# **Test Plan for Hotel Bookings Related Services**

**Created by:** Poonam Prasad

## **1. Objective**

This document outlines the test plan for the **Hotel Bookings related services** application. The objective is to ensure that all REST API features and functionalities work as expected for the target audience: **QA Engineers, API Consumers, Developers, and Stakeholders**.

## **2. Scope**

The scope of this test plan includes:

* **Features to be tested:**
  + **Authentication:** POST /auth
  + **Create Booking:** POST /booking
  + **Retrieve Booking(s):** GET /booking, GET /booking/{id}
  + **Update Booking:** PUT /booking/{id}
  + **Partial Update:** PATCH /booking/{id}
  + **Delete Booking:** DELETE /booking/{id}
  + **Health Check:** GET /ping
* **Types of Testing:**
  + Manual Testing
  + Automated Testing (via Postman/Newman)
  + Performance Testing (via JMeter)
  + Accessibility Testing (Swagger UI for developers and testers)
* **Environments:**
  + Browsers: Google Chrome, Mozilla Firefox, Microsoft Edge
  + OS: Windows 10, macOS, Ubuntu
  + Devices: Laptops, desktops
  + Tools: Postman, JMeter, Curl, Swagger UI
* **Evaluation Criteria:**
  + Number of defects found
  + Severity and priority of bugs
  + Test case pass rate
  + Execution timelines
  + Stakeholder satisfaction
* **Team Roles and Responsibilities:**
  + **QA Lead (1):** Test strategy, planning, reporting
  + **Test Engineers (4):** Test case design, execution, defect logging

## **3. Inclusions**

* **Introduction:** This test plan validates the complete functionality of the RESTful API used for hotel bookings.
* **Test Objectives:**
  + Validate CRUD operations for bookings
  + Confirm authentication with tokens
  + Measure API performance and uptime
  + Ensure accurate error handling and validation

## **4. Exclusions**

* Integration with third-party hotel management systems
* Notifications via email/SMS
* UI tests beyond the Swagger API interface

## **5. Test Environments**

* **Operating Systems:** Windows 10, macOS Ventura, Ubuntu 22.04
* **Browsers:** Chrome v124+, Firefox v115+, Edge v122+
* **Devices:** Laptop/Desktop
* **Network Connectivity:** Wired LAN, Wi-Fi
* **Hardware/Software Requirements:** Min. 4GB RAM, 2GHz processor, Postman 10+, JMeter 5.6
* **Security Protocols:** Token-based authentication via /auth
* **Access Permissions:**
  + Test Engineers: Full API access
  + QA Lead: Management and approvals
  + Developers: Bug resolution and log access
  + Stakeholders: Reporting dashboard access

## **6. Defect Reporting Procedure**

* **Defect Identification Criteria:**
  + HTTP response code issues
  + Functional discrepancies
  + Unauthorized access scenarios
  + Security misconfigurations
* **Steps for Reporting Defects:**
  + Log into JIRA
  + Use template for API bug reports
  + Add request/response samples, headers, and logs
  + Assign priority and developer
* **Triage and Prioritization:**
  + Critical (P1): System failure or data loss
  + Major (P2): Incorrect functionality
  + Minor (P3): Cosmetic or edge cases
* **Tracking Tool:** JIRA
* **Communication Channels:** Slack, Daily Stand-ups
* **Metrics Tracked:**
  + Total defects
  + Open vs resolved bugs
  + Time to resolve
  + Defect density per feature

## **7. Test Strategy**

### **Step 1: Test Scenario and Test Case Creation**

* **Techniques Used:**
  + Equivalence Partitioning
  + Boundary Value Analysis
  + Decision Table Testing
  + Use Case Testing
  + State Transition Testing
  + Exploratory Testing
  + Error Guessing

### **Step 2: Testing Procedure**

* **Smoke Testing:** Ping, Auth, and Booking Creation
* **In-depth Testing:**
  + Validate all request payloads and headers
  + Functional response for each CRUD operation
* **Cross-environment testing:** Parallel execution in different OS/browser environments
* **Bug Logging:** Tracked daily; updates sent in stand-ups
* **Testing Types:**
  + Smoke
  + Sanity
  + Regression
  + Retesting
  + Usability
  + Functional and Security

### **Step 3: Best Practices**

* Shift Left Approach: Involve QA from sprint planning
* Context-Driven Testing
* End-to-End Workflow Testing
* Maintain Postman collection for regression suite

## **8. Test Schedule**

| **Task** | **Assigned To** | **Duration** | **Dates** |
| --- | --- | --- | --- |
| Test Plan Preparation | QA Lead | 2 days | May 1 – May 2 |
| Test Scenario & Case Creation | All QAs | 3 days | May 3 – May 5 |
| Test Execution – Manual | All QAs | 4 days | May 6 – May 9 |
| Automation with Postman/Newman | 2 QAs | 2 days | May 6 – May 7 |
| Performance Testing (JMeter) | 1 QA | 2 days | May 8 – May 9 |
| Defect Retesting & Closure | All QAs | 2 days | May 10 – May 11 |
| Test Summary Report | QA Lead | 1 day | May 13 |

## **9. Test Deliverables**

* Test Plan
* Test Cases (Manual + Automated)
* Execution Report
* Defect Report
* Performance Report
* Test Summary Document

## **10. Entry and Exit Criteria**

### **Requirement Analysis**

* **Entry:** Swagger API documentation available
* **Exit:** Test scenarios documented and reviewed

### **Test Execution**

* **Entry:** Stable build on environment, test cases signed off
* **Exit:** Execution complete, all major defects resolved

### **Test Closure**

* **Entry:** All test artifacts ready
* **Exit:** Test Summary Report submitted, sign-off received

## **11. Tools**

* **JIRA** – Bug tracking
* **Postman/Newman** – Manual and automated API testing
* **JMeter** – Load testing
* **Swagger UI** – API documentation and quick tests
* **Snipping Tool** – Screenshots
* **Excel/Word** – Documentation
* **MindMap Tool** – Scenario mapping

## **12. Risks and Mitigations**

| **Risk** | **Mitigation** |
| --- | --- |
| Resource Unavailability | Distribute test cases among available QAs |
| API Not Responding | Coordinate with Dev team; use mock server |
| Token Expiration | Automate token generation for test scripts |
| Timeline Delay | Prioritize critical and high-usage test flows |

## **13. Approvals**

**Documents for Approval:**

* Test Plan
* Test Scenarios
* Test Cases
* Defect and Summary Reports

**Approvers:**

* QA Lead
* Product Owner
* Developer Lead
* Project Manager