**Test Plan for Restful Booker API**

Created by: Nagraj Bitla

Version: 1.0

Date: [Date]

Application Under Test (AUT): Restful Booker API

Test Type: API Testing

Test Approach: Manual & Automated Testing

1. Objective

This test plan ensures the quality, functionality, reliability, security, and performance of the Restful Booker API, which provides booking management functionality with authentication.

2. Scope

Features to be tested:

Booking Management (CRUD)

Create Booking (POST /booking)

Get Booking (GET /booking/{id})

Update Booking (PUT /booking/{id})

Partial Update Booking (PATCH /booking/{id})

Delete Booking (DELETE /booking/{id})

Authentication & Authorization

Create Token (POST /auth)

System Health Check

Ping (GET /ping)

Health Check (GET /health)

Booking Search

Get Booking IDs (GET /booking)

Types of Testing:

Functional Testing

Regression Testing

Smoke Testing

Performance Testing

Security Testing

API Contract Testing (JSON Schema Validation)

Environments:

Operating Systems: Windows, macOS, Linux

Browsers: Chrome, Firefox, Edge

API Clients: Postman, cURL, REST Assured

Network: Wi-Fi, Cellular

Security Measures: Authentication via Basic Auth & Token Auth

Evaluation Criteria:

Response Codes (200, 400, 401, 404, 500, etc.)

API Response Time & Performance

JSON Response Structure Validation

Authentication Mechanism Functionality

3. Inclusions

Testing API endpoints against different request payloads.

Checking error handling and validation messages.

Ensuring proper authentication and authorization.

Response time verification under different loads.

Validating API responses against expected schema.

4. Exclusions

UI testing (API testing only).

Payment Gateway functionality (if applicable).

5. Test Environment

Base URL: https://restful-booker.herokuapp.com

Authentication: Basic Auth & Token Auth

API Headers:

Content-Type: application/json

Accept: application/json

6. Defect Reporting Procedure

Criteria for defects: Incorrect responses, API downtime, unauthorized access, broken authentication.

Bug tracking tool: JIRA

Bug severity classification:

Critical: API is non-functional.

High: Major functionality is impacted.

Medium: Minor API issues or incorrect responses.

Low: UI inconsistencies in response formatting.

7. Test Strategy

Step 1: Test Case Design

Techniques Used:

Equivalence Partitioning: Test valid and invalid values.

Boundary Value Analysis: Validate edge cases.

Negative Testing: Check error handling.

State Transition Testing: For booking status updates.

Step 2: Test Execution Procedure

Smoke Testing: Verify if all critical APIs are working.

Functional Testing: Validate CRUD operations and authentication.

Regression Testing: Ensure new changes do not impact existing APIs.

Performance Testing: Validate response time with 1000+ requests per second.

Step 3: Defect Reporting

Log issues in JIRA with priority and severity levels.

Provide API request & response details with logs.

8. Test Schedule

Task Time Duration

Test Plan Creation 2 days

Test Case Design 3 days

Test Execution 5 days

Regression Testing 2 days

Performance Testing 2 days

Bug Fix Verification 2 days

Test Summary Report 1 day

9. Test Deliverables

Test Plan Document

Test Cases

Bug Reports (JIRA)

Test Summary Report

10. Entry & Exit Criteria

Phase Entry Criteria Exit Criteria

Requirement Analysis API documentation is ready Test scenarios identified

Test Execution API is deployed & stable All test cases executed

Test Closure Defects are resolved Final test report submitted

11. Tools Used

Postman (Manual API Testing)

JIRA (Bug Tracking)

REST Assured (Automation)

JMeter (Performance Testing)

12. Risks & Mitigations

Risk Mitigation

API Downtime Use mock APIs for testing

Rate Limiting Issues Implement request throttling tests

Security Vulnerabilities Conduct penetration testing

13. Approvals

Document Approval By

Test Plan Test Lead

Test Scenarios QA Manager

Test Summary Report Project Manager

Test Cases for Create Booking API (POST /booking)

TC\_ID Test Scenario Request Details Expected Response Status

TC\_001 Create booking with valid data Send valid JSON payload 201 Created with booking ID ✅

TC\_002 Missing firstname Remove "firstname" from payload 400 Bad Request ✅

TC\_003 Missing lastname Remove "lastname" from payload 400 Bad Request ✅

TC\_004 Missing totalprice Remove "totalprice" 400 Bad Request ✅

TC\_005 totalprice as negative number "totalprice": -100 400 Bad Request ✅

TC\_006 totalprice as string "totalprice": "hundred" 400 Bad Request ✅

TC\_007 Missing bookingdates Remove "bookingdates" 400 Bad Request ✅

TC\_008 Invalid checkin format "checkin": "01-01-2018" 400 Bad Request ✅

TC\_009 checkin after checkout "checkin": "2019-01-01", "checkout": "2018-01-01" 400 Bad Request ✅

TC\_010 Missing depositpaid Remove "depositpaid" 201 Created (assumes default value) ✅

TC\_011 Additional unexpected field Add "extraField": "test" API ignores field & returns 201 ✅

TC\_012 Large firstname field "firstname": "A" \* 1000 400 Bad Request ✅

TC\_013 SQL Injection Test "firstname": "' OR 1=1 --" API must prevent SQL injection ✅

TC\_014 XSS Attack Test "firstname": "<script>alert(1)</script>" API must sanitize input ✅

TC\_015 Invalid Content-Type Content-Type: text/plain 415 Unsupported Media Type ✅

Conclusion

This test plan ensures that the Restful Booker API is thoroughly tested for functionality, security, performance, and reliability. It covers positive and negative scenarios, authentication, and API response validation.

Let me know if you need further refinements!