**Milestone 3**



**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**

**SQL Database Tables and Queries**

**Users Table**

CREATE TABLE users (

id BIGINT UNSIGNED AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(255) NOT NULL,

email VARCHAR(255) UNIQUE NOT NULL,

email\_verified\_at TIMESTAMP NULL,

password VARCHAR(255) NOT NULL,

role ENUM('user', 'admin') DEFAULT 'user',

remember\_token VARCHAR(100) NULL,

created\_at TIMESTAMP NULL,

updated\_at TIMESTAMP NULL

);

**Purpose**: Manages user authentication and authorization

**Key Features**:

* Unique email constraint
* Role-based access control
* Email verification support
* Remember token for persistent login
* Timestamps for audit trail

**Events Table**

CREATE TABLE events (

id BIGINT UNSIGNED AUTO\_INCREMENT PRIMARY KEY,

title VARCHAR(255) NOT NULL,

description TEXT NOT NULL,

date DATE NOT NULL,

time TIME NOT NULL,

venue VARCHAR(255) NOT NULL,

capacity INT NOT NULL,

price DECIMAL(8,2) NOT NULL,

image VARCHAR(255) NULL,

status ENUM('active', 'inactive', 'cancelled') DEFAULT 'active',

user\_id BIGINT UNSIGNED NOT NULL,

created\_at TIMESTAMP NULL,

updated\_at TIMESTAMP NULL,

FOREIGN KEY (user\_id) REFERENCES users(id) ON DELETE CASCADE

);

**Purpose**: Stores comprehensive event information

**Key Features**:

* Event ownership through user\_id
* Status management for event lifecycle
* Image support for event promotion
* Decimal precision for pricing
* Cascading delete for data integrity

**Seats Table**

CREATE TABLE seats (

id BIGINT UNSIGNED AUTO\_INCREMENT PRIMARY KEY,

event\_id BIGINT UNSIGNED NOT NULL,

seat\_number VARCHAR(255) NOT NULL,

row VARCHAR(255) NOT NULL,

section VARCHAR(255) DEFAULT 'main',

status ENUM('available', 'booked', 'reserved') DEFAULT 'available',

price DECIMAL(8,2) NULL,

created\_at TIMESTAMP NULL,

updated\_at TIMESTAMP NULL,

UNIQUE KEY unique\_seat (event\_id, seat\_number),

FOREIGN KEY (event\_id) REFERENCES events(id) ON DELETE CASCADE

);

**Purpose**: Manages individual seat allocation and pricing

**Key Features**:

* Unique seat identification per event
* Section-based organization
* Individual seat pricing capability
* Real-time status tracking
* Referential integrity with events

**Bookings Table**

CREATE TABLE bookings (

id BIGINT UNSIGNED AUTO\_INCREMENT PRIMARY KEY,

user\_id BIGINT UNSIGNED NOT NULL,

event\_id BIGINT UNSIGNED NOT NULL,

ticket\_number VARCHAR(255) UNIQUE NOT NULL,

qr\_code TEXT NOT NULL,

status ENUM('pending', 'confirmed', 'cancelled') DEFAULT 'pending',

quantity INT DEFAULT 1,

selected\_seats JSON NULL,

total\_amount DECIMAL(10,2) DEFAULT 0.00,

booking\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

created\_at TIMESTAMP NULL,

updated\_at TIMESTAMP NULL,

FOREIGN KEY (user\_id) REFERENCES users(id) ON DELETE CASCADE,

FOREIGN KEY (event\_id) REFERENCES events(id) ON DELETE CASCADE

);

**Purpose**: Manages booking transactions and ticket generation

**Key Features**:

* Unique ticket number generation
* QR code storage for digital tickets
* JSON storage for flexible seat selection
* Comprehensive booking status tracking

**METADATA**

**Table: users**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** | **Key Type** |
| id | BIGINT UNSIGNED | AUTO\_INCREMENT, NOT NULL | PRIMARY KEY |
| name | VARCHAR(255) | NOT NULL | – |
| email | VARCHAR(255) | NOT NULL, UNIQUE | UNIQUE |
| email\_verified\_at | TIMESTAMP | NULL | – |
| password | VARCHAR(255) | NOT NULL | – |
| role | ENUM('user','admin') | DEFAULT 'user' | – |
| remember\_token | VARCHAR(100) | NULL | – |
| created\_at | TIMESTAMP | NULL | – |
| updated\_at | TIMESTAMP | NULL | – |

**Table: events**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** | **Key Type** |
| id | BIGINT UNSIGNED | AUTO\_INCREMENT, NOT NULL | PRIMARY KEY |
| title | VARCHAR(255) | NOT NULL | – |
| description | TEXT | NOT NULL | – |
| date | DATE | NOT NULL | – |
| time | TIME | NOT NULL | – |
| venue | VARCHAR(255) | NOT NULL | – |
| capacity | INT | NOT NULL | – |
| price | DECIMAL(8,2) | NOT NULL | – |
| image | VARCHAR(255) | NULL | – |
| status | ENUM('active','inactive','cancelled') | DEFAULT 'active' | – |
| user\_id | BIGINT UNSIGNED | NOT NULL | FOREIGN KEY → users(id) |
| created\_at | TIMESTAMP | NULL | – |
| updated\_at | TIMESTAMP | NULL | – |

**Table: seats**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** | **Key Type** |
| id | BIGINT UNSIGNED | AUTO\_INCREMENT, NOT NULL | PRIMARY KEY |
| event\_id | BIGINT UNSIGNED | NOT NULL | FOREIGN KEY → events(id) |
| seat\_number | VARCHAR(255) | NOT NULL | UNIQUE (event\_id, seat\_number) |
| row | VARCHAR(255) | NOT NULL | – |
| section | VARCHAR(255) | DEFAULT 'main' | – |
| status | ENUM('available','booked','reserved') | DEFAULT 'available' | – |
| price | DECIMAL(8,2) | NULL | – |
| created\_at | TIMESTAMP | NULL | – |
| updated\_at | TIMESTAMP | NULL | – |

**Table: bookings**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** | **Key Type** |
| id | BIGINT UNSIGNED | AUTO\_INCREMENT, NOT NULL | PRIMARY KEY |
| user\_id | BIGINT UNSIGNED | NOT NULL | FOREIGN KEY → users(id) |
| event\_id | BIGINT UNSIGNED | NOT NULL | FOREIGN KEY → events(id) |
| ticket\_number | VARCHAR(255) | NOT NULL, UNIQUE | UNIQUE |
| qr\_code | TEXT | NOT NULL | – |
| status | ENUM('pending','confirmed','cancelled') | DEFAULT 'pending' | – |
| quantity | INT | DEFAULT 1 | – |
| selected\_seats | JSON | NULL | – |
| total\_amount | DECIMAL(10,2) | DEFAULT 0.00 | – |
| booking\_date | TIMESTAMP | DEFAULT CURRENT\_TIMESTAMP | – |
| created\_at | TIMESTAMP | NULL | – |
| updated\_at | TIMESTAMP | NULL | – |

**✅ KEYS Summary**

**Primary Keys**

* users.id
* events.id
* seats.id
* bookings.id

**Foreign Keys**

* events.user\_id → users.id
* seats.event\_id → events.id
* bookings.user\_id → users.id
* bookings.event\_id → events.id

**Unique Keys**

* users.email
* bookings.ticket\_number
* seats.event\_id + seat\_number (Composite Unique Constraint)

**DATABASE QUERIES**

**Query 1: Select all events**

SELECT \* FROM events;

**Query 2: Register a new user**

INSERT INTO users (name, email, password)

VALUES ('Hassan Zaib', 'hassan@example.com', 'hashed\_password');

**Query 3: Book an event**

INSERT INTO bookings (user\_id, event\_id, ticket\_number, qr\_code, status, quantity, total\_amount)

VALUES (2, 5, 'TKT10005', 'QR\_DATA\_HERE', 'pending', 2, 2000.00);

**Query 4: Admin login verification**

SELECT \* FROM users

WHERE email = 'admin@example.com'

AND password = 'hashed\_password'

AND role = 'admin';

**Query 5: Fetch all available seats for a specific event**

SELECT \* FROM seats

WHERE event\_id = 5

AND status = 'available';

**Query 6: Update seat status after booking**

UPDATE seats

SET status = 'booked'

WHERE id IN (10, 11, 12);

**Query 7: Get all bookings for a specific user**

SELECT \* FROM bookings

WHERE user\_id = 2;

**Query 8: Cancel a booking**

UPDATE bookings

SET status = 'cancelled'

WHERE id = 7;

**Query 9: View event details along with admin info**

SELECT events.\*, users.name AS admin\_name, users.email AS admin\_email

FROM events

JOIN users ON events.user\_id = users.id;

**Query 10: Delete an event (admin-only operation)**

DELETE FROM events

WHERE id = 4

AND user\_id = 1;