# **Test Plan Document**

# 1. Introduction

## 1.1 Purpose

The purpose of this test plan is to outline the testing strategy, scope, objectives, and approach for verifying the functionality, performance, and usability of the PHPTRAVELS website. This document ensures that all critical features, including flight bookings, hotel reservations, tour packages, car rentals, and user management, meet the expected quality standards before deployment.

## 1.2 Scope of Testing

### Features to be Tested

The following features will be covered in testing:

### 1. User Authentication (Login, Signup, Profile Management)

* User registration & login with valid/invalid credentials
* Password reset functionality
* Role-based access control (Customer, Admin, Agent)
* Session management & security
* **Manual Testing:** UI validation, form inputs, authentication scenarios
* **Automation Testing:** Login/logout, session management, role-based access
* **API Testing:** Authentication API, session handling, security checks
* **Functional Testing:**
  + User registration & login with valid/invalid credentials.
  + Password reset functionality.
  + Role-based access control (Customer, Admin, Agent).
  + Session management & security.
* **Non-Functional Testing:**
  + Performance under concurrent login attempts.
  + Usability testing for ease of use.
  + Compatibility across different devices and browsers.

### 2. Hotel Booking System

* Search functionality (location, price, rating filters)
* Hotel details page (room availability, images, pricing)
* Booking process (room selection, guest details, payment validation)
* Cancellation & modification policies
* **Manual Testing:** Search functionality, UI validation, room selection
* **Automation Testing:** End-to-end booking flow, price validation, confirmation process
* **API Testing:** Hotel availability, booking API, cancellation API
* **Functional Testing:**
  + Search hotels by location, price, rating.
  + Hotel details page with availability, images, pricing.
  + Booking process: room selection, guest details, payment validation.
  + Cancellation & modification policies.
* **Non-Functional Testing:**
  + Load testing for search queries.
  + Usability testing for booking workflow.
  + Cross-browser and device compatibility.

### 3. Flight Booking System

* Flight search with filters (date, price, stops, airline)
* Booking process (seat selection, passenger details, payment integration)
* E-ticket generation & notifications
* **Manual Testing:** Flight search, seat selection, ticket confirmation
* **Automation Testing:** End-to-end flight booking, validation of filters, price updates
* **API Testing:** Fetching flights, booking flights, retrieving ticket details
* **Functional Testing:**
  + Flight search with filters (date, price, stops, airline).
  + Booking process (seat selection, passenger details, payment integration).
  + E-ticket generation & notifications.
* **Non-Functional Testing:**
  + Performance testing for search response time.
  + UI testing for seat selection experience.
  + Mobile responsiveness.

### 4. Car Rentals

* Search & compare available cars
* Booking process & payment validation
* Rental duration & cancellation policies
* **Manual Testing:** Search & compare available cars, UI validation
* **Automation Testing:** Rental booking flow, payment validation, order confirmation
* **API Testing:** Car availability, booking API, pricing updates
* **Functional Testing:**
  + Search & compare available cars.
  + Booking process & payment validation.
  + Rental duration & cancellation policies.
* **Non-Functional Testing:**
  + Load testing for search feature.
  + User experience validation.
  + Compatibility across platforms.

### 5. Tours & Activities

* Tour listings & search filters
* Booking confirmation process
* Cancellation & refund handling
* **Manual Testing:** Tour search, filters, UI validation
* **Automation Testing:** Booking confirmation, refund processing, user journey validation
* **API Testing:** Fetching tours, booking a tour, managing cancellations
* **Functional Testing:**
  + Tour listings & search filters.
  + Booking confirmation process.
  + Cancellation & refund handling.
* **Non-Functional Testing:**
  + Performance of search functionality.
  + Usability of booking confirmation process.
  + Cross-device compatibility.

### 6. Payment System

* Different payment methods (Credit Card, PayPal, Bank Transfer)
* Payment gateway validation & error handling
* **Manual Testing:** UI validation, form handling, transaction checks
* **Automation Testing:** Payment gateway validation, error handling, refund process
* **API Testing:** Payment processing, transaction history, validation of error handling

### 7. API Testing

* **Hotels API:** Fetch hotel details & availability
* **Flights API:** Retrieve flight data
* **Booking API:** Create, modify, cancel bookings
* **Manual Testing:** Basic API request testing via Postman
* **Automation Testing:** Automated API validation & response verification
* **API Testing:** Validate endpoints for Hotels, Flights, Bookings, Payments
* **Functional Testing:**
  + Hotels API: Fetch hotel details & availability.
  + Flights API: Retrieve flight data.
  + Booking API: Create, modify, cancel bookings.
* **Non-Functional Testing:**
  + Response time testing.
  + Load testing for API calls.
  + Data integrity checks.

### 8. Admin Panel (Back-office Testing)

* User & Role Management
* Inventory & Pricing Updates
* Reporting & Analytics
* **Manual Testing:** User & Role Management, UI validation
* **Automation Testing:** Inventory & Pricing Updates, Reporting & Analytics workflows
* **API Testing:** Admin API functionalities, database interactions
* **Functional Testing:**
  + User & Role Management.
  + Inventory & Pricing Updates.
  + Reporting & Analytics.
* **Non-Functional Testing:**
  + Usability testing for admin workflows.
  + Performance testing for data-intensive operations.
  + Compatibility with multiple screen resolutions.

## 1.3 Testing Objectives

The main objectives of this test plan are:

* To verify that all website features function as intended.
* To identify and resolve defects before release.
* To ensure compatibility across different browsers and devices.
* To validate the accuracy of search results and payment processing.
* To enhance user experience by identifying usability issues.

## 1.4 Assumptions & Constraints

* The website should work across major browsers (Chrome, Firefox, Edge, Safari).
* Testing will be performed on both desktop and mobile devices.
* Security testing is out of scope for this test plan.

## 1.5 Test Team & Responsibilities

The following team members will be involved in the testing process:

|  |  |  |
| --- | --- | --- |
| Feature | Testing Types | Assigned Member |
| User Authentication (Login, Signup, Profile Management) | Manual, API, Automation | Esraa |
| Hotel Booking System | Manual, API, Automation | Haneen |
| Flight Booking System | Manual, API, Automation | Mariam |
| Car Rentals | Manual, API, Automation | Mohamed |
| Tours & Activities | Manual, API, Automation | Esraa |
| Payment System (Checkout & Transactions) | Manual, API, Automation | Mariam |
| Admin Panel – User & Pricing Management | Manual, API, Automation | Haneen |
| Admin Panel – Reports & Analytics | Manual, API, Automation | Mohamed |
| Performance Testing (Load Testing for Booking & Search) | Automation | Haneen |
| API Testing (Hotels, Flights, Bookings, Payments) | API Testing | All Members |
| Loyalty Program (Rewards, Discounts, Points System) | Manual, API, Automation | Esraa |
| Multi-Currency Support (Exchange Rate Updates, Pricing Adjustments) | Manual, API, Automation | Mohamed |
| Customer Support Chat (Live Chat, Chatbot, Ticket System) | Manual, API, Automation | Mariam |

## **1.6. Out of Scope**

* 3rd Party Integrations (unless explicitly required)
* UI/UX Design Consistency (unless requested separately)
* Data Migration & Back-end Database Queries
* Security Testing (to be handled separately)

## **2. Pages to Test**

### **General Pages**

* **Home Page**
  + Display top destinations, deals, and featured hotels
  + Search bar for flights, hotels, and cars
* **Login & Registration**
  + User authentication system
* **Dashboard**
  + Customer dashboard for managing bookings
  + Admin dashboard for monitoring sales and activity
* **Contact & Support**
  + Contact form and customer support chat

### **Booking Pages**

* **Hotels Page**
* **Flights Page**
* **Car Rentals Page**
* **Tours Page**
* **Payment Page**

## **3. Testing Risks & Challenges**

1. **Browser & Device Compatibility**
   1. The website may behave differently on Chrome, Firefox, Safari, and Edge.
   2. Mobile responsiveness issues on Android and iOS devices.
2. **Third-Party API Failures**
   1. Payment gateways (PayPal, Credit Card) may have downtime or unexpected errors.
   2. Hotel and flight booking APIs may return incorrect data or fail under load.
3. **Security Vulnerabilities**
   1. Weak authentication could allow unauthorized access.
   2. Data leaks if personal or payment information isn’t properly encrypted.
4. **Performance & Load Handling**
   1. High traffic on search and booking pages could slow down the system.
   2. API rate limits might cause failures when multiple users book at once.
5. **Automation Test Flakiness**
   1. Dynamic UI elements may cause tests to fail randomly.
   2. API responses may change due to server delays, affecting automated scripts.
6. **Data Consistency Issues**
   1. Booking records might not sync properly between front-end and back-end.
   2. Inventory updates (hotel availability, flight pricing) could be delayed.
7. **Edge Cases in User Actions**
   1. Users may enter unexpected inputs (invalid email, incorrect dates).
   2. Booking modifications and cancellations might cause conflicts.
8. **Time Zone & Localization Issues**
   1. Date and time differences could affect booking confirmations.
   2. Currency conversions may not be accurate across different regions.
9. **Regression Testing Overhead**
   1. Frequent updates to the system may break existing functionality.
   2. Maintaining automation scripts for UI changes requires constant updates

## **KPIs:**

## 1-Bug Detection Rate

**Introduction:**

This report outlines the Bug Detection Rate (BDR) for the PHPTravels website. The testing process focused on key functionalities such as user authentication, search and booking, payment processing, and API responses. The report also includes the identified bugs, their severity, and recommendations for improving the system's reliability.

**Testing Scope**

The following areas were tested:

* **User Authentication**: Login, Registration, Password Reset
* **Search & Booking Functionality**: Hotel, Flight, and Tour searches
* **Payment Processing**: Checkout, Invoice Generation
* **API Endpoints**: Fetching booking details, Payment handling
* **Responsive Design & Browser Compatibility**

**Test Execution & Bug Tracking**

**Test Cases and Results**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  | **Test Case** | **Expected Result** | **Actual Result** | **Status** |  |  |  |  |
|  |  |  |  |  |  |  |  | Login with valid credentials | Redirect to dashboard | Redirects correctly | Passed |  |  |  |  |
|  |  |  |  |  |  |  |  | Search for a hotel in New York | List of hotels should appear | Page loads slowly | Failed |  |  |  |  |
|  |  |  |  |  |  |  |  | Payment using Visa card | Payment should be successful | Transaction fails with an error | Failed |  |  |  |  |
|  |  |  |  |  |  |  |  | API request to fetch bookings | API should return a valid JSON response | Returns HTTP 500 error | Failed |  |  |  |  |

**Bug Detection Rate Calculation**

**Formula:**

**Bug Detection Rate=** ( number of Bugs Found)/(Total Test Cases Executed) ×100

* **Total Bugs Found:** 25

**25/200×100=12.5%**

**Performance Analysis**

**High BDR (> 10%)**: Indicates significant issues in the system that need improvement. **Common issues found**:

* Slow page loading times
* Payment processing failures
* API response errors (HTTP 500)

**Recommendations & Improvements**

* **Enhance testing strategies** to improve bug detection efficiency.
* **Use bug tracking tools** like JIRA or Bugzilla for better issue management.
* **Expand test coverage** by including performance, security, and automation testing.
* **Optimize payment processing** to ensure a seamless user experience.

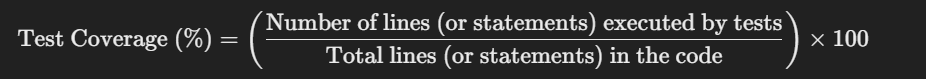
**Conclusion**

A total of **200 test cases** were executed, and **25 bugs** were found, resulting in a **12.5% Bug Detection Rate**. Based on the analysis, improvements in site performance, API reliability, and payment processing are recommended to enhance user experience and system stability

## 2-Test coverage percentage

**Test coverage percentage is a metric that measures how much of your code is executed during testing. It helps assess the effectiveness of your tests and identify untested parts of the code.**

How is Test Coverage Calculated



**for example :**

estimated test coverage **for** car search, availability, booking process, rental policies

|  |  |  |
| --- | --- | --- |
| **Test Case** | **Estimated Coverage** | **Remarks** |
| **Car Search** | 85% | Covers different locations, dates, and car types but may lack edge cases like invalid locations. |
| **Car Availability Check** | 80% | Covers basic availability but may miss tests for high-demand periods or rare car types. |
| **Booking Process** | 90% | Covers most scenarios but may miss edge cases like expired credit cards or incorrect user details. |
| **Rental Policies Validation** | 75% | Covers major rules but may not test all edge cases like special license types or country-based restrictions. |

**Overall Estimated Test Coverage for this test case :** **83%**

3-Response time for API requests

API response time is the time taken by the server to process a request and return a response to the client (browser or application). It is measured in milliseconds (ms).

**Tools Used for Measurement:**

Postman: For sending requests and measuring response time.

**Performance Benchmarks**

|  |  |
| --- | --- |
| **API TYPE** | **TARGET RESPONSE TIME** |
| Authentication | < 400 ms |
| Data Retrieval | < 300 ms |
| Complex Operations (Checkout) | < 2 seconds |

**Results and Recommendations**

Login API : average response time =310 ms.

Register API : average response time =307 ms.

Reset Password API : average response time =354 ms.

Reload (I'am not a robot) API : average response time =279 ms.

Hotel Search API : average response time =320 ms.

Tours Search API : average response time =415 ms.

Cars Search API : average response time =371 ms.

Booking API : average response time =438 ms.

Create Payment Resource API: average response time =213ms

Payment API : average response time = 1 sec.

**API Response Time (Average):**

Total = 310 + 307 + 354 + 279 + 320 + 415 + 371 + 438 + 213 + 1000 = 4007 ms

Average = 4007 ms ÷ 10 = 400.7 ms.