested.
- Defect density to assess the number of defects found per unit code.
- Test execution metrics to measure the time taken to execute each testing phase.

Testing tools and Environment

The test environment for the Kimathi Bus Reservation System will include:

- Testing tools for test automation, defect tracking and performance testing.
- Simulators to simulate different scenarios and test functionalities.
- Specialized hardware and devices for compatibility and performance testing.

Test Procedure

This describes the detailed test procedure, including test tactics and test cases for the Kimathi Bus Reservation System.

Software to be tested

The software to be tested is the Kimathi Bus Reservation System. Exclusions are noted explicitly which includes any third-party integration.

Unit test Cases

Component:

Generate order reference

Travelling and booking info

Contact info

Payment & verification Verify order details

No stubs or drivers are required for unit testing of this component.

Test Cases for Component

Test case 1: Booking info & travelling
Description: Verify that traveler makes bookings for seats for a vehicle to a particular destination during a given duration.

Test steps:

1. Select the preferred destination.

2. Choose type of travelling class.

3. Reserve No of seats.

4. Set the date of travel.

Expected result: The system should process the booking request and proceed to next page.

Test case 2:

Contact Info

Description: Verify that traveler gives correct contact info as required, provide mobile number or e-mail address and gender.

Test steps:

1. Give full name.

2. 2. Fill mobile contact or email-address.

3. Choose gender.

Expected result: The system should process contact information filled and proceed to next unit.

Test case 3:

Payment & verification

Description: Verify that traveler gives the correct method of payment with the transaction ID.

Test steps:

1. Choose mode of payment.

2. Give code of the transaction.

Expected result: The system should process the payment made for the class of seats reserved.

Test case 4:

Verify order details

Description: Confirm the detail already filled in.

Test steps:

1. Confirm details.

Expected result: The system should confirm details and prints a ticket.

Purpose of Tests for Component

The purpose of conducting the unit tests for component is to ensure the individual functionalities related to travelling & booking details, contact info, payment & verification and verify order details work correctly and independently.
These tests will verify the correctness and the overall functionality of each component.

Expected results for Component

test case 1: The traveler successfully confirms travel details

test case 2: The traveler's contact info is stored.

test case 3: payments & verification are established

test case 4: verification of order details supplied.

Integration Testing

Testing procedure for integration:

Integration testing will involve testing the interactions between different software functions and components related to travelling and booking details, contact info, payment and verification and verify order details.
The integration testing procedure will follow a sequential order to ensure key functions are integrated and tested first.

Stubs and drivers required:

No stubs or drivers are required for integration testing.

Test Cases and their Purpose: -

Integration test cases will be designed to test the interactions and data exchange between the different components related to traveler functionality. The purpose of

these test cases is to verify the seamless flow of information and functionality when different components work together.

Expected Results:

The integration test cases should result in successful data exchange and functional interactions between various components, without any system errors or failures.

Validation testing

Testing Procedure for Validation:

Validation testing will involve testing the entire Kimathi Bus Reservation System to ensure it meets all specified requirements and provide appropriate user experience. The validation testing procedure will cover end-to-end user journeys and scenarios.

Test Cases and their Purpose:

Validation test cases will be designed to represent typical user interactions and workflow in the system. The purpose of these test cases is to validate the entire system functions correctly and satisfies user requirements.

Expected results: The expected results for validation test cases should demonstrate that the system works as a whole and provides a satisfactory experience for travelers.

Pass Criterion for all validation tests:

The validation test cases will have specific pass/fail criteria defined based on user requirements and expected system behavior.

High-Order testing (System testing)

High- Order Testing Procedure:

High-Order testing involves conducting various tests to evaluate the overall system performance and behavior.

High-Order Test Methods:

1. Recovery testing: test the system's ability to recover from failures, crashes or unexpected errors.

2. Security testing: Verify the system's security measures to protect traveler data and unauthorized access.

3. Stress testing: Evaluate the system's performance under heavy load or stress condition.

4. Performance testing: Assess the system's overall performance, response time and resource usage.
5. Alpha/beta testing: Conduct real-world testing with a limited number of users to gather feedback and identify potential issues before full release.

Pass/Fail Criteria:

Define the pass/fail criteria for each high-order test method based on the system's requirements and performance expectations.
Specialized testing resources with expertise in unit testing, integration testing, validation testing, high-order testing and system testing will allocated.

Staffing:

The team will consist of qualified testers, test analysts, and a test manager responsible for overseeing the entire testing process.

Test Work Products

- The work products produced as a consequence of the testing procedure include:

- Test plans and test cases for each testing phase

- Test reports documenting the test results, defects found and corrective actions taken

Test record keeping and record

A centralized test repository will be used to store test cases, test scripts and test reports.

Test

Log : A log test will be maintained to keep a chronological record of all tests conducted and their results.