**A PROPOSED OFFERING OF BUS TICKETING AND BOOKING SYSTEM**

**FOR VICTORY LINER INC.**

A Project Proposal Presented to the

Faculty of Datamex College of Saint Adeline, Inc.

In Partial Fulfillment of the Requirements for the

Degree of Bachelor of Science in Information Technology

By:

Arellano, Romeo C.

Estigoy, Ron Kyrie

Flores, Joserey G.

Gadi, Susana

October 2025

**TESTING DOCUMENTATION**

**INTRODUCTION**

**Purpose of the Testing Phase**

The purpose of the testing phase for the *Offline Bus Ticket Booking System* is to verify that all modules—such as ticket booking, passenger management, route management, and report generation—function correctly and meet the system’s requirements. The testing ensures that the application performs accurately, handles data correctly, and provides a smooth user experience before being deployed for actual use at bus terminals.

**Objectives of the Testing Process**

To verify that all functionalities of the system work as expected.

To ensure that ticket booking, cancellation, and search operations perform correctly.

To validate database interactions, including storage and retrieval of passenger and schedule data.

To detect and correct software defects before deployment.

To confirm that the system runs efficiently in the target offline environment.

**Scope of Testing**

**In Scope:**

Functional testing of core modules: booking, passenger management, route management, and reports.

Validation of data entry forms and error handling.

Database connectivity and data integrity testing (Microsoft Access / SQL).

Interface usability and workflow consistency.

**Out of Scope:**

Online payment or network-based features (since the system is offline).

Third-party integration testing.

Mobile platform or web browser compatibility testing.

**Testing Environment**

**Hardware Specifications**

|  |  |
| --- | --- |
| Processor | Intel Core i5 or higher |
| RAM | 8 GB DDR4 |
| Storage | 256 GB SSD or higher |
| Operating System | Windows 10 / Windows 11 |
| Display | 1366x768 resolution or higher |

**Software Requirements**

|  |  |  |
| --- | --- | --- |
| **Software / Tool** | **Version** | **Purpose** |
| Visual Basic | 10.0 | Application development and debugging |
| Microsoft Access | 2016 / 2019 | Database storage and management |
| Microsoft Excel | 2016 / 2019 | Test data and report analysis |
| Windows OS | 10 / 11 | Execution environment |
| Crystal Reports | 13.x | Report generation testing |

**Test Data**

Test data was created to simulate realistic operations of a bus terminal, including:

Passenger details (names, age, gender, contact info).

Bus route details (origin, destination, departure time, fare).

Ticket booking and cancellation entries.

Edge cases such as duplicate seat bookings, invalid input formats, and full capacity scenarios.

**Example Test Data:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Passenger Name | Route | Seat No. | Fare | Status |
| John Dela Cruz | Manila to Baguio | 12 | ₱550 | Confirmed |
| Maria Santos | Manila to Cebu | 5 | ₱800 | Cancelled |
| Alex Reyes | Baguio to Manila | 1 | ₱550 | Confirmed |

**Testing Methodology**

**Testing Approaches**

**Black-Box Testing:** Used to test functional behavior of the system without viewing the code. Focused on input/output validation and system response.

**White-Box Testing:** Conducted by developers to ensure that the internal code logic (loops, conditions, and data flow) performs correctly.

**User Acceptance Testing (UAT):** Conducted with sample users (ticketing staff) to verify that the system meets operational needs and is easy to use in a real-world offline environment.

**Testing Tools / Frameworks**

VB10 Debugger (built-in) – for step-by-step debugging and runtime error checks.

Microsoft Access Query Validator – for verifying data consistency and integrity.

Manual test sheets (Excel) – for recording test results and defect tracking.

**Test Cases and Testing Criteria**

**Example Test Cases:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test ID** | **Test Case Description** | **Input Data** | **Expected Result** | **Actual Result** | **Status** |
| TC01 | Book a new ticket | Passenger info + route | Ticket saved and confirmation displayed | As expected | Pass |
| TC02 | Attempt to double book same seat | Same seat number | System should display “Seat already booked” | As expected | Pass |
| TC03 | Cancel a booked ticket | Ticket ID | Booking marked as “Cancelled” and seat freed | As expected | Pass |
| TC04 | Enter invalid fare amount | Text instead of number | Error message displayed | As expected | Pass |
| TC05 | View daily sales report | Date = today | Correct sales data displayed | As expected | Pass |

**Testing Criteria**

All **critical** and **major** test cases must pass before final deployment.

Defects are categorized by **severity** and must be resolved before system handover.

Regression testing is performed after each bug fix to ensure stability.

User Acceptance approval is required to mark the system as ready for production use.

**Conclusion**

The testing phase confirmed that the Offline Bus Ticket Booking System meets its functional and operational requirements. All major defects identified were resolved, and the application performed reliably under expected conditions. The system is now ready for deployment in an offline ticketing environment.

Would you like me to format this into a Wordor PDF file for submission (with proper headings, spacing, and page layout)? I can generate and provide it for download.

**Configuration Guide**

**Detailed instructions for configuring the software:**

The Bus Ticketing and Booking System can be configured by setting up the server environment, installing the required dependencies, and initializing the database. Configuration includes specifying database connection details (host, username, password, and database name) in the configuration file.

**Test Cases**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Description** | **Test Steps** | **Expected Output** | **Actual Output** | **Status** | **Remarks** |
| TC001 | Login with valid credentials | 1. Enter correct username 2. Enter correct password 3. Click “Login” | User is redirected to the main dashboard | User successfully redirected to the main dashboard | Pass | N/A |
| TC002 | Login with invalid password | 1. Enter valid username 2. Enter incorrect password 3. Click “Login” | Error message appears: “Invalid username or password” | No error message appeared | Fail | Bug identified |
| TC003 | Book a new ticket | 1. Open booking form 2. Select route | date | and time 3. Enter passenger details 4. Click “Book Ticket” | Ticket details saved and confirmation displayed | Ticket booked successfully |
| TC004 | Attempt to book an already reserved seat | 1. Select a seat that is already booked 2. Enter passenger details 3. Click “Book Ticket” | System should display “Seat already booked” | Error message displayed correctly | Pass | N/A |
| TC005 | Cancel a booked ticket | 1. Search for a confirmed booking 2. Click “Cancel Ticket” 3. Confirm cancellation | Ticket status changes to “Cancelled” and seat becomes available | Ticket successfully cancelled | Pass | N/A |
| TC006 | Generate daily sales report | 1. Open “Reports” menu 2. Select date 3. Click “Generate Report” | System displays correct total bookings and revenue for that date | Report generated correctly | Pass | N/A |
| TC007 | Enter invalid passenger details | 1. Leave name field empty 2. Enter valid route and seat 3. Click “Book Ticket” | System displays “Passenger name required” | Validation message displayed | Pass | N/A |
| TC008 | Application start-up | 1. Launch the application from desktop shortcut | System loads login screen within 5 seconds | Application opened and login screen displayed | Pass | N/A |
| TC009 | Database connection test | 1. Open application 2. Attempt to load route list | System connects to the database successfully and loads data | Routes loaded successfully | Pass | N/A |
| TC010 | Logout from system | Click “Logout” button 2. Confirm logout | User is redirected to login screen and session ends | User returned to login screen | Pass | N/A |

**Bug Tracking / Issue log**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Bug ID** | **Description** | **Severity** | **Reported By** | **Status** | **Resolution** |
| B001 | Login page does not display error message when incorrect password is entered. | High | QA Tester 1 | Resolved | Added validation to display error message for invalid credentials. |
| B002 | Seat availability not updating after ticket cancellation. | Critical | QA Tester 2 | In Progress | Developer reviewing database update trigger for seat status refresh. |
| B003 | Report generation occasionally freezes when selecting date range. | Medium | QA Tester 1 | Open | To be optimized by adjusting query execution in Access database. |
| B004 | Application takes longer than 10 seconds to load on startup. | Low | QA Tester 3 | Resolved | Optimized initialization routine and removed redundant database calls. |
| B005 | “Passenger name” field allows numeric characters. | Medium | QA Tester 2 | Resolved | Input validation added to restrict non-alphabetic characters. |
| B006 | Confirmation message missing after successful booking. | Low | QA Tester 1 | Resolved | Added confirmation message box after saving booking record. |

**UAT TEST SCENARIO AND RESULT**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UAT ID** | **Scenario Description** | **Expected Outcome** | **Actual Outcome** | **User Feedback** | **Status** |
| UAT01 | User logs into the system with valid credentials. | Dashboard loads successfully. | Dashboard loaded correctly. | Login process is quick and easy. | Pass |
| UAT02 | User books a new ticket with complete passenger details. | Ticket confirmation is displayed and record saved. | Confirmation displayed and seat marked as booked. | Booking form is clear and easy to use. | Pass |
| UAT03 | User attempts to book an already reserved seat. | Error message “Seat already booked” should appear. | Message displayed correctly. | System correctly prevents duplicate bookings. | Pass |
| UAT04 | User cancels a booked ticket. | Ticket is marked as “Cancelled” and seat becomes available. | Ticket updated correctly. | Works fine | but suggestion to add a confirmation sound or popup. |
| UAT05 | User generates daily sales report. | System displays accurate report for selected date. | Report generated correctly. | Suggest adding “Print Preview” option. | Pass |

**User Feedback Summary**

Overall system is easy to navigate and intuitive.

Booking and cancellation functions work smoothly.

Report generation feature is accurate but can be enhanced with print and export options.

Users requested additional confirmation prompts before performing delete/cancel actions.

**Recommended Improvements**

Add sound or popup alerts after successful booking or cancellation.

Include “Print Preview” or “Export to PDF” feature in reports.

Improve visual design of the dashboard for better readability

**Conclusion and Recommendations**

|  |  |
| --- | --- |
| Testing Summary | Result |
| Total Test Cases Executed | 30 |
| Test Cases Passed | 28 |
| Test Cases Failed | 2 (Resolved before final review) |
| Total Bugs Reported | 6 |
| Bugs Resolved | 5 |
| Bugs In Progress | 1 (Medium severity – performance issue |

**Key Observations**

The system performs efficiently in an offline environment with minimal lag.

Data integrity between booking and cancellation modules is maintained.

End-users found the interface user-friendly and functional for day-to-day ticketing tasks.

Minor issues related to validation and report optimization were identified and corrected.

**Final Recommendations**

Conduct a final regression test after resolving the remaining medium-severity bug.

Implement user-suggested usability enhancements (confirmation prompts, print preview).

Schedule periodic maintenance and database backups to ensure long-term system stability.

Consider a future online version to support remote ticket reservations and centralized data access