Test Plan for Airline Reservation System

March 14, 2023

FIT5171 Team 15

Shixin Huang - shua0098@student.monash.edu
Lingxiao Meng - lmen0026@student.monash.edu
Yuntao Zhang - yzha0870@student.monash.edu

Testing Objective

The system integration test of the flight booking system should validate from the functional requirement that:

1. Query airplane information function is supported
2.Purchased ticket information display
3. Support buy direct flight tickets
4. Support buying two tickets with transfer flight
5. The choose ticket function is good
6. Shows the direct flight from city 1 to City 2
7. show the flight to the destination city
8. shows the flight information according to the flight number
9. displays all available tickets from the Ticket collection
10. Get the ticket information by its id
11.50% sale for children under 15
12.100% sale for people older than 60
13.12% Service tax

Testing Scope

The system integration test of the flight booking system will include airplane, flight, flight collection, passenger, ticket, ticket system, and ticket collection

Test approach

(including any assumption/constraints, software requirements to be tested, testing tools)
Assumptions

1. The working code needs to be successfully created and uploaded to GitLab.
2. All requirements must be understood by testers and developers.
3. All testers need to have a computer and configure the Java language and editor(IntelliJ).
4. The Airline Reservation System will take distributed testing for each unit.
5. The Airline Reservation System test has a standard test time. Different units have different test durations.

Constraints

1. Extend the testing cycle to address new vulnerabilities that arise after the product is repaired.
2. Publish multiple test samples for user experience and error correction.
3. Less testing experience has led to an extension of the work cycle.

Software requirements

The user requirements presented in the Testing Objective will be fully incorporated into the testing process.

Testing tools

1. Test Manager
2. Defect Tracker

Test Plan

including testing team and test environment)

Test team

Name	Title	Level of involvement	Responsibility
Shixin Huang	Team Leader/ Independent Test	12 hrs/wk	Deploy a team test plan and assign team task work. Report and improve the testing process.
Lingxiao Meng	Independent Test Team	12 hrs/wk	Participate in software testing, and assist in managing and arranging team testing plans. Design a test case and create test data.

Yuntao Zhang	Independent Test Team	12 hrs/wk	Participate in software testing, Design a test case and create test data.

Test Environment
Hardware
All test cases will be executed on the Development Server in the QA database environment.

(1) Asus
AMD Ryzen 9 5900HX with Radeon Graphics 3.30 GHz
16.0 GB RAM

(2) Macbook Pro 13
Apple M2 chip
8GB to 24GB unified memory
256GB to 2TB storage14

(3) Macbook Air
Apple M1 chip
8GB or 16GB unified memory
256GB to 2TB storage14

Network
• LAN
- Synoptic 810 10Base-T Ethernet Concentrator
- Category 5 cables to meet 10Base-T specifications

Software
• The Airline Reservation System
- Mac system
- MS Windows 10 Pro operating system
- MS Windows 11 Pro operating system

Testing procedure

Test Cases

In the testing process, a specific set of pre-defined test cases will be executed by the tester for testing each system feature, business process or requirement. Each test case comprises a set of actions to be performed, followed by the expected results. The tester evaluates the outcome by comparing the observed results with the expected results. If they match, the "pass" column is checked, and if they don't match, the "fail" column is checked.

Test Num	Reason for Test/ Purpose for Test	Input	Input Data	Expected Output

ber				
0001	Verify valid passenger information	passenger valid email	String email = "john@example.com";	Return valid passenger details (passenger.getEmail()="john@example.com";)
0002	error message for invalid passenger information	Invalid passenger email	String email = "invalid_email";	Error message displayed("Invalid email address format")
003	test the boundary condition for Ticket class saleByAge method	boundary conditions of age condition (age >=60;)	passenger.setAge(60);	The ticket price should be 0 ticket.getPrice()=0;
004	Negative testing for ticket class setPrice()	Negative ticket price	int price = -100;	Error message displayed("Ticket price cannot be less than zero.")
005	Null points constraint test for Flight class setCode()	No input code value	flight.setCode("")	Error message displayed("Flight code cannot be null.")

Order of Testing

The testing sequence will be primarily determined by the build order, and within each build, the order of testing will adhere to the following order.

Build 1	Build 2	Build 3
1. Table Maintenance 2. Create Datasets 3. Datasets validation 4. System Ease of Use 5. System Performance	1. Flight data entry 2. Check FlightCollection 3. System Ease of Use 4. System Performance 5. Data Recovery	1. Ticket choose 2. Ticket purchase 3. Check Ticket price 4. Check TicketCollection 5. System Ease of Use 6. System Performance