**Assignment: Implement a Ticket Booking System for Events in Java**

**Objective: Build a ticket booking system for events where users can reserve, cancel, and view bookings for various events. This system will help you practice OOP design, multithreading, and handling concurrency in real-time as multiple users interact with event availability simultaneously.**

**System Requirements:**

1. **Event Class**
   * **Attributes:**
     + **eventId (int): Unique identifier for each event.**
     + **eventName (String): Name of the event (e.g., "Concert," "Theater Play").**
     + **totalSeats (int): Total number of seats available for the event.**
     + **bookedSeats (int): Number of seats already booked.**
     + **pricePerTicket (double): Price per ticket for the event.**
   * **Methods:**
     + **getAvailableSeats(): Returns the number of remaining seats.**
   * **Override Methods:**
     + **Override toString() to display event details.**
     + **Override equals() and hashCode() to identify events by eventId.**
2. **Booking Class**
   * **Attributes:**
     + **bookingId (int): Unique identifier for each booking.**
     + **userName (String): Name of the user making the booking.**
     + **eventId (int): ID of the event being booked.**
     + **numberOfTickets (int): Number of tickets booked by the user.**
     + **status (String): Status of the booking (e.g., "Confirmed," "Cancelled").**
3. **EventManager Class**
   * **Manages events and bookings using HashMap<Integer, Event> and HashMap<Integer, Booking>.**
   * **Core Operations:**
     + **Add Event: Add a new event to the system. If an event with the same eventId exists, throw an EventAlreadyExistsException.**
     + **Book Tickets: Allow a user to book tickets for an event. If the event doesn’t have enough available seats, throw a SeatsNotAvailableException.**
     + **Cancel Booking: Allow a user to cancel a booking by bookingId. If the booking doesn’t exist or is already canceled, throw a BookingNotCancelableException.**
     + **View Event Availability: Display all events along with their available seats.**
4. **Custom Exception Handling**
   * **Define custom exceptions:**
     + **EventAlreadyExistsException: Thrown when trying to add an event that already exists in the system.**
     + **SeatsNotAvailableException: Thrown when there are not enough seats available for a booking request.**
     + **BookingNotCancelableException: Thrown when trying to cancel a non-existent or already canceled booking.**
5. **Multithreading**
   * **Implement separate Runnable classes for each core operation (AddEvent, BookTickets, CancelBooking, ViewEventAvailability).**
   * **Run each operation in its own thread, with the main thread waiting for each operation to complete using join().**
6. **Synchronization**
   * **Synchronize the booking and cancellation operations to avoid race conditions when multiple users attempt to book or cancel tickets simultaneously.**

**Bonus Tasks:**

1. **Update Event Details: Allow updating the price per ticket or total seats for an event.**
2. **Search Events by Name: Enable users to search for specific events by name.**
3. **Revenue Calculation: Calculate total revenue for each event based on confirmed bookings.**