**Test Plan for Restful Booker API**

**1. Introduction**

**1.1 Purpose**

This test plan defines the scope, approach, resources, and schedule for testing the **Restful Booker API** available at [**https://restful-booker.herokuapp.com/**](https://restful-booker.herokuapp.com/). The purpose is to ensure the API functions correctly, meets requirements, and is secure, reliable, and performant.

**1.2 Scope**

The test plan covers functional, non-functional, security, and performance testing of the following API endpoints:

* **Authentication:** POST /auth
* **Booking Management:** GET /booking, GET /booking/{id}, POST /booking, PUT /booking/{id}, PATCH /booking/{id}, DELETE /booking/{id}
* **Health Check:** GET /ping

**1.3 Test Objectives**

* Validate that API responses conform to expected input/output requirements.
* Verify that CRUD operations work as expected.
* Ensure authentication and authorization mechanisms function correctly.
* Assess API performance under various load conditions.
* Identify security vulnerabilities and edge cases.

**2. Test Approach**

**2.1 Testing Types**

* **Functional Testing:** Verify API request-response behavior and data validation.
* **Negative & Edge Case Testing:** Handle missing fields, invalid data, and boundary conditions.
* **Security Testing:** Ensure authentication, authorization, and data protection.
* **Performance Testing:** Measure response times under normal and high load conditions.
* **Integration Testing:** Validate interactions with other dependent services.

**2.2 Test Tools**

* **Postman** – API testing and automation
* **JUnit/TestNG** – Automated test execution
* **JMeter** – Performance and load testing
* **OWASP ZAP** – Security testing

**3. Test Environment**

**3.1 Required Tools & Configurations**

* API test environment: https://restful-booker.herokuapp.com/
* Supported browsers for API clients: Chrome, Firefox
* Postman for manual API validation
* Automation frameworks: RestAssured, Cypress (optional)

**3.2 Test Data**

Test data will include:

* Sample valid and invalid booking details.
* Authentication credentials (valid and invalid tokens).
* Special characters, boundary values, and edge cases for stress testing.

**4. Test Cases**

**4.1 Functional Test Cases**

| **Test Case ID** | **Scenario** | **Input** | **Expected Output** |
| --- | --- | --- | --- |
| TC\_01 | Create Booking | Valid JSON payload | 200 OK, booking ID returned |
| TC\_02 | Retrieve Booking by ID | Valid booking ID | 200 OK, correct booking details |
| TC\_03 | Update Booking | Valid booking ID and JSON payload | 200 OK, updated booking details |
| TC\_04 | Delete Booking | Valid booking ID | 201 Created, booking removed |
| TC\_05 | Authentication | Valid credentials | 200 OK, token returned |
| TC\_06 | Authentication | Invalid credentials | 403 Forbidden, error message |

**4.2 Negative & Edge Case Test Cases**

| **Test Case ID** | **Scenario** | **Input** | **Expected Output** |
| --- | --- | --- | --- |
| TC\_07 | Create Booking with Missing Fields | JSON missing required fields | 400 Bad Request, validation error |
| TC\_08 | Retrieve Booking with Invalid ID | Non-existent booking ID | 404 Not Found |
| TC\_09 | Update Booking with Unauthorized Access | Invalid auth token | 403 Forbidden |
| TC\_10 | Large Booking Data | Payload with extremely long strings | 413 Payload Too Large or handled correctly |

**5. Defect Management**

**5.1 Bug Tracking**

All defects found during testing will be logged and tracked in **JIRA/GitHub Issues**.

* **Priority Levels:** Critical, High, Medium, Low
* **Severity Levels:** Blocker, Major, Minor, Trivial

**6. Exit Criteria**

* All high-priority and critical defects are resolved.
* Functional test cases pass successfully.
* Performance and security issues are within acceptable limits.
* API meets acceptance criteria outlined in requirements.

**7. Test Deliverables**

* Test Plan
* Test Cases (Excel/Automated Scripts)
* Bug Reports
* Test Summary Report

**8. Schedule & Timeline**

| **Phase** | **Start Date** | **End Date** |
| --- | --- | --- |
| Test Planning | Day 1 | Day 3 |
| Test Case Design | Day 4 | Day 6 |
| Test Execution | Day 7 | Day 12 |
| Defect Fixing & Retesting | Day 13 | Day 15 |
| Test Closure & Reporting | Day 16 | Day 17 |

**9. Risks & Mitigation**

| **Risk** | **Mitigation** |
| --- | --- |
| API downtime | Test against local mock server when needed |
| Unstable environment | Run tests in a dedicated environment |
| Unexpected changes in API | Maintain versioned API tests |

**10. Approval**

| **Name** | **Role** | **Approval Status** |
| --- | --- | --- |
| Tester | QA Engineer | Pending |
| Test Lead | QA Manager | Pending |
| Product Owner | Business Analyst | Pending |

**End of Test Plan**