Test Plan for Restful Booker API

1. Introduction

1.1 Purpose

This document outlines the testing strategy for the **Restful Booker API** (hosted at https://restful-booker.herokuapp.com) to ensure functionality, reliability, and security for a fictional hotel booking system.

1.2 Objectives

- Validate CRUD operations (Create, Read, Update, Delete) for bookings.
- Verify authentication, error handling, and data validation.
- Ensure performance, security, and compatibility.

2. Scope

2.1 Inclusions

Testing Type	**Coverage**
Functional Testing	All API endpoints (POST/GET/PUT/DELETE) with valid/invalid inputs.
Data Validation	Boundary values, mandatory fields, and data-type checks.
Error Handling	HTTP status codes (4xx/5xx) and error messages for malformed requests.
Security Testing	SQL injection, XSS, authentication bypass, and HTTPS compliance.
Performance Testing	Response time under load (e.g., 100 concurrent users).
Integration Testing	Interactions between booking, authentication, and payment endpoints.

2.2 Exclusions

- UI/UX testing (purely API-focused).
- Third-party integrations (unless specified).
- Long-term endurance testing.

3. Test Environments

3.1 Host URLs

Environment	**URL**
QA	https://restful-booker.herokuapp.com
Pre-Prod	https://restful-booker.herokuapp.com

3.2 Platforms & Tools

Category	**Details**
OS	Windows 10, macOS, Linux
Browsers	Chrome, Firefox, Edge, Safari
Mobile Devices	Android (Chrome), iOS (Safari)
Tools	Postman, RestAssured, JIRA (defect tracking), JMeter (performance testing).

4. Test Strategy

4.1 Test Design Techniques

- **Equivalence Partitioning**: Group valid/invalid inputs (e.g., booking dates).
- **Boundary Value Analysis**: Test min/max values for input fields.
- **Error Guessing**: Validate edge cases (e.g., empty payloads).

4.2 Phases

1. **Smoke Testing**: Verify critical endpoints (e.g., POST /booking).
2. **Regression Testing**: Post-bug fixes.
3. **Exploratory Testing**: Ad-hoc scenarios (e.g., concurrency issues).

5. Test Schedule

Task	**Duration**	**Owner**
Test Plan Creation	2 days	QA Lead
Test Case Design	5 days	QA Team
Test Execution (Sprint 1)	10 days	QA Team
Defect Triage & Retesting	3 days	Dev/QA Team
Final Report Submission	1 day	QA Lead

6. Entry/Exit Criteria
6.1 Entry Criteria

- Requirements documented and approved.
- Test environment ready.
- Test data prepared.

6.2 Exit Criteria

- All P0/P1 test cases pass.
- ≤ 5% critical defects unresolved.
- UAT sign-off received.

7. Defect Management
7.1 Reporting

- **Tool**: JIRA.
- **Fields**: Steps to reproduce, severity (P0-P2), screenshots/logs.

7.2 Roles

Role	**Responsibility**
QA Tester	Log defects with detailed repro steps.
Developer	Fix and retest.
QA Lead	Prioritize and track defect resolution.

8. Risks & Mitigations

Risk	**Mitigation**
Unstable test environment	Use Docker for local backups.
Tight deadlines	Prioritize P0 test cases first.
Incomplete requirements	Conduct daily syncs with stakeholders.

9. Tools

Purpose	**Tool**
API Testing	Postman, RestAssured
Performance Testing	JMeter
Defect Tracking	JIRA
Documentation	Confluence