UNIVERSITÉ Concordia UNIVERSITY

INSE6260 Software Quality Assurance

Room Reservation Project

Acceptance Test Plan

04/06/2024

Merlin Duval - 40286249
Yash Khosla - 40232363

1. Overview

1.1. Purpose

The purpose of this Acceptance Test Plan is to outline the procedures and criteria to verify that the developed room reservation software meets the specified requirements as listed in the Software Requirement Specification (SRS). It ensures that the system has been properly implemented and that its utilization is secure regarding university policies.

1.2. Scope

The acceptance testing will cover all functional and non-functional requirements outlined in the SRS. This includes testing the reservation of rooms, modification and removal of reservations, checking room availability, user interface accessibility and security, system performance, reliability, availability, security, maintainability, and portability. It also involves testing the system's compatibility with different operating systems and adherence to university policies and database requirements.

1.3. Output

The output of the acceptance testing process will be a comprehensive evaluation report detailing the results of each test case. It will include information on any deviations from expected behavior, identified defects, and recommendations for further improvements. The report will serve as a basis for determining whether the system is ready for deployment and user acceptance.

2. Testing

2.1. General Approach

Overview: The testing procedure for the University Room Reservation System encompasses a comprehensive evaluation of its functionality, usability, security, and compliance with university policies. It involves thorough testing of various features such as room reservation, modification, and cancellation to ensure a seamless user experience and adherence to specified requirements.

General Approach for Acceptance Testing the University Room Reservation System:

Venues/Sites: Acceptance testing will primarily occur within the university's intranet environment, where the room reservation system will be deployed. This ensures that testing reflects real-world usage scenarios and accurately assesses system performance.

Testing Structure: The testing process will be structured into distinct phases, including user scenario testing, boundary testing, and security testing. Test cases will be grouped based on functional areas to facilitate systematic and thorough evaluation.

Business Units Involved: Various stakeholders within the university, including students, teachers, administrative staff, and IT personnel, will participate in acceptance testing. Their involvement is crucial for validating the system's functionality, usability, and alignment with organizational needs.

Internal and External Resources: The Testing Team will comprise internal personnel with expertise in room reservation processes and software testing methodologies. Additionally, external consultants or auditors may be engaged to conduct an independent assessment of the system's compliance with security standards and regulatory requirements.

Audit Assessment: An audit assessment may be conducted to evaluate the system's adherence to university policies, data protection regulations, and industry best practices. This assessment ensures that the system meets all necessary requirements and mitigates potential risks.

2.1.1. System Testing

Responsibility: System Testing will be the responsibility of the System Supplier, overseen by the System Supplier Project Manager or a designated Test Manager.

Testing Team: System Testing will be conducted jointly by development personnel from the System Supplier and personnel assigned to the project from the Business Unit.

Scope: System and integration testing will be performed to ensure that the entire system functions as intended, aligning with the Design Specifications.

Test Plan: A comprehensive System Test Plan will be prepared by the System Supplier Development Team, outlining the testing approach, methodologies, and test scenarios.

Testing Environment: System Testing will take place in the test environment, which is located on the development server. Workstations and printers located at a specified location will be used for testing purposes.

Data: Testing will be performed against selected live data as well as special-purpose test data, ensuring thorough validation of system functionality.

Volume and Stress Testing: System Testing will include volume testing to evaluate the system's performance with a high number of operations and records processed. Stress testing will also be conducted to assess system behavior under high-frequency operation processing.

Recording Results: Results of tests will be meticulously recorded. In instances where system components do not perform as expected, Test Problem Reports will be raised and registered, initiating the process of issue resolution and corrective action.

System Testing serves as a crucial phase in the software development lifecycle, ensuring that the system functions seamlessly and meets the specified requirements outlined in the Design Specifications.

2.1.2. Acceptance Testing

Responsibility: Acceptance Testing will be the responsibility of the Business Unit, overseen by the Business Unit Test Manager. This ensures that the testing process aligns closely with the business requirements and objectives.

Test Plan: Acceptance Testing will be conducted according to an Acceptance Test Plan prepared by the Business Unit. This comprehensive plan will outline the testing approach, methodologies, test scenarios, success criteria, and reporting procedures. It will serve as a roadmap for executing and evaluating the testing process.

Testing Environment: Acceptance Testing will take place on the production server, utilizing workstations and printers located at a specified location. This ensures that the testing environment closely mirrors the actual operational environment, allowing for realistic testing scenarios and outcomes.

Standard Tests: Standard tests will be established for each program type, including online inquiry, online update, batch processes, standard reports, word processing reports, etc. These standard tests provide a baseline for evaluating the functionality of the system across different modules and processes.

Test Cases: A specific set of test cases will be developed for each function, ensuring comprehensive coverage of all system functionalities. These test cases will encompass various scenarios, inputs, and expected outcomes, enabling thorough validation of the system's behavior.

Interface Files: Tests will be conducted on the importation of various interface files provided by external organizations, ensuring compatibility and seamless integration. This ensures that data exchange between the new system and external systems occurs smoothly and accurately.

Data Migration: Tests for data migration functions will be undertaken to ensure the accurate transfer of data from legacy systems to the new system. This involves validating data mapping, transformation, and integrity to safeguard against data loss or corruption during migration.

Security and Integrity: Specific tests will be conducted to verify security measures, concurrent data access, database integrity, and user group privileges. This ensures that the system maintains data confidentiality, integrity, and availability while adhering to security policies and regulations.

Audit Trail: The audit trail will be monitored to ensure it accurately records all required data changes, providing a comprehensive record of system activities. This facilitates traceability, accountability, and compliance with regulatory requirements.

Performance Criteria: Tests will be conducted against predefined performance criteria detailed in the agreement between the Business Unit and the System Supplier. This includes testing system response times, throughput, scalability, and resource utilization to ensure optimal performance under varying conditions.

User Procedures: User procedures, batch processes, periodic processes, and controls will be tested to ensure they function as intended. This involves validating user workflows, data entry processes, exception handling, and error recovery mechanisms to enhance user experience and productivity.

Backup and Recovery: Backup and recovery procedures will be tested to verify their effectiveness in safeguarding data and ensuring business continuity. This includes testing backup schedules, data recovery processes, and disaster recovery strategies to mitigate risks associated with data loss or system failures.

Documentation Verification: User, technical, and operations documentation will be verified to ensure accuracy and completeness. This includes reviewing user manuals, system documentation, technical specifications, and operational procedures to facilitate system understanding, training, and support.

Integration Impact: The impact of the new system on existing systems will be tested to identify any potential integration issues or conflicts. This involves testing data flows, interfaces, dependencies, and interactions between the new system and existing systems to ensure seamless integration and interoperability.

Acceptance Testing serves as a critical step in the validation process, ensuring that the system meets the Functional Specifications and is ready for deployment in a live operational environment. By meticulously testing the system against predefined criteria and scenarios, Acceptance Testing validates its readiness to support business operations effectively and efficiently.

2.2. Environment

The technical requirements for Acceptance Testing include:

Hardware

- Workstations: Standard workstations with the necessary specifications to run the system.
- Printers: Printers for generating test reports and documents as needed.

Software

- System Software: The system software being tested, including all modules and components.
- Third-party Software: Any third-party software required for the functioning of the system

or for specific test scenarios.
- Testing Tools: Tools for test management, defect tracking, and performance monitoring.
- Interfaces: Any interfaces required for integration testing or data migration testing.

Data
- Test Data: Sets of test data representing various scenarios and conditions to be used during testing.
- Live Data: Selected live data to be used for testing against real-world scenarios.
- Interface Files: Test files provided by external organizations for testing interface functionalities.

Networking
- Local Area Network (LAN): Network connectivity to ensure communication between workstations and servers.
- Internet Access: Access to the internet may be required for certain test scenarios or for downloading updates or patches.

Testing Environment Setup
- Configuration: Configuration of hardware and software components according to the system requirements.
- Installation: Installation of the system software and any required dependencies.
- Data Setup: Loading of test data and configuration of databases for testing purposes.

Security
- Access Control: Implementation of access controls to ensure that only authorized personnel have access to the testing environment.
- Data Security: Measures to ensure the security and integrity of test data and sensitive information.

Backup and Recovery
- Backup Procedures: Procedures for regular backup of test data and system configurations.
- Recovery Procedures: Procedures for restoring the test environment to a previous state in case of failures or errors.

2.3. Resources

Physical Resources

- Accommodation: Adequate space for setting up the testing environment, including workstations, printers, and networking equipment.
- IT Hardware/Software: Workstations, servers, printers, networking devices, and any specialized hardware or software required for testing.

Human Resources

- Departmental Staff: Current departmental staff involved in the project, including testers, developers, project managers, and quality assurance personnel.

- Additional Staff and External Consultants: Additional testers, subject matter experts, consultants, or contractors hired specifically for the testing phase. These may include specialists in performance testing, security testing, or usability testing.

Financial Resources

- Cost and Budget Implications: Funding allocated for testing activities, including the procurement of hardware, software, and additional resources. This may also include expenses related to hiring external consultants or conducting specialized testing activities.

2.4. Reporting

Reports

- Test Scripts/Cases: Detailed documentation outlining the test scenarios, steps, and expected results for each function or feature being tested.
- Recommendations and Risk Strategy: A report summarizing any recommendations for improvements or enhancements to the system based on the testing results. Additionally, a risk strategy document highlighting potential risks identified during testing and proposed mitigation measures.

Records

- Test Input and Output Information: Records of the inputs provided to the system during testing, including test data, parameters, and configurations. This will also include the corresponding output generated by the system in response to the inputs.
- Test Results: Both detailed and summary reports documenting the outcomes of each test case, including any deviations from expected behavior, defects identified, and overall system performance metrics.

All reports and information generated during the Acceptance Testing process will be maintained as project records by the university<Business Unit>. These records will serve as valuable documentation for future reference, audits, and ongoing system maintenance.

2.5. Test Case Report

This report documents the test activities and results for each test case conducted during the testing phase. The following table outlines the necessary information for each test case:

Test Case ID	**Process Name/Code**	**Test Case Description**	**Data Input**	**Expected Result**	**Actual Results Description**	**Result Code**
TC001	Room Reservation	Reservation Creation	User A, Room X, Time Slot Y	Reservation successful	Reservation created successfully with confirmation message.	Accept
TC002	Room Reservation	Reservation Modification	User B, Reservation Z, Room W, Time Slot V	Reservation updated successfully	Reservation modified as expected, new confirmation received.	Accept
TC003	Room Reservation	Reservation Cancellation	User C, Reservation P	Reservation canceled successfully	Reservation removed from system, confirmation received.	Accept

This Test Case Report provides a structured overview of each test case conducted, including input data, expected and actual results, and any associated problem review numbers for tests requiring further review.

3. Testing Prerequisites

This section outlines requirements that must be satisfied and actions that must be implemented prior to the commencement of acceptance testing.

3.1. Quality Assurance

The Room reservation software will have been subject to the system developer's Quality Assurance process prior to delivery and subsequent Acceptance Testing by the Department. This includes :

- Formal technical reviews during software development
- Control of software documentation and the changes to it
- Design inspection ensuring testability and extensibility
- Coding convention and good manners.

3.2. Test Cases

Test Case Design:

Test cases will be designed to cover all aspects of the room reservation software, including positive and negative scenarios, boundary conditions, and error handling. Each test case will be documented with a unique identifier, description, input data, expected results, and pass/fail criteria.

Functional Test Cases:

Functional test cases will validate the core functionalities of the software, including:

- Creating a new reservation with valid and invalid input.
- Modifying an existing reservation with different user permissions.
- Checking room availability for various time slots and dates.
- Removing a reservation and verifying room availability afterward.

User Interface Test Cases:

User interface test cases will assess the usability and accessibility of the software's interface, covering aspects such as:

- Navigation between reservation screens.
- Input validation and error messages.
- Consistency of layout and design across different devices and screen sizes.

Performance Test Cases:

Performance test cases will evaluate the responsiveness and scalability of the software under various load conditions, including:

- Creating multiple concurrent reservations.
- Simulating peak usage periods to assess system response times.
- Monitoring resource utilization and server response times under load.

Security Test Cases:

Security test cases will identify and mitigate potential security vulnerabilities in the software, including:

- Input validation to prevent SQL injection and other forms of injection attacks.
- Authentication and authorization checks for user access control.
- Data encryption and protection of sensitive information.

Compatibility Test Cases:

Compatibility test cases will ensure that the software functions correctly across different browsers, devices, and operating systems, covering:

- Compatibility testing on popular web browsers (e.g., Chrome, Firefox, Safari).
- Testing on various devices (e.g., desktop, tablet, mobile) and screen resolutions.
- Verification of compatibility with different operating systems (e.g., Windows, macOS, iOS, Android).

Regression Test Cases:

Regression test cases will verify that new changes do not introduce defects or regressions in existing functionality. This includes:

- Re-testing of previously validated functionalities after new feature implementations or bug fixes.
- Verifying that fixes for reported defects do not introduce new issues.

Error Handling Test Cases:

Error handling test cases will assess the software's ability to detect, report, and recover from errors gracefully, including:

- Testing error messages and notifications for clarity and relevance.
- Simulating unexpected scenarios (e.g., server downtime, network errors) and verifying system behavior.

3.3. Collation of Test Data

Test Data Requirements:

Test data requirements will be identified and documented during the test planning phase. This includes:

- Sample user information for creating reservations.
- Test scenarios covering various room configurations and availability statuses.
- Data representing different time slots, dates, and durations for reservations.
- Existing reservations for testing modification and removal functionalities.

Data Collection and Preparation:

Test data will be collected and prepared according to the defined requirements. This involves:

- Generating or obtaining sample user data to simulate real-world usage scenarios.
- Preparing test cases with relevant data inputs to cover different aspects of the software's functionality.
- Ensuring data integrity and consistency throughout the testing process.

Data Storage and Management:

Test data will be securely stored and managed to facilitate efficient testing. This includes:

- Organizing test data in a structured format for easy access and retrieval.
- Implementing version control mechanisms to track changes to test data over time.
- Ensuring compliance with data privacy and security regulations to protect sensitive information.

Data Usage in Testing:

Test data will be used systematically during the execution of test cases. This involves:

- Incorporating test data into automated test scripts to simulate user interactions.
- Manually inputting test data as necessary to validate specific scenarios.
- Monitoring and logging data usage to identify any discrepancies or anomalies during testing.

Data Cleanup and Reset:

After testing is completed, test data will be cleaned up and reset to its original state. This includes:

- Removing test reservations and associated data from the system.
- Restoring any modified configurations or settings to their default values.
- Verifying that the system returns to a clean and consistent state for subsequent testing cycles.

Documentation:

Test data usage, preparation procedures, and cleanup steps will be documented for future reference. This includes:

- Recording any issues or challenges encountered during the handling of test data.

- Updating test data documentation as needed to reflect changes in the software or testing requirements.

4. Testing Procedure

The testing procedure involves executing a series of tests to ensure that the system functions according to the specified requirements and design. This includes various types of testing such as unit testing, system testing, and acceptance testing.

4.1. Test Schedule

This schedule outlines the specific dates for conducting different tests, including setting critical system dates and the transactions to be tested on each day. The testing team will be responsible for executing these tests according to the schedule.

4.2. Test Results

The process for documenting test results involves the following steps:

Test Execution: The testing team executes the planned tests according to the defined test cases and procedures.

Recording Results: After each test is executed, the results are recorded systematically. This includes documenting whether the test was performed, the outcome of the test (Acceptable, Complete failure, Partial failure), and any relevant comments or observations.

Test Results Scale: The test results are categorized using the scale provided in Table 7. Each test outcome is assigned a code (N for Not tested, A for Acceptable, F for Complete failure, P for Partial failure).

Collation and Review: The Test Manager is responsible for collating all test results and reviewing them to ensure accuracy and completeness.

Code	Description
N	Not tested
A	Acceptable
F	Complete failure
P	Partial failure (test completed)

This scale is used to categorize the outcomes of each test, providing a clear indication of the status of the system with respect to the testing process.

The <Business Unit> Test Manager will provide informal reports on the status of testing during Cycle 1, ensuring that stakeholders are informed about the progress and outcomes of the testing activities.

4.3. Review of Test Results

Upon completion of testing, a comprehensive analysis of test results will be conducted, identifying trends, patterns, and areas requiring attention. Defects will undergo triage and prioritization based on severity and impact, with root cause analysis informing targeted corrective actions. Detailed documentation of findings will be provided to stakeholders, facilitating transparency and informed decision-making. Lessons learned from the review process will be documented to drive continuous improvement in testing strategies and quality assurance practices, ensuring the software meets the desired standards before release.

4.4. Corrective Action

Upon identifying issues through testing and stakeholder feedback, a root cause analysis will be conducted to determine underlying factors, followed by prioritizing and promptly resolving issues based on severity and impact. Transparent communication will be maintained with stakeholders throughout the process, providing regular updates on resolution progress and potential impacts. After resolution, thorough validation and verification testing will ensure effectiveness, with documentation of actions taken and lessons learned to inform future improvements in software quality and processes.

4.5. Acceptance and Release

The acceptance criteria will be defined to align with stakeholder requirements, validated through User Acceptance Testing (UAT) involving key stakeholders to ensure the software meets their expectations. Release planning will determine readiness based on established criteria, facilitating coordination between teams for smooth deployment. Post-release support will address any issues promptly, with a feedback loop in place to gather insights for continuous improvement and future enhancements to the room reservation software.