**ITWorks – Test Plan**

**Project name:** **SAAirline Ticketing**

**Author : Travis Griggs**

Version: 1.0

Date: 5/12/2024

Status: Draft

**For more information**

|  |  |
| --- | --- |
| **Project Contact:** |  |
| Name: Travis Griggs  Email: Travis.Griggs@students.tafesa.edu.au |  |

**Revision History:**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | **Date** | **Author(s)** | **Change Description** |
| 1.0 | 12/11/2024 | Travis Griggs | *First draft.* |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

TABLE of Contents

[1. Introduction 4](#_Toc521390588)

[1.1. Test Plan Objectives 4](#_Toc521390589)

[2. Scope 4](#_Toc521390590)

[3. Test Strategy 4](#_Toc521390591)

[3.1 System Test 4](#_Toc521390592)

[3.2. Unit Test 4](#_Toc521390593)

[3.3. Security Test 4](#_Toc521390594)

[3.4. Automated Test 4](#_Toc521390595)

[3.5. Stress, Volume Test, Performance Test 4](#_Toc521390596)

[3.6. Recovery Test 5](#_Toc521390597)

[3.7. Documentation Test 5](#_Toc521390598)

[3.8. Beta Test 5](#_Toc521390599)

[3.9. User Acceptance Test 5](#_Toc521390600)

[4. Environment Requirements 5](#_Toc521390601)

[5. Test Schedule 5](#_Toc521390602)

[6. Control Procedures 5](#_Toc521390603)

[6.1 Reviews 5](#_Toc521390604)

[6.2 Bug Review meetings 6](#_Toc521390605)

[6.3 Change Request 6](#_Toc521390606)

[6.4 Defect Reporting 6](#_Toc521390607)

[7. Functions To Be Tested 6](#_Toc521390608)

[8. Resources and Responsibilities 6](#_Toc521390609)

[8.1. Resources 6](#_Toc521390610)

[8.2. Responsibilities 6](#_Toc521390611)

[9. Deliverables 7](#_Toc521390612)

[10. Suspension / Exit Criteria 7](#_Toc521390613)

[11. Resumption Criteria 8](#_Toc521390614)

[12. Dependencies 8](#_Toc521390615)

[12.1 Personnel Dependencies 8](#_Toc521390616)

[12.2 Software Dependencies 8](#_Toc521390617)

[12.3 Hardware Dependencies 8](#_Toc521390618)

[12.3 Test Data & Database 8](#_Toc521390619)

[13. Risks 8](#_Toc521390620)

[13.1. Schedule 8](#_Toc521390621)

[13.2. Technical 8](#_Toc521390622)

[13.3. Management 8](#_Toc521390623)

[13.4. Personnel 8](#_Toc521390624)

[13.5 Requirements 9](#_Toc521390625)

[14. Tools 9](#_Toc521390626)

[15. Documentation 9](#_Toc521390627)

[16. Approvals 9](#_Toc521390628)

# Introduction

The SAAirline Ticketing application is a C# class library developed for handling flight bookings, capturing customer information, and calculating discounts based on membership. The system is currently under development, with only the core business classes available for testing.

## Test Plan Objectives

* Establish activities required to conduct unit tests on business logic classes.
* Communicate test plan and deliverables with stakeholders.
* Define the scope, strategy, and responsibilities for conducting unit tests.
* Document the process to verify functionality, detect defects, and manage defect tracking.

# Scope

Testing aims to ensure that the available business logic classes meet functional requirements for bookings, discounts, seat availability, and invoicing. Tests will be conducted on individual classes and methods without integration or UI testing, as the input/output screens are unavailable.

# Test Strategy

The test strategy for the SAAirline Ticketing project is designed to verify that the core functionalities of the class library meet the expected requirements.

## System Test

N/A

## 3.2. Unit Test

Unit testing will be the primary testing method used, focusing on the SAAirline Ticketing class library’s methods and constructors. Each method will be tested independently using NUnit to ensure they meet expected outcomes under a variety of input conditions.

## 3.3. Security Test

N/A

## 3.4. Automated Test

N/A

## 3.5. Stress, Volume Test, Performance Test

N/A

## 3.6. Recovery Test

N/A

## 3.7. Documentation Test

N/A

## 3.8. Beta Test

N/A

## 3.9. User Acceptance Test

N/A

# 4. Environment Requirements

* Platform: Windows
* Operating System: Windows 10 or later
* Memory: Minimum 4GB RAM
* Processor: Intel i5 or equivalent
* IDE/Framework: Visual Studio with NUnit for C# unit testing

# 5. Test Schedule

This section covers the test schedule on different test type. Only include the type of tests apply to your situation. Put N/A if it is not applicable e.g.

* Ramp up / System familiarization N/A - xx/xx/xxxx
* Unit Tests 12/11/2024 - 12/11/2024
* Integration Tests (if in scope) xx/xx/xxxx - xx/xx/xxxx
* User Acceptance Test (if in scope) xx/xx/xxxx - xx/xx/xxxx

# 6. Control Procedures

## 6.1 Reviews

Team reviews will be held for requirements, test case, and defect report reviews.

## 6.2 Bug Review meetings

Weekly meetings to discuss reported defects, with the development team providing updates on fixes.

## 6.3 Change Request

Changes to test requirements will be reviewed by the Change Control Board (CCB) for impact analysis.

## 6.4 Defect Reporting

Defects are documented in the tracking system, with updates provided by developers on fix status.

# 7. Functions To Be Tested

The following is a list of functions that will be tested:

* Customer Information Management: Add and retrieve customer information
* Book Seats: Validate seat allocation based on class (economy/first) and availability.
* Set Discounts: Test discounts for different membership levels (Global, Asia).
* Invoice Generation: Ensure invoice reflects correct seat numbers, discounts, and total charges.

**Test case design table**

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case ID | Function | Method Being Tested | Description |
| TC 1 | Invoice Management | Invoice.setNumberOfSeats() | Verify that the number of seats is accurately set on the invoice. |
| TC 2 | Invoice Management | Invoice.setPriceCode() | Confirm that the price code is correctly set for calculating the seat charge. |
| TC 3 | Invoice Management | Invoice.getDiscount() | Validate correct discount is applied based on membership type (e.g., AsiaWorld, GlobalWorld). |
| TC 4 | Invoice Management | Invoice.getCharge() | Confirm that the charge is calculated correctly based on the price code for the seat class. |
| TC 5 | Seat Management | Seat.bookSeats() | Ensure that booking seats updates availability correctly. |
| TC 6 | Activity Management | Activity.bookSeats() | Validate booking seats in Activity to confirm customer assignment and seat availability. |
| TC 7 | Customer Information | Customer.Name (null value test) | Verify that assigning a null value to the customer name is handled properly. |
| TC 8 | Invoice Management | Invoice.setTheCust() | Check that an existing Customer object can be assigned to an invoice without issues. |
| TC 9 | Seat Management | Seat.bookSeats() (exception test) | Validate error handling when booking seats with invalid numbers (e.g., negative values). |
| TC 10 | Invoice Management | Invoice.setRowNum() (exception test) | Test row setting for out-of-range row numbers to confirm appropriate error handling. |

**Requirements Validation Matrix**

|  |  |  |
| --- | --- | --- |
| Requirement ID | Requirement Description | Test Case ID(s) |
| RQ1 | The system shall allow customers to add/update their information. | TC 7, TC 8 |
| RQ2 | The system shall enable customers to book seats based on availability. | TC 4, TC 5, TC 9 |
| RQ3 | The system shall handle seat booking requests with input validation. | TC 9, TC 10 |
| RQ4 | The system shall calculate and display the correct discount for members. | TC 3 |
| RQ5 | The system shall apply pricing based on selected seat type and membership. | TC 1, TC 2, TC 3 |
| RQ6 | The system shall accurately retrieve booking details associated with customers. | TC 6 |
| RQ7 | The system shall prevent null entries for required fields like customer name. | TC 7 |

# 8. Resources and Responsibilities

This section specifies when testing will start and end.

## 8.1. Resources

The test team will consist of:

* 1 x test manager (Jake Brown)
* 1 x testers (Travis Griggs)

## Responsibilities

|  |  |
| --- | --- |
| Test Manager | Responsible for project schedules and the overall success of the project. Participate on Change Control Board (CCB). |
| Testers | Responsible for performing the execution of different kind of tests. |
| Product owner | Will coordinate staff to perform Beta and User Acceptance testing (e.g. ICT manager). \*\*Note: For unit testing, the product owner does not need to be involved. |

# Deliverables

|  |  |  |
| --- | --- | --- |
| **Deliverable** | **Responsibility** | **Completion Date** |
| Develop Test cases | Testers | 06/11/2024 |
| Test Case Review | Test manager, Testers | 12/11/2024 |
| Develop test scripts / procedures | Testers | 16/11/2024 |
| Requirements validation matrix | Test manager | 12/11/2024 |
| Execute test scripts / procedures | Testers | 18/11/2024-21/11/2024 |
| Complete defect report | Everyone testing the product | On-going |
| Document and communicate test status/coverage summary report | Test manager | Weekly |
| Execute Beta tests | Product owner - delegate | 22/11/2024-23/11/2024 |
| Document and communicate Beta test status/coverage | Product owner - delegate | 24/11/2024 |
| Execute User Acceptance tests | Product owner - delegate | 26/11/2024 – 28/11/2024 |
| Document and communicate Acceptance test status/coverage | Product owner - delegate | 28/11/2024 |
| Final Test Summary Report | Test manager | 29/11/2024 |

# 10. Suspension / Exit Criteria

This section specifies when the testing process can be aborted.

* Code with critical defects limits test execution.
* Necessary resources (environment or personnel) are unavailable.

# Resumption Criteria

Testing resumes once any critical defect is resolved or required resources are reinstated.

# Dependencies

## Personnel Dependencies

Testers and developers need close collaboration to ensure quick defect resolutions.

## Software Dependencies

## C# class library and NUnit framework.

## 12.3 Hardware Dependencies

Workstations with required system specifications.

## Test Data & Database

**Test Data:** Specific inputs for each test case, covering typical, boundary, and invalid inputs.

**Storage for Results:** Structured tracking of test outcomes and defect logs.

# Risks

## 13.1. Schedule

A delay in test execution could impact delivery timelines due to the fixed milestone dates. Effective project management is crucial to ensure deadlines are met.

## 13.2. Technical

If testing fails, project completion will be delayed. Critical defects have high priority for fixes.

## 13.3. Management

Insufficient support could impact resources and timelines. Management provides resources as needed and supports timeline adjustments.

## 13.4. Personnel

Tester turnover could delay progress. To mitigate this, it is important to cross-train team members and maintain a pool of backup testers.

## 13.5 Requirements

Changes in requirements could affect testing timelines. Change Control Board approval for requirement adjustments.

# 14. Tools

* **NUnit Framework**: Core tool for automating unit tests.
* **Visual Studio**: IDE for test development and debugging.
* **Defect Tracking System**: Logs and monitors defects to manage resolution.
* **Documentation Tools**: Microsoft Word and Excel for test plan and report documentation.

This setup supports efficient testing, defect management, and documentation of test results.

# Documentation

This section lists the documents that will be available at the end of the test phase:

* Test Plan
* Test Cases
* Requirements Validation Matrix
* Defect reports
* Test Summary Report

# 16. Approvals

|  |  |  |
| --- | --- | --- |
| **Name** | **Role** | **Date** |
|  |  |  |
| 1. **Travis Griggs** | Tester | xxxxxxxx |
|  |  |  |
| 1. **Jake Brown** | Test manager | xxxxxxxx |
|  |  |  |
| **3.** | ICT manager (if applicable) | xxxxxxxx |