12/11/2024

Test Plan

Tain Ticket Management system

**Group Members:**  
Awais Rafique 22i-2511

Hassan 22i-2434

Azmarr Kashif 22i-2716

**TEST PLAN**

## TEST PLAN IDENTIFIER TMS-1.0

1. **REFERENCES**

None Identified.

## INTRODUCTION:

This is the Master Test Plan for the Train Ticket Reservation System project. This plan addresses all components related to the online train ticket booking process, including both direct functionalities and supporting modules. The primary focus is to ensure that the system provides a reliable, secure, and user-friendly platform for train ticket reservations while maintaining data integrity and transaction accuracy.

The system encompasses three main user interactions:

1. Customer operations (booking, cancellation, and history tracking)
2. Administrative functions (train management and scheduling)
3. System operations (database transactions and session management)

The **testing will be conducted at three levels:**

**Unit Testing**: Individual components and methods

**Integration Testing**: Module interactions and workflow verification

**System Testing:** End-to-end functionality and user acceptance

The testing approach emphasizes white-box testing methodologies, specifically focusing on Statement Coverage and Decision Coverage for critical business logic components. Special attention will be given to:

1. User authentication and authorization
2. Booking transaction integrity
3. Concurrent booking scenarios
4. Payment processing accuracy
5. Data consistency across operations

The estimated timeline for comprehensive testing is eight (8) weeks, with:

**Unit Testing**: 3 weeks

**Integration Testing:** 3 weeks

**System Testing:** 2 weeks

**Critical dependencies include:**

1. Database connectivity and operations
2. Session management functionality
3. Payment gateway integration
4. User interface responsiveness

This test plan will ensure that all core functionalities meet the specified requirements while maintaining system reliability and data accuracy throughout the reservation process.

## TEST ITEMS

The following items will be tested as part of this test plan:

**A. Core Service Components (Version 1.0)**

* TrainServiceImpl.java
* UserServiceImpl.java
* BookingServiceImpl.java
  + These components handle the core business logic and must be thoroughly tested for all possible scenarios.

**B. Database Utility Components (Version 1.0)**

* DBUtil.java: Database connection and operation management
* application.properties: Database configuration settings
  + Critical for system stability and data integrity.

**C. Authentication and Authorization Components**

* TrainUtil.java: Session management and user validation
* UserRole.java: Role-based access control definitions
  + These components ensure system security and proper access control.

**D. Data Model Components**

* TrainBean.java
* UserBean.java
* HistoryBean.java
* BookingDetails.java
  + Responsible for data structure and validation.

**E. Transaction Management Components**

* BookingService interface and implementation
* Transaction tracking and history management
  + Critical for booking integrity and financial accuracy.

**F. Supporting Documentation**

* Database Schema (version 1.0)
* API Documentation (version 1.0)
* User Manual (version 1.0)

**G. Configuration Files**

* web.xml (version 1.0)
* pom.xml (version 1.0)
  + Essential for system configuration and dependency management.

**H. Integration Points**

* Oracle Database (version 19c)
* Servlet Container (Apache Tomcat 8.5)
* JDBC Driver (version specified in pom.xml)

## SOFTWARE RISK ISSUES

Critical system components with potential risks have been identified:

**A. Database Operations**

* Concurrent booking transactions causing data inconsistency
* Database connection failures during critical operations
* Deadlock situations in high-load scenarios
* Data corruption during transaction rollback

**B. Authentication and Security**

* Session hijacking vulnerabilities
* SQL injection possibilities in search operations
* Unauthorized access to admin functions
* Password encryption vulnerabilities

**C. Transaction Management**

* Payment gateway timeout issues
* Incomplete booking transactions
* Double booking of the same seats
* Transaction rollback failures

**D. System Recovery**

* Booking state recovery after a system crash
* Session state recovery
* Database consistency after failure
* Transaction log integrity

**E. Integration Points**

* Oracle database version compatibility
* JDBC driver stability
* Tomcat server configuration issues
* JVM memory management

**F. Data Integrity**

* Seat availability synchronization
* Booking history accuracy
* User profile consistency
* Payment record maintenance

**G. Performance**

* High concurrent user load
* Large dataset query performance
* Session pool exhaustion
* Database connection pool limits

## FEATURES TO BE TESTED

The following features will be thoroughly tested:

**A. User Authentication and Management**

* User registration process
* Login/logout functionality
* Password change mechanism
* User profile management

**B. Train Management**

* Train addition and removal
* Schedule updates
* Seat availability tracking
* Fare calculation

**C. Booking System**

* Ticket booking process
* Payment processing
* Booking confirmation
* Seat allocation
* Booking history tracking

**D. Admin Functions**

* Train schedule management
* User management
* System monitoring
* Transaction management

**E. Database Operations**

* Data consistency
* Transaction integrity
* Concurrent booking handling
* Backup/restore operations

**F. Search Functionality**

* Train search
* Booking history search
* Availability check
* Fare enquiry
  1. **FEATURES NOT TO BE TESTED**

The following features are outside the scope of testing:

**A. UI/UX Design Elements**

* These are purely presentational aspects and don't affect core functionality. HTML/CSS styling and layout will not be explicitly tested.

**B. Browser Compatibility**

* The application is designed for standard web browsers. Specific browser compatibility testing is not part of this scope.

**C. Network Infrastructure**

* The underlying network architecture and hardware infrastructure are not part of the application testing scope.

**D. Third-party Analytics**

* Any external analytics or tracking tools integrated with the system are outside the testing scope.

**E. Report Generation Tools**

* External reporting tools used to analyze booking data are not included in the testing scope. Only the data export functionality will be verified.

**F. Performance Under Extreme Load**

* Stress testing beyond normal operating conditions is not included in this testing phase.

**G. Email Notification System**

* The email delivery system and SMTP server functionality are outside the testing scope.

**H. Print Functionality**

* Physical printing capabilities and printer compatibility are not included in the test scope.

## APPROACH

## 7.1 Testing Levels

## The testing for the Train Ticket Management System will consist of Unit, System/Integration, and Acceptance testing levels. The following approach will be used for each level:

## A. UNIT Testing

## Conducted by the developer and approved by the development team leader.

## Proof of unit testing (test case list, sample output, data printouts, defect information) must be provided to the team leader before acceptance.

## All unit test information will also be shared with the test manager.

## B. SYSTEM/INTEGRATION Testing

## Performed by the test manager and development team leader, with assistance from developers as needed.

## Programs will enter System/Integration testing after resolving all critical defects.

## Programs may have up to two major defects if they do not impede testing and have a workaround.

## C. ACCEPTANCE Testing

## Performed by end users with support from the test manager and development team leader.

## Acceptance testing will run parallel with the existing manual processes (if applicable) for a predefined period after System/Integration testing.

## Programs will enter acceptance testing only after all critical and major defects are resolved.

## Programs may have one major defect if it does not impede testing and has a workaround.

## All critical and major defects must be corrected and verified before final acceptance.

## Distributors (or other stakeholders): will participate in the initial acceptance test phase, with additional participants being added as their systems are verified. Careful coordination will ensure test data does not enter the live production system.

* 1. **Configuration Management/Change Control**
  2. **Movement of Programs**
* Movement of programs between development, testing, and production environments will be controlled through an existing Configuration Management system.
* The same process will ensure version control and tracking for all program changes.
  1. **Testing Environment**
* All unit and initial system testing will occur on the **Development Environment (RED system)**.
* Once the system achieves stability (no critical or major defects outstanding), initial pilot testing will occur on the **Production Environment (BLUE system)** in a parallel mode.
* Controls will be implemented to prevent production file updates during pilot testing.
  1. **Parallel Testing**
* Parallel testing will compare data between the existing manual processes and the new system.
* This approach will identify discrepancies and ensure accurate data migration and processing.
  1. **Change Control Procedures**
* All changes, enhancements, and modifications will follow published change control procedures.
* Any deviations from standard procedures will be documented in the project plan's change control section.

This approach ensures stability, accuracy, and a smooth transition to the new **Train Ticket Management System** while mitigating risks during deployment.

* 1. **Test Tools**

**8.2 Test Tools**

The following tools and methods will be used for testing the **Train Ticket Management System**:

**A. Configuration Management**

* **Git** will be used as the version control system for managing the source code.
* Code changes will be tracked using a Git branching strategy, and Pull Requests (PRs) will ensure thorough code reviews.

**C. Development and Debugging Tools**

* Editing and debugging will be performed using **Eclipse Enterprise addition for developers**
* The integrated debugging tools in these IDEs will assist in identifying and resolving issues during development.

**D. Test Data Management**

* Test data will primarily be created using **SQL PLUS using Oracle DB** to populate the development database.
* Production data will not be modified directly. If required, data will be copied from production to a staging database for testing using SQL commands.

**E. API Testing**

* **Postman** or **Swagger** will be used for testing the system’s APIs to ensure proper data flow and response integrity.

**F. Load and Performance Testing**

* **Junit** or **Locust** will be used to simulate high user traffic and validate the system's performance under stress.

**G. Beta Testing**

* Initial testing will involve beta testers (train staff or selected users) to validate end-to-end scenarios using the system.

**H. Documented Testing Approaches**

* Detailed test cases will be created and documented to ensure comprehensive testing coverage.
* The testing will include:
  + **White-Box Testing** (from Assignment 2):
    - **Statement Coverage**: Ensuring every line of code is executed at least once.
    - **Decision Coverage**: Validating all logical decisions (true/false paths) in the code.
  + **Black-Box Testing** (from Assignment 3):
    - **Equivalence Partitioning**: Dividing input data into valid and invalid partitions to ensure correct system behavior for all.
    - **Boundary Value Analysis**: Testing at the boundaries of input ranges to identify edge case issues.

**8.3 Meetings:**

The following meeting schedule will be followed during the testing phase:

1. **Biweekly Progress Meetings**:
   * The testing team will meet every two weeks to evaluate progress, discuss defect trends, and address problems.
2. **Development Team Coordination**:
   * The test team leader will meet with the development team and project manager on alternate weeks to ensure alignment.
3. **Emergency Meetings**:
   * Additional meetings will be scheduled as required to address critical issues or bottlenecks.

**8.4 Measures and Metrics**

The following metrics will be tracked to monitor the testing progress and system quality:

**By Development Team (During Unit Testing):**

1. Number of defects identified per module, categorized by severity (Critical, Major, Minor).
2. Defect origin (Requirement, Design, Code).
3. Time spent resolving defects, tracked for Critical and Major issues. Minor defects will be reported in totals.

**By Testing Team (During All Testing Phases):**

1. Defects identified by module and severity.
2. Defect origin (Requirement, Design, Code).
3. Time spent investigating defects, tracked for Critical and Major issues. Minor defects will be reported in totals.
4. Number of times a module or feature is submitted for retesting after fixes.
5. Escaped defects found at higher testing levels that should have been caught earlier.

These metrics will be compiled and shared biweekly with the test manager and project team to ensure transparency and continuous improvement in the testing process.

## ITEM PASS/FAIL CRITERIA

The testing process for the **Train Ticket Management System** will be considered complete when the following conditions are met:

1. The system successfully processes ticket bookings, cancellations, and modifications for a continuous period of one month without critical or major defects.
2. The financial reports generated from the system reconcile accurately with historical data obtained from manual ticketing records.
3. The system passes all predefined functional and performance test cases, with all critical and major defects resolved.
4. Stakeholders, including train staff and administrative users, are satisfied with the accuracy and usability of the system.

Once the above criteria are met, the system will be deployed live for production use. Further activations, such as extending the system to additional regions or departments, will occur as needed and will follow a separate verification process for each activation.

**9. SUSPENSION CRITERIA AND RESUMPTION REQUIREMENTS**

The following outlines scenarios where testing may be suspended and the requirements for resumption:

**A. Lack of Test Data**

* If production-like test data (ticketing records, schedules, or user profiles) is unavailable at the start of the testing phase, testing will be delayed until sufficient data is acquired or simulated.
* Testing activities dependent on such data will remain on hold, while independent modules can continue testing.

**B. Infrastructure or Resource Unavailability**

* In case of unavailability of servers, databases, or internet connectivity critical for system operations, testing will be suspended.
* Testing will resume once the necessary infrastructure is restored, and all affected modules are validated.

**C. Delays in Third-Party Integrations**

* If third-party APIs or payment gateways integrated into the system are delayed or unavailable, testing for dependent modules will be suspended.
* Resumption will occur once the third-party service is restored, with focus on validating the integration before proceeding to system-level testing.

**D. Delays in Delivering UI Prototypes**

* If there are delays in delivering UI prototypes or feedback on screen designs, testing for the user interface will be postponed.
* Testing for backend functionalities can continue during this period.

**E. Team or Resource Unavailability**

* If testing personnel or key stakeholders are unavailable due to unforeseen circumstances, testing may be partially suspended.
* Resumption will depend on the availability of resources and a revised testing schedule.

## TEST DELIVERABLES

### Test Plan:

### File format: PDF or Word document

### 2. Test Cases:

### File format: Excel spreadsheet or exported file from a test management tool

### 3. Defect Reports:

### File format: Exported file from Bugzilla (or similar tool) or screenshots compiled in a PDF

### 4. Automation Scripts:

### File format: Folder containing script files, compressed into a ZIP file

### 5. Static Analysis Report:

### File format: PDF or Word document

### 6. Project Source Code:

### File format: Compressed ZIP file of the complete project source code

### 7. Team Contribution Report:

### File format: PDF or Word document

1. **ENVIRONMENTAL NEEDS**

The following resources and setups are required to support the comprehensive testing effort for the **Train Ticket Management System**:

**A. Development and Production System Access**

* Full access to both the development and production environments for testing, data acquisition, and validation.

**B. Database Connectivity**

* A reliable connection to the database server for running queries, retrieving test data, and verifying data integrity during testing phases.

**C. Integration with Payment Gateways**

* A functional and secure integration with third-party payment gateways to test booking transactions, cancellations, and refunds.

**D. Networking Infrastructure**

* Stable internet connectivity and access to network resources for testing remote API integrations and simulating user transactions from various locations.

**E. Access to Backup and Recovery Processes**

* Inclusion in the nightly backup schedule to ensure test data can be restored if needed during testing.

**F. Availability of Test Data**

* A set of production-like data, including schedules, pricing, and user profiles, to be used for simulating real-world scenarios during system and integration testing.

**G. Hardware and Devices**

* Access to test devices like desktops, laptops, and mobile phones with various operating systems to ensure cross-platform compatibility.

**11. STAFFING AND TRAINING NEEDS**

The project requires dedicated personnel and adequate training to ensure successful testing and deployment of the **Train Ticket Management System**:

**A. Staffing Needs**

1. A dedicated tester should be assigned full-time during the system/integration and acceptance testing phases.
   * Initially, a part-time tester will participate in design reviews and test planning, transitioning to full-time approximately four months into the project.
2. If a separate testing resource is unavailable, the project manager or test manager will assume this responsibility.

**B. Training Needs**

1. **Developers and Testers**
   * Training on the system architecture, core functionalities, and API integrations for effective testing.
2. **Customer Service and Administrative Staff**
   * Training on new user interface screens and reports to familiarize them with system operations.
3. **Operations Staff**
   * Comprehensive training on data backup and recovery processes and handling database updates in the production environment.
4. **Third-Party Integration**
   * Training for at least one developer and an operations team member on the configuration and management of third-party payment gateways.
5. **End-User Training**
   * Conduct sessions for ticket booking staff and end-users on using the system, focusing on key functionalities like ticket booking, cancellations, and generating reports.

This structured approach to resource allocation and training ensures that all stakeholders are adequately prepared for system testing and deployment.

## SCHEDULE

Time has been allocated within the project plan for the following testing activities. The specific dates and times for each activity are defined in the project plan time line. The persons required for each process are detailed in the project time line and plan as well. Coordination of the personnel required for each task, test team, development team, management and customer will be handled by the project manager in conjunction with the development and test team leaders.

* 1. Review of Requirements document by test team personnel (with other team members) and initial creation of Inventory classes, sub-classes and objectives.
  2. Development of Master test plan by test manager and test with time allocated for at least two reviews of the plan.
  3. Review of the System design document by test team personnel. This will provide the team with a clearer understanding of the application structure and will further define the Inventory classes, sub-classes and objectives.
  4. Development of System/Integration and Acceptance test plans by test manager and other essential personnel with time allocated for at least two reviews of the plans.
  5. Review of the Detail design document(s) by test team personnel as required. This will provide the team with a clearer understanding of the individual program structure and will further define the Inventory classes, sub-classes and objectives.
  6. Unit test time within the development process.
  7. Time allocated for both System/Integration and Acceptance test processes.

## PLANNING RISKS AND CONTINGENCIES

* 1. The Train Ticket Management System testing process involves three team members working collaboratively to ensure comprehensive coverage of all test cases. A potential risk is the limited availability of testers during critical stages, such as document reviews and Acceptance Testing. Any delays in team participation may result in adjustments to the testing schedule. To address this, all critical reviews and testing milestones will be rescheduled as necessary to ensure that quality and completeness are not compromised.
  2. If all testers are unavailable at a specific point, one team member may act as a temporary representative to maintain testing progress. However, reviews of user interfaces and generated reports will require the approval of all testers to guarantee accuracy and usability. This approach ensures that no key aspect of the system is overlooked, preserving the integrity of the testing process.

**13. APPROVALS**

* 1. The following individuals are responsible for providing approvals throughout the development and testing phases of the Train Ticket Management System:
  2. **Project Sponsor:** Systems Limited
  3. **Development Manager:** Awais Rafique
  4. **Testing Lead:** Azmaar Kashif
  5. **Development Team Lead:** Qasim
  6. **Client Representative:** Ali bin Shahid

This framework ensures clear accountability and structured oversight, facilitating a smooth testing and approval process.