**Test Plan for Parking Management System (PMS).**  
**Version**: 1.0  
**Date**: 20.10.2024  
**Prepared By**: Nurul Islam  
**Approved By**: Muntasir Abdullah Mizan.

**1. Introduction**

**1.1 Purpose**

The purpose of this Test Plan is to outline the testing strategy for the Parking Management System. This document details the scope, objectives, resources, schedule, and testing methods to ensure that the system meets its requirements and functions as expected.

**1.2 Scope**

This Test Plan covers functional, non-functional, and regression testing of the **Parking Management System**. It includes testing of all user roles (Admin, User) and functionalities (Parking category, Rate Calculation (Hourly, Daily & Weekly Rate), Lost Ticket fee and Billing Information).

**1.3 Definitions, Acronyms, and Abbreviations**

* **Valet Parking**: A service where a valet parks the customer's car for a daily or hourly rate.
* **Short-Term Parking**: Hourly parking for short durations.
* **Long-Term Parking**: Parking for extended periods, available in both garage and surface lots.
* **Long-Term Surface Parking**: less accessibility to direct building entry but ample space for larger vehicles.
* **Economy Lot Parking**: Affordable parking options with lower daily rates.
* **Lost Ticket Fee**: A fee charged when a parking stub is lost.

**2. Test Objectives**

* Verify the accuracy of fee calculations based on parking type and time.
* Ensure the system responds within acceptable performance standards.
* Validate user interfaces for accessibility, usability, and security.
* Confirm that the system processes payments and handles lost ticket scenarios correctly.
* Confirm that performance, security, and compliance requirements are met.
* Confirm that the system can handle expected user loads.
* Identify and resolve defects before deployment.

**3. Test Scope**

**3.1 In-Scope**

* User authentication (registration, login)
* Functional testing: Parking category’s, rate calculations and Lost ticket.
* Data validation: Parking availability, booking date/time, and billing information.
* External interface testing: hardware and software compatibility
* Payment integration
* User interfaces (web and mobile)
* Performance and load testing
* Security testing

**3.2 Out-of-Scope**

* Third-party services outside the direct control of the system (e.g., payment gateway failures).
* Non-functional requirements beyond those specified.

**4. Test Approach**

**4.1 Testing Levels**

* **Unit Testing**: Conducted by developers to verify individual components.
* **Integration Testing**: Ensures that different modules work together.
* **System Testing**: Validates the complete system against the requirements.
* **User Acceptance Testing (UAT)**: Performed by end users to ensure the system meets their needs.
* **Performance Testing**: Ensure the application meets performance requirements like response time and uptime.
* **Security Testing**: Verify that sensitive information is secure and the system complies with data protection regulations.

**4.2 Types of Testing**

* **Functional Testing**: Validate all functional requirements.
* **Non-Functional Testing**: Includes performance, security, usability, and compatibility testing.
* **Regression Testing**: Verify that new changes do not adversely affect existing functionalities.

**4.3 Test Items**

The following modules will be tested:

* **Parking Booking Module**: Includes parking category’s selection, date/time selection, and parking confirmation.
* **Rate Calculation Module**: Handles hourly, daily, and weekly rate calculations.
* **Lost Ticket Charge Module**: Applies lost ticket charge we’ll verify that the system accurately applies a $10.00 lost ticket fee when a ticket is reported as lost
* **User and Admin Interfaces**: Verify the functionality and usability of both interfaces.
* **Data Validation**: Ensure real-time availability updates, parking date validation, and secure billing processing.

**5. Test Resources**

**5.1 Team Roles**

* **Test Manager**: Muntasir Abdullah Mizan.
* **Tester**: Nurul Islam

**5.2 Tools**

* **Test Management Tool**: [e.g., JIRA, TestRail]
* **Automation Tools**: [e.g., Selenium, Postman]
* **Performance Testing Tools**: [e.g., JMeter, LoadRunner]
* **Security Testing Tools**: [e.g., OWASP ZAP, Burp Suite]

**6. Test Schedule**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sl** | **Activity** | **Objective** | **Start Date** | **End Date** | **Time Frame** | **Remarks** |
| **1.** | **Preparation Phase** | Test Planning, Test Case Development. | **------------** | **------------** | 1-6 days |  |
| **2.** | **Functional Testing** | Room Booking, Rate Calculation, Overtime Charges, Cancellation Policy | **---------------** | **------------** | 7-12 days |  |
| **3.** | **Data Validation Testing** | Room Availability, Booking Date and Time, Billing Information | **--------------** | **------------** | 13-15 |  |
| **4.** | **Non-Functional Testing** | Performance Testing, Security Testing, Compliance Testing | **---------------** | **----------** | 16-17 |  |
| **5.** | **Review & Reporting** | Test Report Compilation, Bug Fixing and Re-testing, | **---------------** | **------------** | 18-20 |  |
| **6.** | **Sign-off & Deployment Preparation** | Final Review | **---------------** | **-----------** | 21-23 |  |

**7. Test Cases**

**7.1 Functional Test Cases**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Requirement ID** | **Description** | **Test Case ID** | **Test Case Description** | **Expected Result** |
| FR-1, FR-2 | Valet Parking Fee Calculation | TC-001 | Verify valet rate for full day and less than 5 hours | System calculates correct rate: $18 for a day, $12 for <= 5 hours |
| FR-3 to FR-6 | Short-Term Parking Fee Calculation | TC-002 | Test hourly and half-hourly charges with daily max | Correct charges applied: $2 first hour, $1 each additional half-hour, max $24/day |
| FR-7 to FR-9 | Long-Term Garage Fee Calculation | TC-003 | Verify hourly, daily, and weekly max with 7th day free | System applies appropriate rates with 7th day free for weekly stays |
| FR-10 to FR-12 | Long-Term Surface Parking Fee Calculation | TC-004 | Test surface lot rates with daily and weekly max | $2/hour, max $10/day, max $60/week |
| FR-13 to FR-15 | Economy Lot Parking Fee Calculation | TC-005 | Validate economy rates with daily and weekly max | Rates applied as $2/hour, max $9/day, max $54/week |
| FR-16 | Lost Ticket Fee | TC-006 | Confirm lost ticket handling with $10 fee | System applies $10 lost ticket fee |

**7.2 Non-Functional Test Cases**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Non-Functional Req** | **Description** | **Test Case ID** | **Test Case Description** | **Expected Result** |
| Performance | Display fee calculation within 2 seconds | TC-007 | Measure system response for fee calculation | Response time ≤ 2 seconds |
| Reliability | Ensure 99.9% uptime | TC-008 | Simulate load to check stability | System remains stable without failures |
| Usability | User-friendly interface with clear labels | TC-009 | Test labels and instructions visibility | Users can navigate and understand instructions easily |
| Security | SSL/TLS encryption & access control | TC-010 | Attempt unauthorized access & verify encryption | Unauthorized access blocked, SSL encryption active |
| Maintainability | Update rates without downtime | TC-011 | Change parking rates and assess availability | Rates updated with no downtime impact |

**7.3 Interface Testing**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Interface Type | Test Case ID | Test Description | Expected Result | Interface Type |
| User Interface | TC-012 | Verify UI for attendants and customers | Intuitive interface for ticket management and payment | User Interface |
| Hardware Interface | TC-013 | Validate connectivity with ticket printers/card readers | Seamless interaction and accurate ticket issuance | Hardware Interface |

**7. Test Deliverables**

* Test Plan Document
* Test Cases Document
* Test Scripts (for automation)
* Test Execution Reports
* Defect Reports
* UAT Feedback and Summary
* Test Closure Report

**8. Roles and Responsibilities**

* **Test Manager**: Oversee the testing process, manage resources, and communicate with stakeholders.
* **Test Engineers**: Develop test cases, execute tests, and report defects.
* **Developers**: Assist in defect resolution and provide technical support during testing.
* **UAT Participants**: End-users who will validate the system’s functionality and usability.

**9. Test Environment**

* **Hardware**:
* Servers: Cloud-based hosting with at least 8GB RAM, 4 vCPUs.
* Client machines: Desktop and mobile devices with various operating systems (Windows, macOS, iOS, Android).
* **Software**:
* Web browsers: Chrome, Firefox, Safari, Edge.
* Mobile devices: Latest versions of Android and iOS.
* **Network**: [Specify network configurations, if any]

**10. Risk Management**

* **Risk**: Delay in test case development due to incomplete requirements.
  + **Mitigation**: Engage business analysts for clarity on requirements early.
* **Risk**: Limited availability of test resources.
  + **Mitigation**: Cross-train team members on critical tasks.

**11. Approval**

|  |  |  |  |
| --- | --- | --- | --- |
| **Role** | **Name** | **Signature** | **Date** |
| Test Manager | [Name] | [Signature] | [Date] |
| Project Manager | [Name] | [Signature] | [Date] |
| QA Lead | [Name] | [Signature] | [Date] |