TEST PLAN FOR 

Prepared By – Dinakar Reddy

Table of Contents

1. Overview/Context
2. Scope
3. Test Strategy

3.1 Objectives

3.2 Test Levels

3.3 Test Types

3.4 Test Techniques

3.5 Test Deliverables

1. Test Environments
2. Test Schedule
3. Entry and Exit Criteria
4. Defect Reporting and Management

7.1 Defect Criteria

7.2 Reporting Steps

7.3 Triaging & Prioritization

7.4 Tools

7.5 Roles & Responsibilities

7.6 Communication

7.7 Metrics

1. Tools
2. Risks and Mitigations
3. Approvals
4. Test Deliverables

### 1. Overview/Context

This test plan defines the strategy, scope, and approach for testing the Event Booking System. The goal is to ensure that all functional and non-functional requirements, including performance, security, and usability, are met. This plan serves as a guideline for all stakeholders involved in the testing process, ensuring a high-quality system before deployment.

**2. Scope**

This test plan applies to the following modules of the Towner Event Booking System:

* **Welcome Page:** Navigation, Event Browsing, UI/UX
* **User Registration:** Email/Mobile Registration, OTP Verification, Password Security
* **Login Module:** Email/Password, OTP Login, Session Management
* **Ticket Booking:** Single & Group Booking, Payment, Ticket Resale
* **E-Ticket Delivery:** Email/SMS Delivery, Ticket Resend, Event Reminders
* **Booking History:** Past/Upcoming Bookings, Cancellations, Refund Status

**Testing will cover:**

* Functional Testing
* UI/UX Testing
* Security Testing
* Performance Testing
* Integration Testing

**Out of Scope:** Load testing beyond expected concurrent users, testing on unsupported browsers/devices.

### 3. Test Strategy

#### ****3.1 Objectives****

* Ensure all system functionalities meet business and user requirements.
* Identify and resolve defects before the system is released.
* Validate system performance under various conditions.
* Ensure security measures protect user data and transactions.

**3.2 Test Levels**

|  |  |  |
| --- | --- | --- |
| **Test Level** | **Responsibility** | **Purpose** |
| Unit Testing | Developers | Validate individual components |
| Integration Testing | Developers & Testers | Check interaction between modules |
| System Testing | Testers | End-to-end system validation |
| User Acceptance Testing (UAT) | End Users | Validate business requirements |

#### ****3.3 Test Types****

|  |  |  |
| --- | --- | --- |
| **Functional Testing** |  |  |
| **Testing Type** | **Performed By** | **When to Use** |
| Unit Testing | Developers | During development |
| Smoke Testing | QA Team | After a new build is deployed |
| Functionality Testing | QA Team | After Smoke Test passes |
| Integration Testing | Developers & Testers | After Unit Testing |
| Regression Testing | Testers | After new features/bug fixes |
| UAT | End Users | Before deployment |
|  |  |  |
| **Non-Functional Testing** |  |  |
| **Testing Type** | **Performed By** | **When to Use** |
| Performance Testing | Performance Testers | Before release |
| Load Testing | Performance Testers | Before release |
| Security Testing | Security Testers | Before deployment |
| Usability Testing | UI/UX Testers | During development or before release |
| Accessibility Testing | Testers | Before release |

**3.4 Test Techniques**

|  |  |  |
| --- | --- | --- |
| **Technique** | **Who Does It?** | **When to Use?** |
| Black-box Testing | Testers | Verify system behavior without internal code knowledge |
| White-box Testing | Developers | Analyze internal logic and structure |
| Grey-box Testing | Testers & Developers | Ensure security & integration work correctly |

**3.5 Test Deliverables**

* Test Plan
* Test Cases
* Test Execution Reports
* Defect Reports
* Test Summary Reports

### 4. Test Environments

* **Operating Systems:** Windows 10, macOS, Android, iOS
* **Browsers:** Chrome, Firefox, Edge, Safari
* **Network Conditions:** 3G/4G/5G, Wi-Fi, Wired Connections
* **Security Testing:** Authentication, Encryption, Data Protection

### 5. Test Schedule

|  |  |  |  |
| --- | --- | --- | --- |
| **Task** | **Start Date** | **End Date** | **Responsible Team** |
| Test Plan Creation | YYYY-MM-DD | YYYY-MM-DD | QA Team |
| Test Case Design | YYYY-MM-DD | YYYY-MM-DD | QA Team |
| Test Execution | YYYY-MM-DD | YYYY-MM-DD | QA Team |
| Defect Reporting & Fixes | YYYY-MM-DD | YYYY-MM-DD | QA & Dev Team |
| Test Summary Report | YYYY-MM-DD | YYYY-MM-DD | QA Team |

### 6. Entry and Exit Criteria

**Entry Criteria:**

* Requirements documents are finalized.
* Test cases are reviewed and approved.
* Test environment is stable.
* Code is deployed for testing.

**Exit Criteria:**

* All test cases are executed successfully.
* Critical defects are fixed and retested.
* UAT is completed and approved.

### 7. Defect Reporting and Management

#### ****7.1 Defect Criteria****

Any deviation from expected system behavior, including functional, security, or performance issues.

#### ****7.2 Reporting Steps****

1. Identify the defect and document steps to reproduce.
2. Attach screenshots/log files if applicable.
3. Assign severity (Critical, High, Medium, Low).

#### ****7.3 Triaging & Prioritization****

* **Critical:** Blocks key functionality, must be fixed immediately.
* **High:** Major functionality affected, needs urgent fix.
* **Medium:** Minor issue, fix before release.
* **Low:** Cosmetic/UI issue, can be resolved later.

#### ****7.4 Tools****

* GitHub (Defect Tracking)
* Selenium (UI Automation)
* Postman (API Testing)
* JMeter (Performance Testing)

#### ****7.5 Roles & Responsibilities****

* **QA Team:** Execute test cases and report defects.
* **Development Team:** Fix and retest defects.
* **Project Manager:** Approve fixes and test completion.

#### ****7.6 Communication****

* Daily stand-up meetings (Teams/Slack)
* Defect status updates in GitHub

#### ****7.7 Metrics****

* **Defect Density:** Total defects / Total test cases
* **Resolution Time:** Time taken to fix critical defects
* **Test Coverage:** (Executed Test Cases / Total Test Cases) \* 100

**8. Tools**

|  |  |
| --- | --- |
| **Tool** | **Purpose** |
| Selenium | UI Automation Testing |
| Postman | API Testing |
| JMeter | Performance Testing |
| GitHub | Defect Management |
| TestRail | Test Case Management |

**9. Risks and Mitigations**

|  |  |
| --- | --- |
| **Risk** | **Mitigation** |
| Tight Deadlines | Prioritize test cases, optimize resources |
| Test Environment Instability | Backup test environment & proactive monitoring |
| Frequent Requirement Changes | Agile testing approach & continuous updates |

### 10. Approvals

**This test plan requires approval from:**

* Project Manager
* QA Lead
* Development Lead

### 11. Test Deliverables

* Test Plan
* Test Cases
* Defect Reports
* Test Summary Reports