**Test Plan for Event Management API**

**1. Scope & Objective**

Validate functionality, reliability, and robustness of the Event Management API.

Endpoints Covered:

* POST: http://{baseurl}/v1/bookings – Book an event.
* GET: http:// {baseurl}/v1/events – List all events.
* PUT: http:// {baseurl}/v1/bookings/{bookingReference}/cancel – Cancel a booking.

**2. Key Functional Test Scenarios**

1. **Event Listing**

* Retrieve all available events (status 200, correct structure).
* Response includes key fields: theme, reference, amount, status, venue, totalCapacity, startTime, endTime, etc.

1. **Event Booking**

* Book an event with available capacity.
* Book an event that is fully booked (should be waitlisted or error).
* Verify booking details in response (fee, reference, status, event Name).
* Booking with and without shouldWaitList.

1. **Cancel Booking**

* Successfully cancel a valid booking.
* Attempt to cancel already cancelled booking.
* Confirm booking status is updated after cancellation.

**3. Edge Cases & Negative Tests**

* Book an event with an invalid or non-existent event reference.
* Attempt booking when userId is missing or invalid.
* Book event with malformed request body (e.g., missing booking type).
* Cancel booking with invalid or non-existent booking reference.
* Cancel booking for an event already ended.
* Try booking/cancelling without authentication (if implemented).
* Attempt to overbook when capacity is zero.

**4. Performance Testing Suggestions**

* Simulate high load by sending multiple booking requests concurrently.
* Test event listing with large datasets (if supported).
* Measure response times for booking and cancellation operations.

**5. Security Testing Suggestions**

* Test API with invalid/missing userId to verify authentication/authorisation (even if not fully implemented).
* Attempt SQL injection or malformed input in request body.
* Check for sensitive data leakage in API responses.
* Validate HTTP headers: X-Frame-Options, X-Content-Type-Options, X-XSS-Protection.
* Attempt booking/cancellation by one user for another user's booking (if applicable).

**6. Test Automation Tools**

* **Java + RestAssured + JUnit:** (recommended; aligns with company stack)
* RestAssured: API testing
* JUnit: test runner, assertions
* Maven/Gradle: build & dependencies
* **Alternative:** Python + Pytest + Requests (if preferred)

**Why:**

* Widely used for API automation.
* Supports data-driven testing and reporting.
* Good integration with CI/CD.

**7. Data-Driven Tests**

* Parameterise tests with various event references, booking types, and shouldWaitList values
* Run booking/cancellation tests for both open and fully booked events.

**8. Test Setup & Teardown**

* Before each test: Ensure test data exists (e.g., list events, pick an available one).
* After each test: Clean up created bookings where possible.