Smart Event Management System

Detailed Design Document

1. Physical Architecture

The Smart Event Management System is structured to ensure seamless interaction between users (customers and admins) and the underlying technology. The architecture includes the following layers:

1.1 Frontend

The frontend is the user-facing part of the system, designed for both customers and admins. It provides a clean, intuitive, and responsive interface.

- **Customer Interface**:
  - Allows customers to search for shows based on filters like location, genre, and date.
  - Enables ticket booking with seat selection and cancellation options.
  - Provides access to their purchase history.
- **Admin Interface**:
  - Lets admins create, update, and manage shows.
  - Displays reports and analytics to aid decision-making.

1.2 Backend

The backend serves as the brain of the system, handling all server-side logic. It ensures smooth operations by managing user authentication, ticket transactions, dynamic pricing, and report generation.

- Developed using the Django REST framework for robust API functionality.
- Handles communication between the frontend and the database or other subsystems.

1.3 Database

The database is the storage layer, maintaining essential information about users, shows, seats, and transactions. Key features include:

- **Relational structure**: Ensures logical data organization and efficient retrieval.
- **Scalability**: Supports growing data volumes with high performance.

- **Backup and recovery**: Safeguards data integrity and minimizes downtime.

1.4 Dynamic Pricing Engine

This subsystem dynamically adjusts ticket prices based on:

- Historical data.
- Current sales trends.
- Time left until the event. The goal is to maximize revenue while offering competitive pricing.

1.5 Reports Engine

The reports engine generates detailed analytics for admins to:

- Monitor sales performance.
- Identify trends in customer preferences.
- Optimize pricing and show management.

2. Component Descriptions

2.1 Frontend

The frontend is built using modern web technologies to ensure responsiveness across devices. Pages include:

- **Search Page**: Allows filtering by genre, location, and time.
- **Booking Page**: Provides seat selection and payment options.
- **Admin Dashboard**: Displays tools for managing shows and viewing analytics.

2.2 Backend

The backend connects the frontend with core logic and the database. It includes:

- APIs for data interaction.
- Security features like JWT-based authentication.
- Real-time updates for ticket availability.

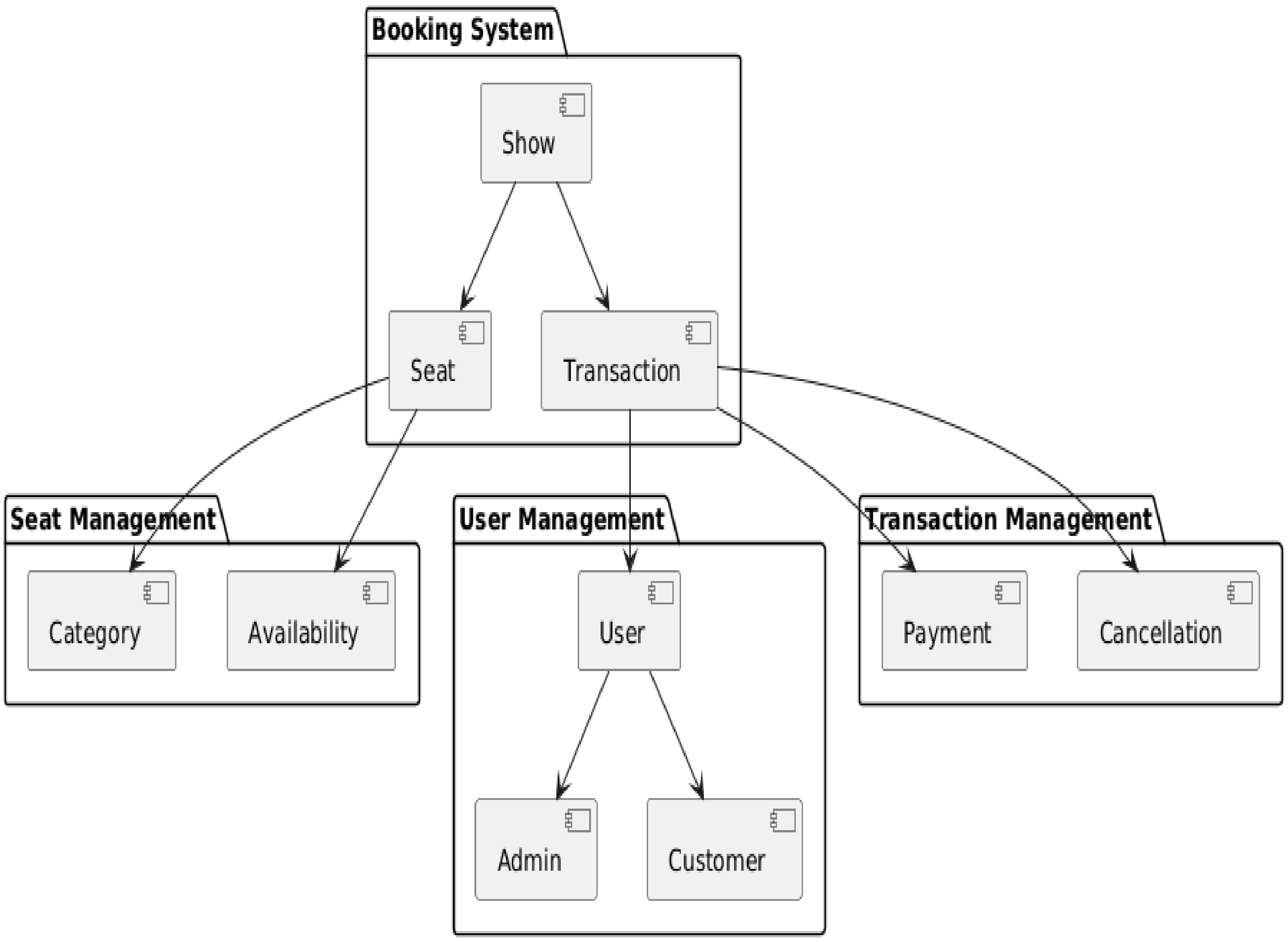
2.3 Database

The database includes tables for:

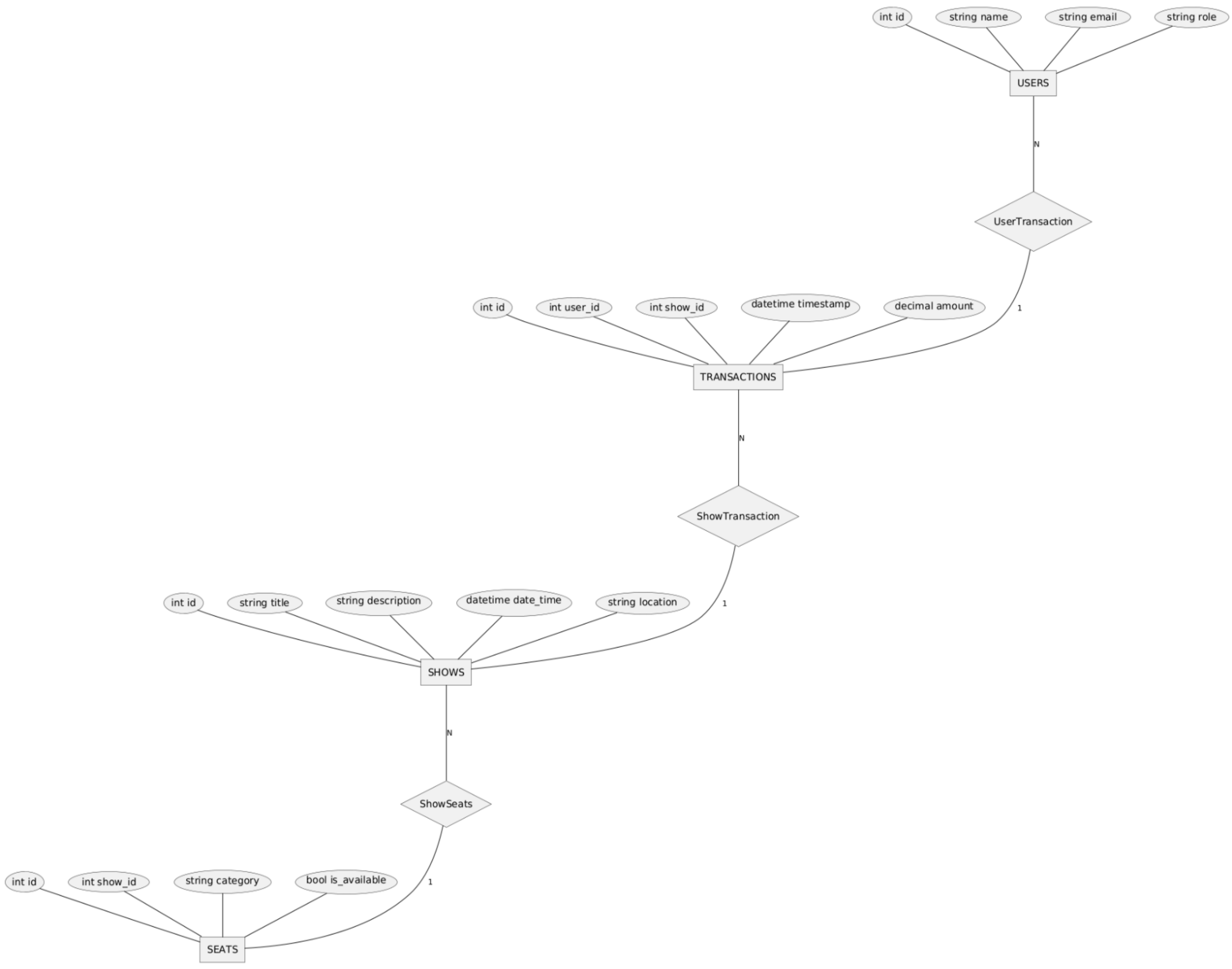
- Users: Stores customer and admin details.

- Shows: Tracks event data.
- Seats: Manages seat availability and categories.
- Transactions: Logs purchases and cancellations.

3. Component Diagram



4. Entity-Relationship Diagram (ERD)



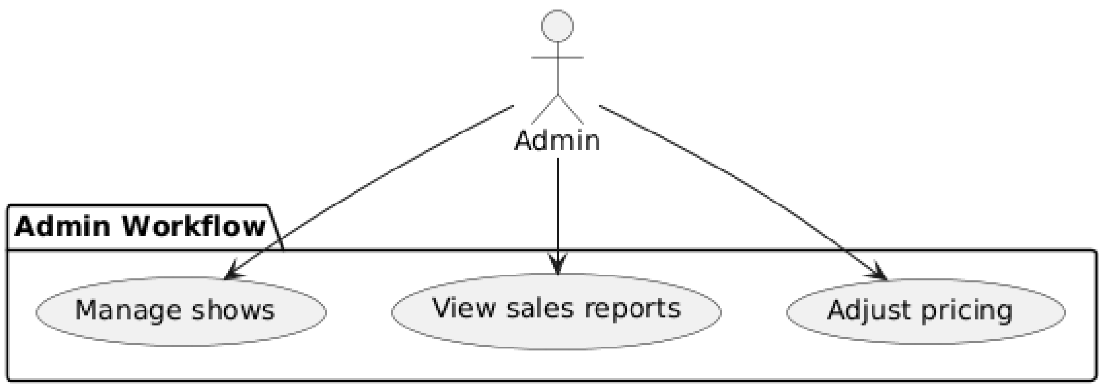
ERD Explanation

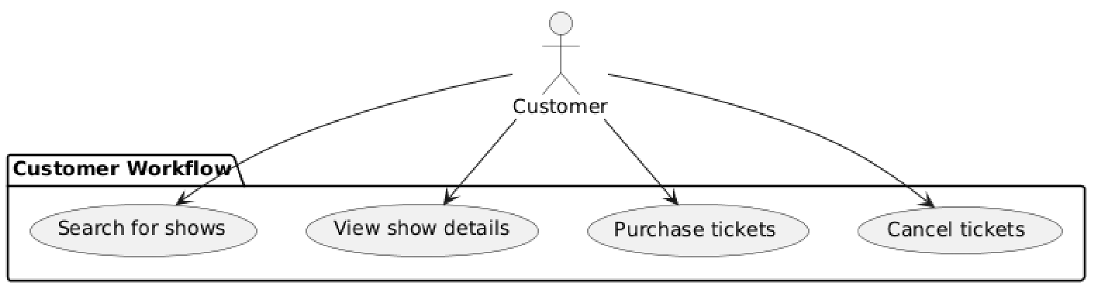
The ERD outlines the logical relationships in the system:

- **Users**: Differentiates between admins and customers. Admins manage the system, while customers book tickets.
- **Shows**: Contains event-specific information, enabling filtering and booking.
- **Seats**: Tracks seat categories and availability, supporting dynamic pricing and efficient bookings.
- **Transactions**: Links users to their bookings and maintains a record of payments and cancellations.

This structure ensures consistency, integrity, and efficiency in data operations.

5. Use Case Diagrams





6. Algorithms

6.1 Dynamic Pricing Algorithm

This algorithm ensures optimized ticket pricing based on demand and market trends.

Inputs:

- Historical ticket prices for similar shows.
- Real-time ticket sales data.
- Remaining time until the show starts.

Process:

1. Collect historical price data and normalize it.
2. Analyze current ticket sales velocity.
3. Incorporate time-based adjustments (e.g., closer to event = higher price).
4. Compute a weighted average and adjust based on market trends.

Output:

- A dynamic price range suggestion, ensuring competitive and profitable pricing.

Advantages:

- Maximizes revenue without overpricing.
- Adjusts to changing demand in real-time.

6.2 Seat Availability Algorithm

Ensures real-time tracking and updates of seat status.

Process:

1. Query the database for current seat availability.
2. Update seat status upon booking or cancellation.
3. Provide instant feedback to users on available options.

7. Security and Usability Considerations

7.1 Security

- All sensitive data is encrypted.
- HTTPS is enforced for secure data transmission.
- Role-based access control ensures only authorized access to admin functionalities.

7.2 Usability

- Accessible design ensures compatibility with screen readers.
- Responsive interfaces adapt seamlessly to mobile and desktop platforms.

8. Future Extensions

- Integration with multiple payment gateways for customer convenience.
- Advanced AI-based demand forecasting for ticket sales.
- Multilingual support to reach a global audience.