**Password for DB - mpteam@5**

**When we sign in to the website:**

1. **As admin – we can see all the events as well as the attendees of the events**
2. **As organizer – we can see all the events created by me and their specific list of people registered for that event. No privilege to see other organizers events**
3. **As attendee – we can see all the events and their details that we have registered for**

**When a person first sign up into the website, below details need to be provided.**

**USER\_DETAILS**

user\_id

user\_name

email

password

role [admin/organizer/attendee]

user\_image (optional)

user\_reated\_at

**There are few genere that are provided through this event management website and they are fixed. Any event organized should be with in this list.**

**GENERE\_DETAILS**

genre\_id

genre\_name

{Concert, Comedy show, Product launch, Tech conference, Hackathon/Bootcamp, Yoga & Meditation, Food, festival, Art exhibition}

**When an organizer is creating an event he needs to provide this information**

**EVENT\_DETAILS**

event\_id

event\_name

event\_image

description

date

location

event\_created\_at

no\_of\_tickets

general\_tickets

vip\_tickets

early\_bird\_tickets

general\_ticket\_price

vip\_ticket\_price

early\_bird\_ticket\_price

genere

**When an attendee is registering for an event then he/she need to provide the below details**

**ATTENDEE\_DETAILS**

attendee\_id

user\_id

event\_id

name

email

phone

event\_registered\_at

gender

dob

ticket\_type

ticket\_price

**Attendee can give a review on the event during/post the event**

**REVIEW\_DETAILS**

review\_id

event\_id

attendee\_id

rating

comment

review\_created\_at

**Attendee can give feedback post the event**

**FEEDBACK\_DETAILS**

feedback\_id

event\_id

attendee\_id

feeback\_text

feedback\_created\_at

**Payment details of a ticket for an event is stored in the payment\_details\_table.**

**PAYMENT\_DETAILS**

payment\_id

event\_id

attendee\_id

amount\_paid

payment\_status (success/failed)

-- USERS TABLE (Stores user accounts)

CREATE TABLE user\_details (

user\_id SERIAL PRIMARY KEY,

user\_name VARCHAR(100) NOT NULL,

email VARCHAR(150) UNIQUE NOT NULL,

password TEXT NOT NULL, -- Hashed password

role VARCHAR(20) CHECK (role IN ('admin', 'organizer', 'attendee')) NOT NULL,

user\_image TEXT DEFAULT NULL, -- Optional

user\_created\_at TIMESTAMP DEFAULT NOW()

);

-- GENRE TABLE (Fixed list of event genres)

CREATE TABLE genre\_details (

genre\_id SERIAL PRIMARY KEY,

genre\_name VARCHAR(100) UNIQUE NOT NULL

);

-- Insert fixed genres

INSERT INTO genre\_details (genre\_name) VALUES

('Concert'), ('Comedy show'), ('Product launch'), ('Tech conference'),

('Hackathon/Bootcamp'), ('Yoga & Meditation'), ('Food festival'), ('Art exhibition');

-- EVENTS TABLE (Stores event information)

CREATE TABLE event\_details (

event\_id SERIAL PRIMARY KEY,

event\_name VARCHAR(255) NOT NULL,

event\_image TEXT DEFAULT NULL,

description TEXT NOT NULL,

date TIMESTAMP NOT NULL,

location VARCHAR(255) NOT NULL,

event\_created\_at TIMESTAMP DEFAULT NOW(),

no\_of\_tickets INT NOT NULL,

general\_tickets INT NOT NULL,

vip\_tickets INT NOT NULL,

early\_bird\_tickets INT NOT NULL,

general\_ticket\_price DECIMAL(10,2) NOT NULL,

vip\_ticket\_price DECIMAL(10,2) NOT NULL,

early\_bird\_ticket\_price DECIMAL(10,2) NOT NULL,

genre\_id INTEGER REFERENCES genre\_details(genre\_id) ON DELETE RESTRICT,

organizer\_id INTEGER REFERENCES user\_details(user\_id) ON DELETE CASCADE

);

-- ATTENDEE TABLE (Stores event registrations)

CREATE TABLE attendee\_details (

attendee\_id SERIAL PRIMARY KEY,

user\_id INTEGER REFERENCES user\_details(user\_id) ON DELETE CASCADE,

event\_id INTEGER REFERENCES event\_details(event\_id) ON DELETE CASCADE,

name VARCHAR(100) NOT NULL,

email VARCHAR(150) NOT NULL,

phone VARCHAR(15),

event\_registered\_at TIMESTAMP DEFAULT NOW(),

gender VARCHAR(10) CHECK (gender IN ('Male', 'Female', 'Other')),

dob DATE NOT NULL

);

-- REVIEWS TABLE (Stores event reviews)

CREATE TABLE review\_details (

review\_id SERIAL PRIMARY KEY,

event\_id INTEGER REFERENCES event\_details(event\_id) ON DELETE CASCADE,

attendee\_id INTEGER REFERENCES attendee\_details(attendee\_id) ON DELETE CASCADE,

rating INTEGER CHECK (rating BETWEEN 1 AND 5) NOT NULL,

comment TEXT DEFAULT NULL,

review\_created\_at TIMESTAMP DEFAULT NOW()

);

-- FEEDBACK TABLE (Stores attendee feedback)

CREATE TABLE feedback\_details (

feedback\_id SERIAL PRIMARY KEY,

event\_id INTEGER REFERENCES event\_details(event\_id) ON DELETE CASCADE,

attendee\_id INTEGER REFERENCES attendee\_details(attendee\_id) ON DELETE CASCADE,

feedback\_text TEXT NOT NULL,

feedback\_created\_at TIMESTAMP DEFAULT NOW()

);

-- PAYMENT TABLE (Tracks ticket payments)

CREATE TABLE payment\_details (

payment\_id SERIAL PRIMARY KEY,

event\_id INTEGER REFERENCES event\_details(event\_id) ON DELETE CASCADE,

attendee\_id INTEGER REFERENCES attendee\_details(attendee\_id) ON DELETE CASCADE,

amount\_paid DECIMAL(10,2) NOT NULL,

payment\_status VARCHAR(10) CHECK (payment\_status IN ('success', 'failed')) NOT NULL

);