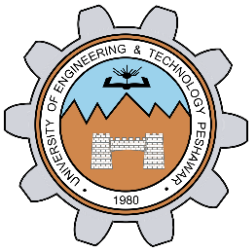
Project Milestone 1



Spring 2025

Group Members

Name	Registration Number
Hassan Zaib Jadoon	22PWCSE2144
Ahsan Raza	22PWCSE2099
Mutahhar Fayyaz	22PWCSE2176

"On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work."

Submitted to:

Engr. Summeyea Salahuddin

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

Database Design

Conceptual Design

The database design follows a normalized approach ensuring data integrity, reducing redundancy, and maintaining referential consistency. The schema supports complex relationships between entities while maintaining performance.

Entity Relationship Diagram

Entity Name	Attributes	Primary Key	Foreign Key(s)	Relationship Types
Users	id, name, email, password, created_at	id	-	**Disjoint specialization** with Admins and Normal Users
Admins	id, name, email, password, created_at	id	-	**One-to-Many** with Events
Events	id, admin_id, title, description, date, time, venue, available_seats, price, created_at	id	admin_id → Admins(id)	**Many-to-One** with Admins, **One-to-Many** with Bookings, **One-to-Many** with Seats
Bookings	id, event_id, customer_name, customer_email, customer_phone, status, booking_date	id	event_id → Events(id)	**Many-to-One** with Events, **Many-to-Many** with Seats (via Booking_Seats)
Seats	id, event_id, seat_number, section, price, status, created_at	id	event_id → Events(id)	**Many-to-One** with Events, **Many-to-Many** with Bookings (via Booking_Seats)
Booking_Seats	id, booking_id, seat_id, booking_date, created_at	id	booking_id → Bookings(id), seat_id → Seats(id)	**Many-to-One** with Bookings, **Many-to-One** with Seats

Important Notes:

1. **Disjoint Relationship**: The Users entity has a **disjoint specialization** where users are categorized as either:
    - **Admin Users**: Have administrative privileges to manage events
    - **Normal Users**: Regular users who can book events

This means a user cannot be both an admin and a normal user simultaneously - they must be one or the other.

Detailed Relationship Analysis:

Specialization Relationships:

- **Users ↔ Admins/Normal Users**: **Disjoint Specialization** (ISA relationship)
    - A user can be either an Admin OR a Normal User, but not both

Binary Relationships:

1. **Admins ↔ Events**: **One-to-Many** (1:M)
    - One Admin can manage multiple Events
    - Each Event is managed by exactly one Admin
2. **Events ↔ Bookings**: **One-to-Many** (1:M)
    - One Event can have multiple Bookings
    - Each Booking is for exactly one Event
3. **Events ↔ Seats**: **One-to-Many** (1:M)
    - One Event can have multiple Seats
    - Each Seat belongs to exactly one Event
4. **Bookings ↔ Seats**: **Many-to-Many** (M:N)
    - One Booking can reserve multiple Seats
    - One Seat can be part of multiple Bookings (over time, but only one active booking)
    - Resolved through **Booking_Seats** junction table

Ternary Relationships:

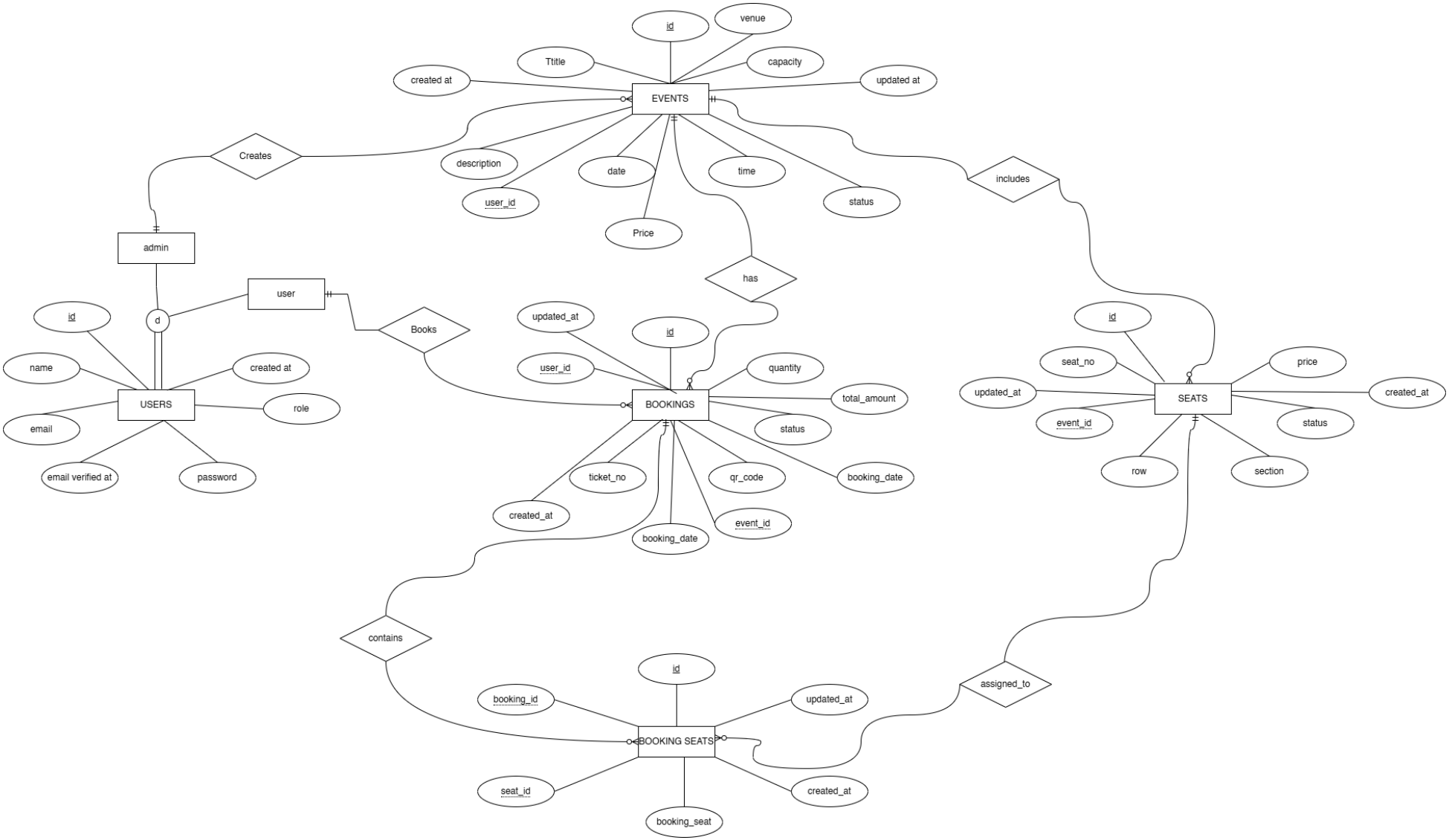
- **Booking_Seats** acts as a **junction entity** resolving the Many-to-Many relationship between Bookings and Seats

Cardinality Summary:

- **Admin : Events** = 1:M
- **Event : Bookings** = 1:M
- **Event : Seats** = 1:M
- **Booking : Seats** = M:N (via Booking_Seats)

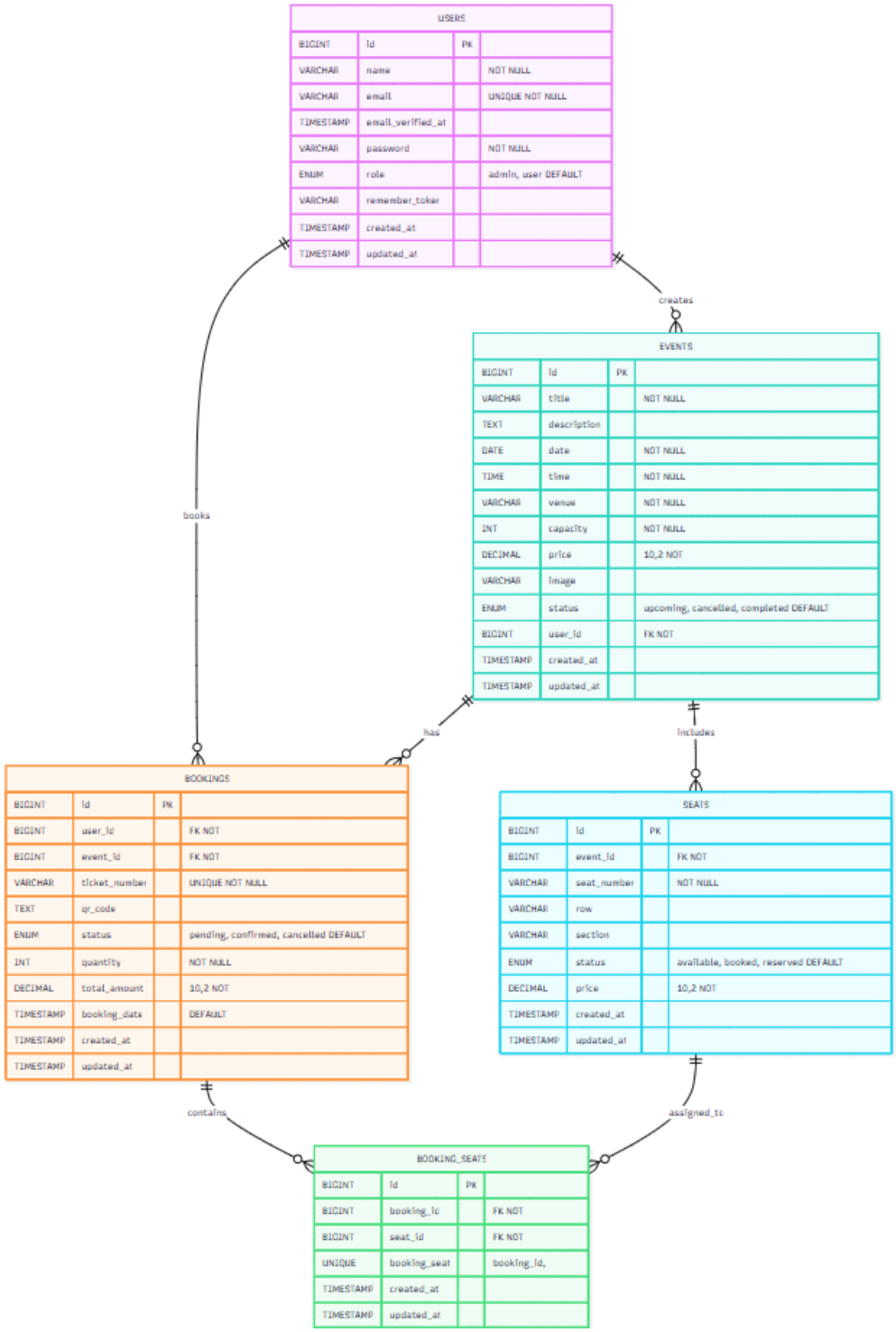
Finalized Conceptual Schema

Entity Relationship Diagram (ERD):



Enhanced ERD:

EERD includes additional attributes and relationships such as event ownership and booking status.



Business Rules:

- An admin can create multiple events.
- Each event is created and managed by one admin only.
- A customer can book a seat for an event without creating a user account.
- Each booking must be linked to one event.
- Each event must have a valid title, date, time, venue, capacity, and price before being listed.
- Events must have a status of either active, inactive, or cancelled. Only active events can be booked.

- The system must not allow seat capacity to be negative.
- Once all seats are booked or reserved, no new bookings can be made.
- Each seat is unique per event, identified by seat number.
- Seats must be marked as available, booked, or reserved. Booked seats cannot be double-booked.
- Each booking must generate a unique ticket number and QR code.
- The total amount for a booking is calculated based on the number of selected seats and seat price.
- Admins can edit or delete any event they created.
- When an event is deleted, all associated seats and bookings are also deleted.
- The system stores customer details such as name, email, and phone for every booking.
- A booking must include at least one seat.
- Only bookings with pending or confirmed status are considered active. Cancelled bookings release their seats.
- A user with role 'admin' can manage events. A user with role 'user' can only book events.
- Seats can have custom prices. If not set, the default event price is used.

References

1. Lecture Slides (Lecture 2 & Lecture 3) - Database Design
2. draw.io - for drawing ERD and EERD diagrams (https://draw.io)
3. ChatGPT by OpenAI - for code generation and documentation structure (https://chat.openai.com)
4. W3Schools - for syntax reference (https://www.w3schools.com/sql/)
5. MySQL Official Docs - https://dev.mysql.com/doc/