Ex 1: Use Case Diagram

Necessary Elements for the Online Event Ticketing System

Actors*:*

- **Customer:** Browses events, purchases tickets, and checks in at events.
- **Event Organizer:** Creates and manages events, provides event details.
- **System Administrator:** Manages the system's backend operations.
- **Payment Gateway:** Processes payments securely.
- **Check-in Staff:** Scans QR codes at the venue for validation.

Classes:

1. **Event:**
    - Attributes: Event ID, Name, Description, Date, Time, Venue, Ticket Price, Available Seats.
    - Methods: CreateEvent(), UpdateEvent(), ViewDetails().
2. **Customer:**
    - Attributes: Customer ID, Name, Email, Phone Number.
    - Methods: Register(), Login(), BrowseEvents(), PurchaseTicket(), ViewTickets().
3. **Ticket:**
    - Attributes: Ticket ID, Event ID, Customer ID, QR Code, Status (valid/invalid).
    - Methods: GenerateQRCode(), ValidateTicket().
4. **Payment:**
    - Attributes: Payment ID, Amount, Payment Status, Transaction Details.
    - Methods: ProcessPayment(), RefundPayment().
5. **CheckIn:**
    - Attributes: CheckIn ID, Event ID, Ticket ID, Timestamp.
    - Methods: ScanQRCode(), ValidateEntry().
6. **System:**
    - Attributes: System ID, Name, Version.
    - Methods: SendEmail(), GenerateReports().

States:

- **Customer States:**
    - Browsing Events
    - Purchasing Tickets
    - Viewing Tickets
    - Checking In
- **Ticket States:**
    - Not Purchased
    - Purchased
    - Canceled
    - Checked In

Activities*:*

1. **Browse Events:**
    - Customer searches for events by date, location, or category.
    - Event details are displayed.
2. **Purchase Tickets:**
    - Customer selects an event and ticket type.
    - Makes a payment through the gateway.
    - Receives an electronic ticket with a QR code.
3. **Check-in at Event:**
    - Customer presents QR code.
    - Check-in staff scans QR code for validation.
4. **Create and Manage Events:**
    - Event organizers add, edit, or delete event details.
5. **Process Payment:**
    - Secure payment through the gateway.
    - Refunds if cancellation occurs.

System Activities:

- **Generate QR Code:** System generates QR codes for tickets.
- **Send Notifications:** System sends email confirmations and reminders.
- **Validate Tickets:** System verifies ticket authenticity during check-in.

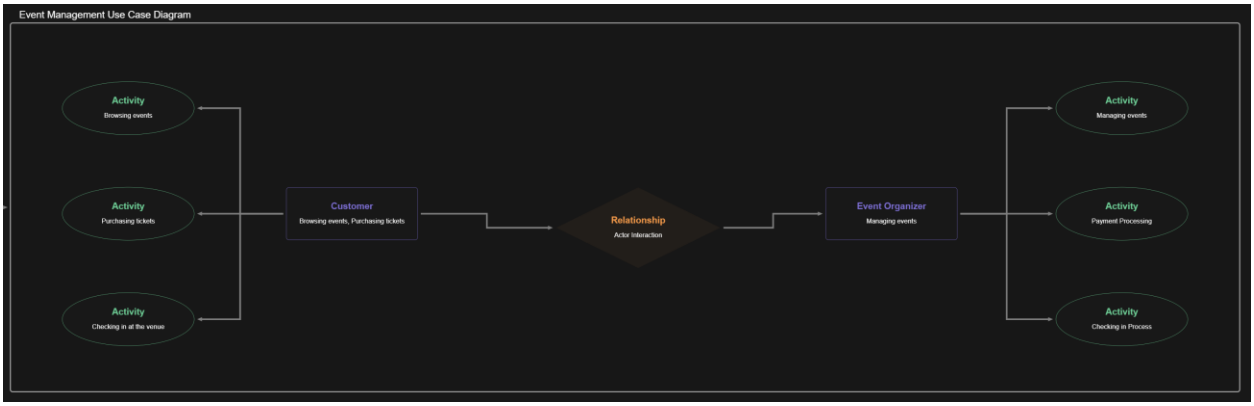

Fig. 1. Use Case Diagram

Ex 2: Class Diagram

➢ **Main Classes:**

- **Event:** EventID, Name, Description, Date, Time, Venue, TicketPrice, AvailableSeats.
- **Customer:** CustomerID, Name, Email, PhoneNumber.
- **Ticket:** TicketID, EventID, CustomerID, QRCode, Status.
- **Payment:** PaymentID, Amount, PaymentStatus, TransactionDetails.
- **QRCodeGenerator:** QRCodeData, GenerateQRCode().
- **System:** Name, Version, SendNotifications().

➢ **Relationships:**

- **Customer** associates with **Ticket** (1-to-many).
- **Event** aggregates **Ticket** (1-to-many).
- **Ticket** uses **QRCodeGenerator**.
- **Payment** is associated with **Customer** and **Ticket**.

Fig. 2. Class Diagram

Ex 4: State Machine Diagram

1. **Entity:** Ticket.
2. **States:**
    - Available
    - Reserved
    - Purchased
    - CheckedIn
    - Invalid
3. **Transitions:**
    - Reserved -> Purchased (on payment confirmation).
    - Purchased -> CheckedIn (on QR Code scan at the venue).
    - Any state -> Invalid (on cancellation or failure).

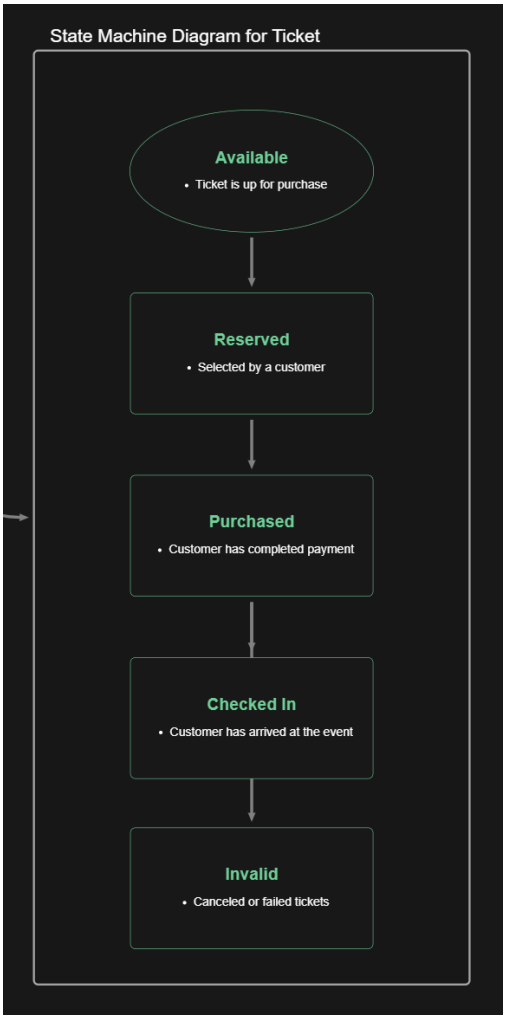

Fig. 3. State Machine Diagram

Ex 5: Activity Diagram

1. **Process:** Ticket purchasing and event check-in.
2. **Activities:**
    - Customer selects an event.
    - System processes the payment.
    - System generates a QR Code for the ticket.
    - QR Code is sent to the customer.
    - Customer presents QR Code at the venue.
    - System validates the QR Code.
    - Customer is checked in.

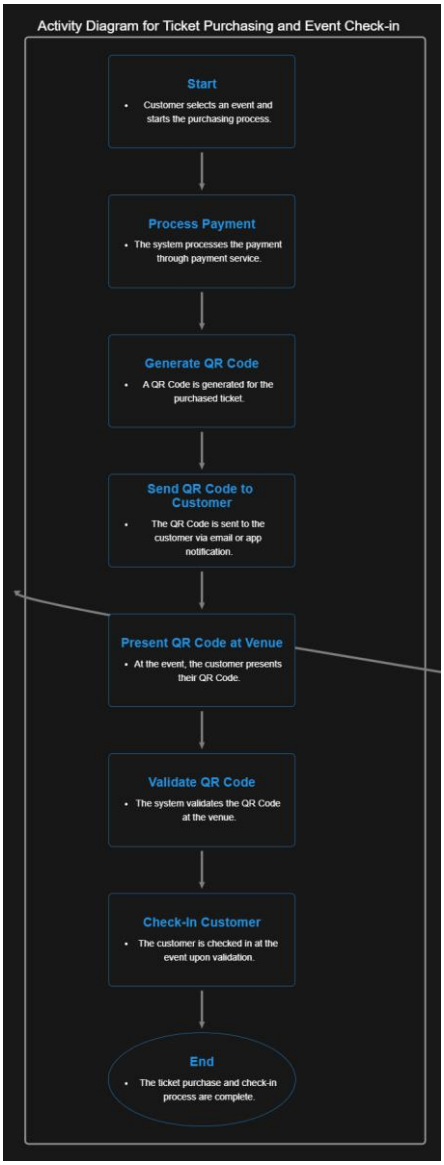

Fig. 4. Activity Diagram