Test Plan Document

December 01, 2023

Rel-Event: Real time Event Management System

11443	Anirudh B. Mitta
11446	Chitra Mahadevaiah
11451	Siddartha Reddy Legala
11474	Sai Rohith Burle

1. Introduction

1.1. Purpose

This test plan describes the testing approach and overall framework that will drive the testing of the web application Rel-Event: Real-time Event Management System. The document introduces:

Test Strategy: rules the test will be based on, including the givens of the project (e.g.: start / end dates, objectives, assumptions); description of the process to set up a valid test (e.g.: entry / exit criteria, creation of test cases, specific tasks to perform, scheduling, data strategy).

Execution Strategy: describes how the test will be performed and process to identify and report defects, and to fix and implement fixes.

Test Management: process to handle the logistics of the test and all the events that come up during execution (e.g.: communications, escalation procedures, risk and mitigation, team roster)

1.2. Scope

The scope of the test plan for the upcoming release of our Event Management System (EMS) encompasses the comprehensive testing of essential functions, including user authentication and authorization, event operations, ticketing system, community interaction, coordination tools, user profiles, mobile responsiveness, payment integration, and location services. The test plan will thoroughly examine the new features introduced in this release, considering potential risk areas, constraints, and dependencies. It will focus on differentiating the release type, whether it involves bug fixes, minor feature enhancements, or major feature additions.

The test plan will also outline specific testing goals, objectives, and the required testing information, particularly relevant in the context of launching a new event management site within the release. Continuous improvement efforts will be integrated by capturing lessons learned for future releases, fostering a culture of feedback and refinement throughout the testing process.

1.3. Planning (Roadmap)

The planning section of this test plan outlines the roadmap for executing testing activities throughout the development lifecycle. It encompasses a strategic approach to ensure the successful validation of the EMS release.

The testing process will be divided into distinct phases, each serving a specific purpose in maintaining product quality. Initial phases will focus on requirements analysis and test planning, defining the test environment, and creating test cases aligned with the identified scope.

Following this, the execution phase will commence, encompassing functional, non-functional, performance, security, and usability testing. The integration of automated testing tools will be leveraged where applicable to enhance efficiency and coverage.

The conclusion of testing will be marked by a comprehensive review of test results, and the generation of test reports highlighting the overall quality and readiness of the EMS release. This structured approach ensures a systematic and thorough evaluation of the software, aiming for a high-quality, reliable, and feature-rich Event Management System.

1.4. Testing Tools Used

In the testing phase of the EMS release, Selenium, a powerful automated testing tool, will be strategically employed to enhance efficiency in executing functional tests, specifically targeting repetitive scenarios and facilitating regression testing. Regular evaluations of the testing tools, such as Selenium, will be conducted throughout the testing lifecycle to ensure their continued effectiveness and alignment with evolving testing requirements. This iterative approach allows for adaptability, ensuring that the chosen tools consistently contribute to the optimization of testing processes and the thorough evaluation of the EMS release's overall quality and reliability.

1.5. Testing Environment

The configuration of the test environment for the EMS release is aligned with the AWS cloud deployment. Virtual servers, storage, and networking configurations are collectively managed to simulate the planned production setup. This collaborative approach ensures a comprehensive understanding of the test environment's intricacies among team members, fostering shared responsibility and knowledge transfer. Regular updates and maintenance of the test environment are conducted jointly to reflect changes in the AWS deployment. This coordinated effort contributes to a reliable and accurate evaluation of the EMS release before its deployment on the AWS cloud.

2. Unit Testing Plan

2.1 Modules Identified for Testing:

In the Unit Testing plan, specific modules have been identified for rigorous testing to ensure the integrity and functionality of individual components. The identified modules include:

1. User Authentication and Authorization Module:
   - Validate user registration, login, and password recovery processes.
   - Verify authentication mechanisms and role-based access controls.
2. Event Operations Module:
   - Test the creation, modification, and deletion of events.
   - Ensure accurate display of event details such as date, time, and location.
3. Ticketing System Module:
   - Verify the creation, purchase, and cancellation of tickets.
   - Validate ticket pricing, availability, and associated event information.
4. Community Interaction Module:
   - Test forums, discussions, and social interaction features.
   - Validate user-generated content moderation and notification functionalities.
5. Coordination Tools Module:
   - Validate planning and coordination tools among event organizers.
   - Test communication channels, task assignments, and updates.
6. User Profiles Module:
   - Test user profile creation, editing, and deletion.
   - Verify accurate display of user details and customization options.

7. Mobile Responsiveness Module:

- Ensure the system's usability and visual appeal on various mobile devices.
- Test responsive design for different screen sizes.

8. Payment Integration Module:

- Validate payment gateway integration for various modes.
- Verify successful and unsuccessful transaction scenarios.

9. Location Services Module:

- Test location-based features such as event mapping and venue information.
- Ensure accurate tracking of user location if applicable.

Each module will undergo comprehensive testing with a focus on unit-level functionality, data flow, and interaction with other modules. This targeted approach aims to identify and address any issues at the individual component level, ensuring the overall reliability and performance of the Event Management System.

2.2 Unit Test Plan

Module Name: User Authentication and Authorization

Sl. No	Test ID	Test Description	Input	Expected Output	Actual Output	Test Result (Pass/Fail)
1	UA01	Validate User Registration	User details	Successful registration notification		
2	UA02	Verify User Login	Valid credentials	Successful login and access		
3	UA03	Test Password Recovery	User email	Password reset confirmation		
4	UA04	Validate Role-Based Access Control	User roles	Access permissions enforced correctly		

Module Name: Event Operations

Sl. No	Test ID	Test Description	Input	Expected Output	Actual Output	Test Result (Pass/Fail)
1	EO01	Test Event Creation	Event details	Successful creation and notification		
2	EO02	Verify Event Modification	Updated details	Successful modification and notification		
3	EO03	Test Event Deletion	Event ID	Successful deletion and notification		
4	EO04	Validate Event Details Display	Event ID	Accurate display of date, time, and location		

Module Name: Ticketing System

Sl. No	Test ID	Test Description	Input	Expected Output	Actual Output	Test Result (Pass/Fail)
1	TS01	Verify Ticket Creation	Ticket details	Successful creation and confirmation		
2	TS02	Validate Ticket Purchase	Ticket ID	Successful purchase and receipt		
3	TS03	Ticket Cancellation	Ticket ID	Successful cancellation and notification		

3. Integration Test Plan

3.1 Methodology Used

The Integration Test Plan adopts the Top-down Integration methodology for comprehensive validation of the Event Management System (EMS). This approach initiates testing from higher-level modules and progressively integrates lower-level modules, ensuring a systematic examination of the system's functionality.

3.2 Approach

Top-down Integration Approach:

1. Start with Higher-Level Modules:

- Initiate testing with modules representing user-facing functionalities, such as User Authentication and Authorization, Event Operations, and Ticketing System.

2. Progressive Integration:

- Validate interactions and data flow between higher-level modules, ensuring seamless collaboration.

- Integrate progressively with lower-level modules, addressing dependencies as testing proceeds.

3. Early Detection of System-level Issues:

- Identify and address system-level integration issues early in the testing process, minimizing the risk of critical issues emerging later.

4. User Experience Validation:

- Prioritize the validation of functionalities that directly impact the user experience, aligning testing with user expectations.

Benefits:

- **Systematic Validation:** The top-down approach provides a systematic way to validate the collaboration and data flow between modules.

- **Early Issue Identification**: Early detection of system-level issues enhances the efficiency of issue resolution and reduces the likelihood of critical problems surfacing late in the testing process.

This Integration Test Plan, utilizing the Top-down Integration methodology, aims to ensure the seamless integration of modules within the EMS, contributing to a reliable and fully functional system.

4. System Testing / User Acceptance Test / Beta Testing

4.1 Testing Methodology

4.1.1. System Testing:

In the System Testing phase, a Black Box Testing methodology is employed to comprehensively validate the Event Management System (EMS). This approach involves testing the entire system's functionalities without internal code visibility. The testing scenarios encompass functional, performance, and security aspects to ensure the system's compliance with specified requirements.

4.1.2. User Acceptance Test (UAT):

User Acceptance Testing follows a White Box Testing methodology, actively involving end-users in the testing process. This phase ensures that the EMS meets business needs, is user-friendly, and aligns with organizational objectives. Test scenarios include business process validation, usability testing, and verification of acceptance criteria defined in collaboration with stakeholders.

4.1.3. Beta Testing:

Beta Testing adopts a Field-Testing approach, releasing a pre-release version of the system to a select group of external users. This phase gathers real-world feedback and uncovers potential issues before the official release. Beta Testing scenarios involve real-world usage validation, feedback collection, and compatibility testing across various devices and environments.

4.2 Description

4.2.1. System Testing:

The objective of System Testing is to validate the overall functionality, performance, and security of the EMS before deployment. The execution involves comprehensive functional testing, load testing for performance evaluation, and security testing to identify and mitigate potential vulnerabilities. The deliverables include detailed test reports highlighting functional, performance, and security testing results, along with bug reports and resolutions.

4.2.2. User Acceptance Test (UAT):

UAT aims to ensure that the EMS aligns with user expectations, business processes, and usability standards. The execution involves user-driven testing scenarios, usability assessments, and verification of acceptance criteria defined in collaboration with stakeholders. The deliverables include UAT test results showcasing user satisfaction, system alignment, and documentation of identified issues and resolutions.

4.2.3. Beta Testing:

The objective of Beta Testing is to obtain real-world feedback from external users to improve system robustness and user satisfaction. The execution involves releasing beta versions to a limited user group for diverse testing scenarios, collecting user feedback through surveys and forums, and analyzing usage patterns to identify potential improvements. The deliverables include beta testing reports highlighting feedback, identified issues, and suggested enhancements, along with recommendations for system improvements based on user input. This comprehensive testing strategy ensures that the EMS undergoes thorough evaluation, aligning with user expectations and industry standards before its official release.