Logic Breakdown

System-Only Processes (22)

1. **Auto-release locked seats**
   Automatically releases seats after a timeout if checkout is not completed, preventing seat hoarding.

2. **Concurrency resolution**
   Resolves race conditions when multiple users attempt to book the same seat simultaneously.

3. **Inventory synchronization**
   Keeps cached seat availability aligned with the primary database to prevent stale data.

4. **Dynamic pricing adjustment**
   Updates ticket prices based on demand and predefined pricing rules.

5. **Tax calculation**
   Automatically applies region-specific taxes during checkout.

6. **Payment retry logic**
   Retries failed transactions safely without causing duplicate charges.

7. **Email generation**
   Generates booking confirmations, tickets, and cancellation emails automatically.

8. **SMS OTP generation**
   Generates and sends one-time passwords for authentication and verification.

9. **Fraud detection execution**
   Continuously evaluates transactions and behavior to detect suspicious activity.

10. **Report generation**
    Produces sales, booking, and performance reports on a scheduled basis.

11. **Session cleanup**
    Terminates inactive sessions to free system resources and reduce risk.

12. **Cache invalidation**
    Clears outdated cache entries after booking, cancellation, or seat updates.

13. **Recommendation engine execution**
    Generates personalized event recommendations asynchronously.
14. **Waitlist management**
    Activates waitlists when events are sold out and reallocates seats when available.
15. **Event reminder scheduling**
    Schedules reminders based on event date and user preferences.
16. **Refund calculation**
    Computes refund amounts based on cancellation policies and timing.
17. **Data archival**
    Archives historical booking data for compliance and analytics.
18. **Audit logging**
    Records critical system actions for traceability and compliance.
19. **Capacity alerts**
    Triggers alerts when demand approaches venue or system limits.
20. **Session persistence handling**
    Maintains session state across network interruptions.
21. **Rate limiting**
    Restricts excessive requests to prevent abuse and system overload.
22. **Database replication**
    Replicates data across nodes to ensure fault tolerance and availability.

Conditional Branches (8)

1. **Payment method routing**
   Routes transactions to the appropriate gateway based on selected payment method.
2. **Seat availability display logic**
   Determines whether seats are shown as available, locked, waitlisted, or sold out.
3. **User authentication state check**
   Controls access to personalized features based on login status.
4. **Refund eligibility evaluation**
   Applies refund rules based on time remaining before the event.

5. **Error recovery decision logic**
   Determines whether to retry, rollback, or abort an operation after failure.

6. **Accessibility feature activation**
   Enables alternate views or interaction modes when accessibility needs are detected.

7. **Alternate seat suggestion logic**
   Suggests replacement seats when selected seats become unavailable.

8. **Fraud risk threshold evaluation**
   Flags or blocks transactions exceeding predefined risk scores.

Error States (10)

1. **Seat no longer available (SEA_001)**
   Occurs when another user completes booking first.

2. **Insufficient payment funds (PAY_002)**
   Triggered when the payment method lacks sufficient balance.

3. **Invalid or expired card (PAY_003)**
   Raised when card details fail validation.

4. **Session expired (SES_001)**
   Occurs after prolonged user inactivity.

5. **Venue not found (VEN_001)**
   Triggered when a venue is removed or unpublished.

6. **Duplicate booking detected (BKG_001)**
   Prevents repeated bookings for the same user and event.

7. **Invalid OTP entered (SEC_001)**
   Raised when authentication codes do not match.

8. **Payment gateway timeout (GWY_001)**
   Occurs when the payment processor does not respond.

9. **Database connection lost (DB_001)**
   Triggered during backend connectivity failures.

10. **Concurrent booking conflict (CON_001)**
    Occurs during race conditions under high traffic.

Edge and Exception Cases (8)

1. **Last-minute seat conflict**
   High-demand scenarios where many users attempt to book the final seat.

2. **Event cancellation**
   Triggers automated refunds and notifications.

3. **Traffic spike**
   Sudden surge in users requiring rate limiting and load balancing.

4. **Currency fluctuation**
   Exchange rates are locked during checkout to prevent pricing mismatch.

5. **Internet disconnection during payment**
   Requires server-side validation and safe retry handling.

6. **Card expiration during checkout**
   Detected in real time to prevent settlement failure.

7. **Timezone boundary issues**
   All event times are normalized to UTC to avoid inconsistencies.

8. **Accessibility change mid-booking**
   Allows preference updates without restarting the booking flow.

Backend Triggers and Automated Processes (6)

1. **Seat lock timeout trigger**
   Automatically releases locked seats after timeout expiration.

2. **Scheduled inventory synchronization**
   Periodically syncs cache and database inventory.

3. **Dynamic pricing update trigger**
   Adjusts prices based on demand thresholds.

4. **Payment retry trigger**
   Initiates retries after transient payment failures.

5. **Notification dispatch trigger**
   Sends confirmations, reminders, and alerts automatically.

6. **Fraud monitoring trigger**
   Continuously scans transactions and behavior in the background.