SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

SCHOOL OF COMPUTING

DEPARTMENT OF DATA SCIENCE AND BUSINESS SYSTEMS

18CSC303J DATABASE MANAGEMENT SYSTEM

MINI PROJECT REPORT

Event Management System

Semester: VI

Team Members

Ritish Chaudhary RA1911003011011
CHITTURU ANISH KASYAP RA1911003010999
R. VARUN REDDY RA1911003011003

Content Page

INTRODUCTION

Event management is an application aiming to manage creation and development of events, festivals and conferences.

Event management involves having a user friendly application that lets a user plan and host an event. He can list any kind of event, and list its organizer, sponsors, tags, venue, payment plans, etc. A user on the other end can also participate in said events and get information about the pricing, like how much extra he is paying from the average participant, what's the minimum and maximum price paid by any participant for the event.

MOTIVATION

The event management system is the heart of your operational platforms.

It sits in the middle and orchestrates processes from your various events to guarantee intelligent fulfilment and improved customer experiences.

Typically, lots of manual hours are spent in order to keep record and generate reports of events, and it's never centralized, or at a position where it's accessible to everyone.

Our application aims to solve all these dilemmas by having a database to hold all the records and using SQL to perform queries and apply aggregate functions, all while having a user-friendly interface where everyone from anywhere in the world can access it. The data is centralized in a way which is available to all the event managers.

Modules Description

MODULE 1: Proposed ER Diagram

COMPLETED DURING: 5th April to 7th April

MODULE 2: Schema Conversion

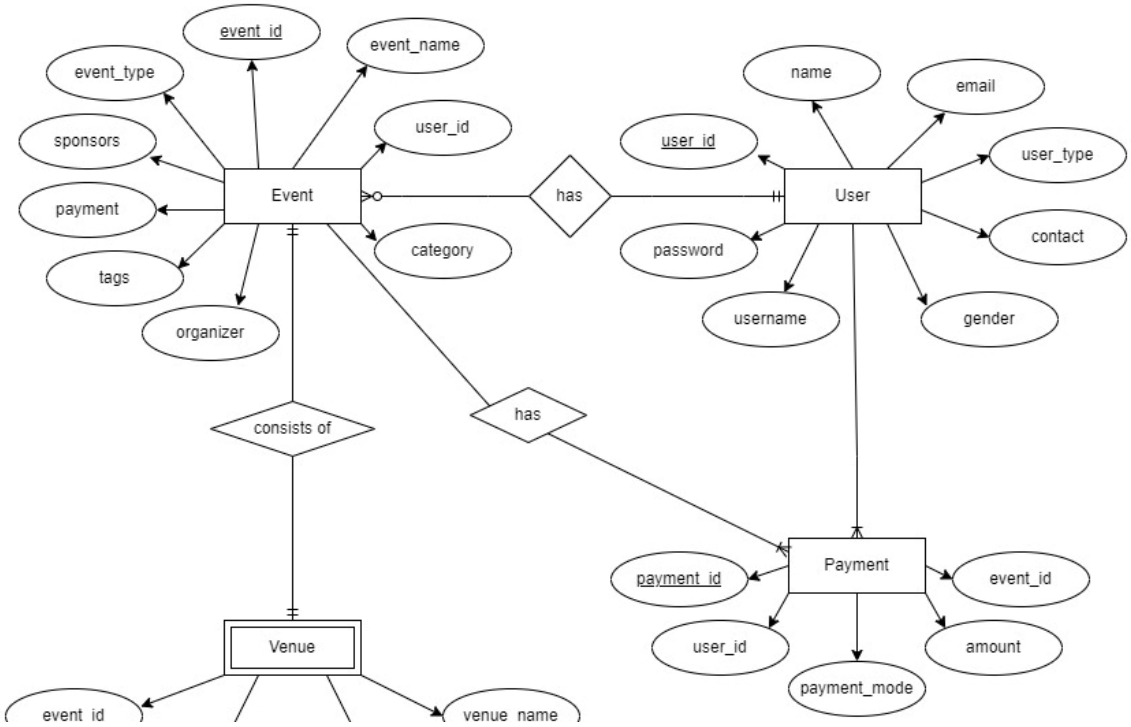
COMPLETED DURING: 7th April to 10th April

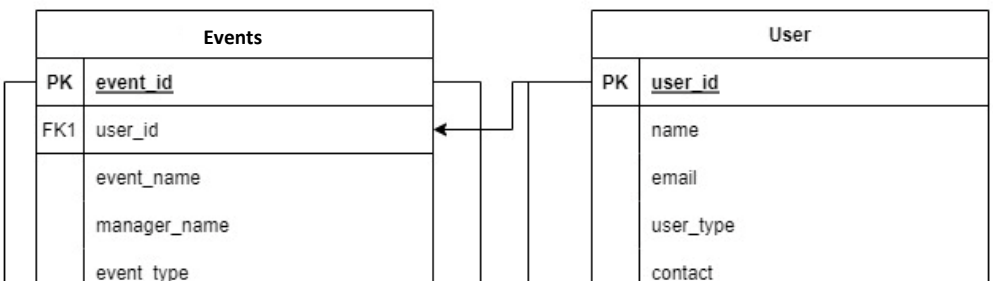
MODULE 3: Code Implementation

COMPLETED DURING: 10th April to 17th April

Proposed ER Diagram

An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system. ER Diagrams are most often used to design or debug relational databases in the fields of software engineering, business information systems, education and research. Also known as ERDs or ER Models, they use a defined set of symbols such as rectangles, diamonds, ovals and connecting lines to depict the interconnectedness of entities, relationships and their attributes. They mirror grammatical structure, with entities as nouns and relationships as verbs.





Schema Conversion

add_event(event_id,user_id,event_name,manager_name,event_type,sponsors,tags,organizer,payment,timedate,venue_name)

edit_event(event_id,user_id,event_name,manager_name,event_type,sponsors,tags,organizer,payment,timedate,venue_name)

delete_event(event_id)

total_participants(event_id)

total_payment_collected(event_id)

add_user(user_id, name, user_type, contact, gender, username, password)

get_user_details(user_id)

get_event_details(event_id)

price_info(event_id)

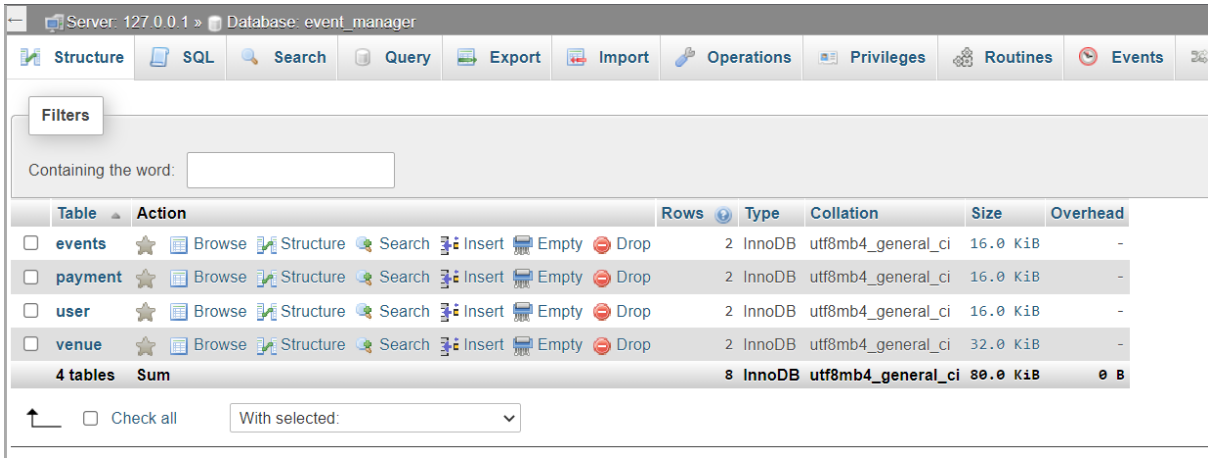
buy_ticket(event_id,user_id ,payment_mode, amount)

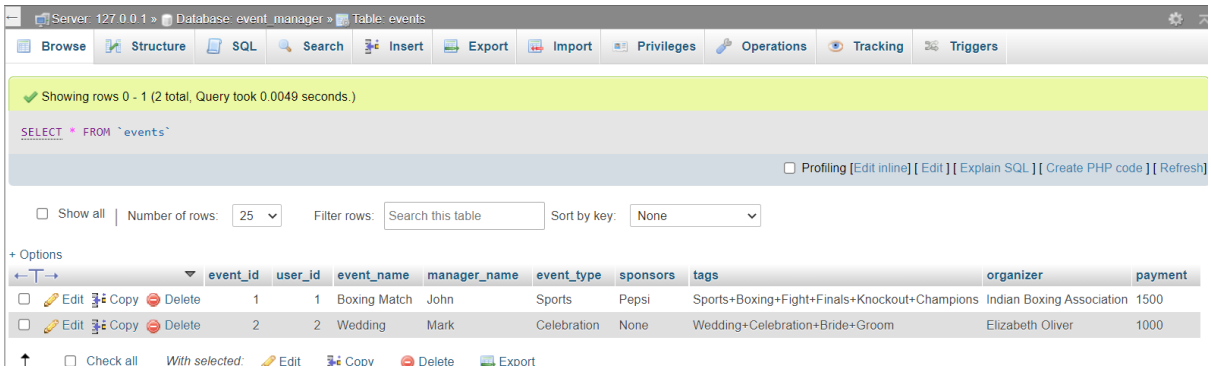
Implementation Requirements

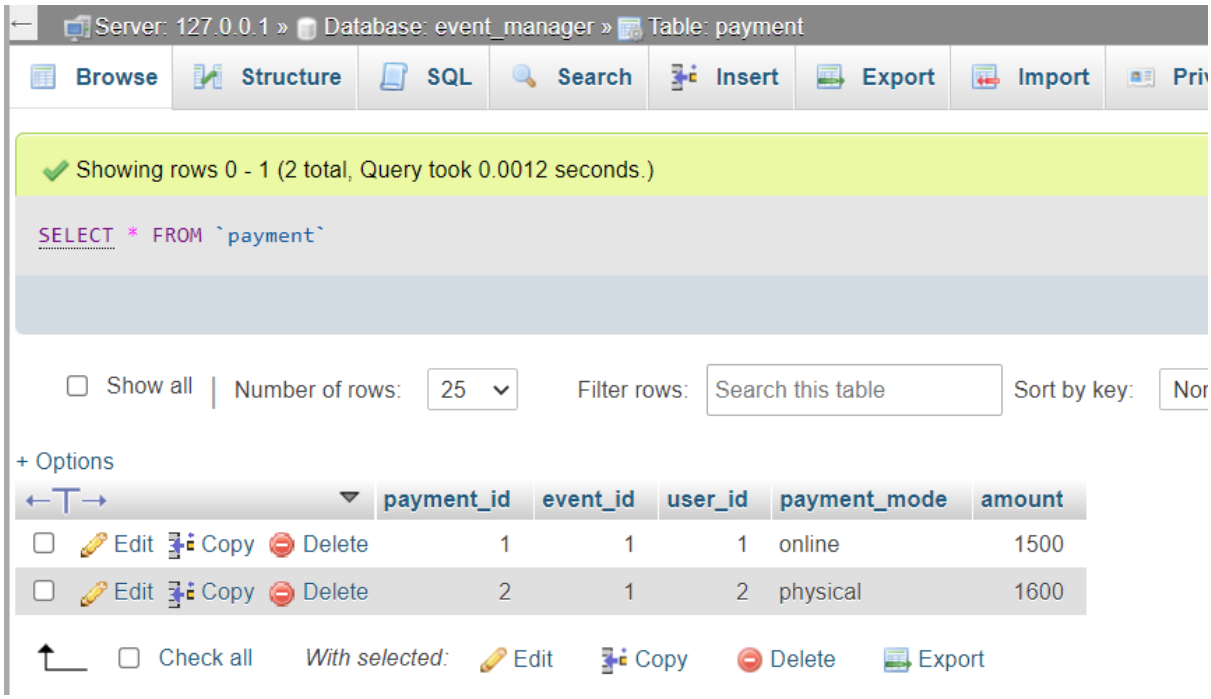
- Python
- mysql connector
- xampp
- command line

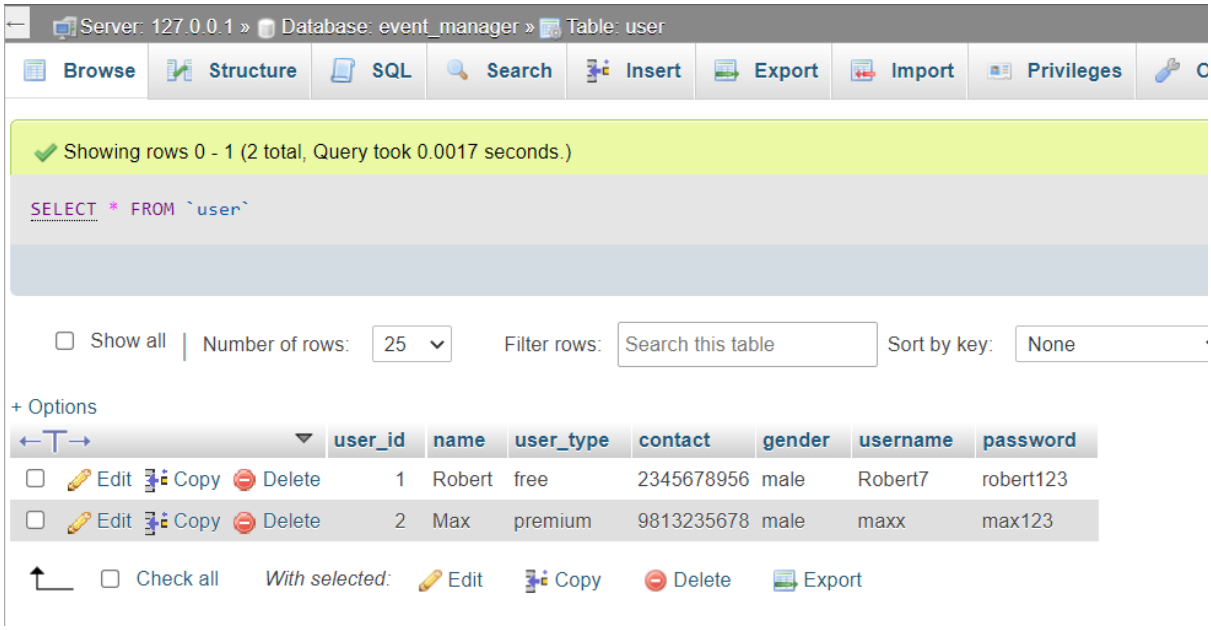
Output Screenshots:

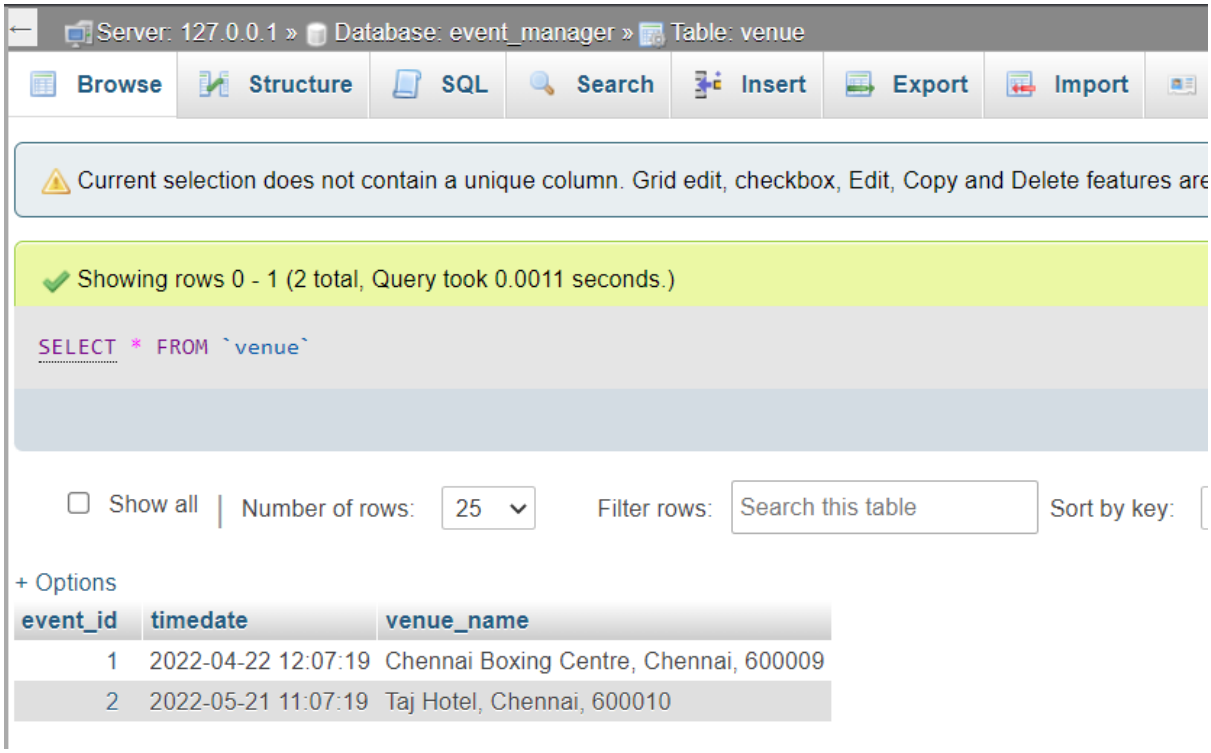
```
Total Participants:
Number of Participants in event 1 is :  2


Total Payment Collected:
Total Amount Collected for event 1 is :  3100


User Details:

Name:  Robert
Contact Number:  2345678956
Gender :  male


Event Details:

Event Name :  Boxing Match
Manager Name :  John
Event Type:  Sports
Event Sponsor(s):  Pepsi
Organizer:  Indian Boxing Association
Tags:  Sports, Boxing, Fight, Finals, Knockout, Champions
Venue Name:  Chennai Boxing Centre, Chennai, 600009
Date:  2022-04-22
Time :  12:07:19


Price Information:

Avg Price for Event 'Boxing Match' is :  1550.0000
Minimum Price for Event 'Boxing Match' is :  1500
Maximum Price for Event 'Boxing Match' is :  1600
```


Server: 127.0.0.1 » Database: event_manager
Structure SQL Search Query Export Import Operations Privileges Routines Events
Filters
Containing the word:
Table Action Rows Type Collation Size Overhead
events Browse Structure Search Insert Empty Drop 2 InnoDB utf8mb4_general_ci 16.0 KiB -
payment Browse Structure Search Insert Empty Drop 2 InnoDB utf8mb4_general_ci 16.0 KiB -
user Browse Structure Search Insert Empty Drop 2 InnoDB utf8mb4_general_ci 16.0 KiB -
venue Browse Structure Search Insert Empty Drop 2 InnoDB utf8mb4_general_ci 32.0 KiB -
4 tables Sum 8 InnoDB utf8mb4_general_ci 80.0 KiB 0 B
Check all With selected:


Server: 127.0.0.1 » Database: event_manager » Table: events
Browse Structure SQL Search Insert Export Import Privileges Operations Tracking Triggers
Showing rows 0 - 1 (2 total, Query took 0.0049 seconds.)
SELECT * FROM `events`
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None
+ Options
event_id user_id event_name manager_name event_type sponsors tags organizer payment
Edit Copy Delete 1 1 Boxing Match John Sports Pepsi Sports+Boxing+Fight+Finals+Knockout+Champions Indian Boxing Association 1500
Edit Copy Delete 2 2 Wedding Mark Celebration None Wedding+Celebration+Bride+Groom Elizabeth Oliver 1000
Check all With selected: Edit Copy Delete Export


Server: 127.0.0.1 » Database: event_manager » Table: payment
Browse Structure SQL Search Insert Export Import Pri
Showing rows 0 - 1 (2 total, Query took 0.0012 seconds.)
SELECT * FROM `payment`
Show all | Number of rows: 25 Filter rows: Search this table Sort by key: Nor
+ Options
payment_id event_id user_id payment_mode amount
Edit Copy Delete 1 1 1 online 1500
Edit Copy Delete 2 1 2 physical 1600
Check all With selected: Edit Copy Delete Export


Server: 127.0.0.1 » Database: event_manager » Table: user
Browse
Structure
SQL
Search
Insert
Export
Import
Privileges
Showing rows 0 - 1 (2 total, Query took 0.0017 seconds.)
SELECT * FROM `user`
Show all
Number of rows: 25
Filter rows: Search this table
Sort by key: None
+ Options
user_id name user_type contact gender username password
Edit Copy Delete 1 Robert free 2345678956 male Robert7 robert123
Edit Copy Delete 2 Max premium 9813235678 male maxx max123
Check all With selected: Edit Copy Delete Export


Server: 127.0.0.1 » Database: event_manager » Table: venue
Browse
Structure
SQL
Search
Insert
Export
Import
Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are
Showing rows 0 - 1 (2 total, Query took 0.0011 seconds.)
SELECT * FROM `venue`
Show all
Number of rows: 25
Filter rows: Search this table
Sort by key:
+ Options
event_id timedate venue_name
1 2022-04-22 12:07:19 Chennai Boxing Centre, Chennai, 600009
2 2022-05-21 11:07:19 Taj Hotel, Chennai, 600010

APPENDIX -A

Conclusion:-

We have implemented all the schemas and done all the operations with it on Xampp, Python and the command line.

References:-

https://www.w3schools.com/python/python_mysql_getstarted.asp

https://www.mysql.com/products/connector/

https://dev.mysql.com/doc/connector-python/en/

https://stackoverflow.com/

Github Source Code link:-

https://github.com/Aryan7Mohan/Event_Manager

APPENDIX -B

Source code:-

```
import mysql.connector
from mysql.connector import errorcode
from datetime import date, datetime, timedelta

try:
  cnx = mysql.connector.connect(user='root',
                                database='event_manager')
except mysql.connector.Error as err:
  if err.errno == errorcode.ER_ACCESS_DENIED_ERROR:
    print("Something is wrong with your user name or password")
  elif err.errno == errorcode.ER_BAD_DB_ERROR:
    print("Database does not exist")
  else:
    print(err)
else:
  cnx.close()


def
add_event(event_id,user_id,event_name,manager_name,event_type,sponsor
s,tags,organizer,payment,timedate,venue_name):
    cnx = mysql.connector.connect(user='root', database='event_manager')
    cursor = cnx.cursor()


    #tomorrow = datetime.now().date() + timedelta(days=1)
    #event_id   user_id event_name  manager_name    event_type  sponsors
tags    organizer   payment
```

```
    add = ("INSERT INTO events "
                "(event_id,user_id ,event_name, manager_name, event_type,
sponsors,tags,organizer,payment) "
                "VALUES (%s,%s, %s, %s, %s, %s, %s, %s, %s)")

cursor.execute(add,(event_id,user_id,event_name,manager_name,event_typ
e,sponsors,tags,organizer,payment))
    cnx.commit()


    add2= ("INSERT INTO venue "
                "(event_id,timedate,venue_name) "
                "VALUES (%s,%s, %s)")


    cursor.execute(add2,(event_id,timedate,venue_name))
    cnx.commit()




    cursor.close()
    cnx.close()



#add_event(2,2,"Wedding","Mark","Celebration","None","Wedding+Celebratio
n+Bride+Groom","Elizabeth Oliver",1000,"2022-05-21 11:07:19","Taj Hotel,
Chennai, 600010")




def
edit_event(event_id,user_id,event_name,manager_name,event_type,sponsor
s,tags,organizer,payment,timedate,venue_name):
    cnx = mysql.connector.connect(user='root', database='event_manager')
```

```
    cursor = cnx.cursor()


    #tomorrow = datetime.now().date() + timedelta(days=1)
    #event_id   user_id event_name  manager_name    event_type  sponsors tags    organizer   payment


    edit = ("UPDATE events "
                "SET event_name = %s, payment=%s, manager_name = %s, event_type=%s, sponsors=%s, tags=%s, organizer=%s  "
                "WHERE user_id=%s and event_id=%s")



cursor.execute(edit,(event_name,payment,manager_name,event_type,sponsors,tags,organizer,user_id, event_id))
    cnx.commit()


    edit2= ("UPDATE venue "
                "SET event_id=%s,timedate=%s,venue_name=%s WHERE event_id=%s")


    cursor.execute(edit2,(event_id,timedate,venue_name,event_id))
    cnx.commit()


    cursor.close()
    cnx.close()



#edit_event(4,4,"shubs and aryans birthday","aryan mohan","21th birthday","none","birthday+friends only+celebration","aryan",100,"2022-04-22 12:07:19","not aryans ghar")



def delete_event(event_id):
```

```
    cnx = mysql.connector.connect(user='root', database='event_manager')
    cursor = cnx.cursor()

    #tomorrow = datetime.now().date() + timedelta(days=1)
    #event_id   user_id event_name  manager_name    event_type  sponsors tags    organizer   payment

    delete = (""" DELETE FROM venue WHERE event_id="""+str(event_id)+""" """)

    cursor.execute(delete)
    cnx.commit()

    delete2 = (""" DELETE FROM events WHERE event_id="""+str(event_id)+""" """)
    cursor.execute(delete2)
    cnx.commit()

    cursor.close()
    cnx.close()

#delete_event(1)



def total_participants(event_id):
    #from payments table, get total participants in that event
    print("\n\nTotal Participants:")
    cnx = mysql.connector.connect(user='root', database='event_manager')
    cursor = cnx.cursor()

    #tomorrow = datetime.now().date() + timedelta(days=1)
```

```
    #event_id   user_id event_name  manager_name    event_type  sponsors tags    organizer   payment


    count = (""" SELECT count(payment_id) from payment WHERE event_id="""+str(event_id)+""" """)


    cursor.execute(count)
    result=cursor.fetchall()
    print("Number of Participants in event "+str(event_id)+" is : ",result[0][0])
    cnx.commit()


    cursor.close()
    cnx.close()

total_participants(1)



def total_payment_collected(event_id):
    #add all payment
    print("\n\nTotal Payment Collected:")
    cnx = mysql.connector.connect(user='root', database='event_manager')
    cursor = cnx.cursor()


    #tomorrow = datetime.now().date() + timedelta(days=1)
    #event_id   user_id event_name  manager_name    event_type  sponsors tags    organizer   payment


    count = (""" SELECT sum(amount) from payment WHERE event_id="""+str(event_id)+""" """)


    cursor.execute(count)
    result=cursor.fetchall()
```

```
    print("Total Amount Collected for event "+str(event_id)+" is : ",result[0][0])
    cnx.commit()

    cursor.close()
    cnx.close()

total_payment_collected(1)



def add_user(user_id, name, user_type, contact, gender, username,
password):
  cnx = mysql.connector.connect(user='root', database='event_manager')
  cursor = cnx.cursor()

  create_user = ("INSERT INTO event_manager.user "
                 "(user_id, name, user_type, contact, gender, username, password) "
                 "VALUES (%s, %s,%s,%s,%s,%s,%s)")

  cursor.execute(create_user,(user_id, name, user_type, contact, gender,
username, password))
  cnx.commit()

  cursor.close()
  cnx.close()



#add_user(2, 'Max', 'premium', '9813235678', 'male', 'maxx', 'max123')



def get_user_details(user_id):
  print("\n\nUser Details:")
```

```
  cnx = mysql.connector.connect(user='root', database='event_manager')
  cursor = cnx.cursor(buffered=True)

  fetch_user = (f"select name, contact, gender from user where
user_id={str(user_id)}")

  cursor.execute(fetch_user)
  for row in cursor:
    print("\nName: ",row[0])
    print("Contact Number: ",row[1])
    print("Gender : ",row[2])


  cursor.close()
  cnx.close()

get_user_details(1)


def get_event_details(event_id):
  print("\n\nEvent Details:")
  cnx = mysql.connector.connect(user='root', database='event_manager')
  cursor = cnx.cursor(buffered=True)

  fetch_event = (f"select c.event_name, c.manager_name,
c.event_type,v.venue_name, v.timedate,c.sponsors, c.organizer,c.tags from
events as c join venue as v on v.event_id = c.event_id where
c.event_id={str(event_id)}")

  cursor.execute(fetch_event)
  result=cursor.fetchall()
```

```
  print("\nEvent Name : ",result[0][0])
  print("Manager Name : ",result[0][1])
  print("Event Type: ",result[0][2])
  print("Event Sponsor(s): ",result[0][5])
  print("Organizer: ",result[0][6])
  print("Tags: ",', '.join(result[0][7].split('+')))
  print("Venue Name: ",result[0][3])
  print("Date: ",result[0][4].date())
  print("Time : ",result[0][4].time())


  cursor.close()
  cnx.close()


get_event_details(1)


def price_info(event_id):
  print("\n\nPrice Information:")
  cnx = mysql.connector.connect(user='root', database='event_manager')
  cursor = cnx.cursor()


  #tomorrow = datetime.now().date() + timedelta(days=1)
  #event_id   user_id event_name  manager_name    event_type  sponsors
tags    organizer   payment


  price = (""" SELECT avg(amount),min(amount),max(amount) from payment
WHERE event_id="""+str(event_id)+""" """)


  cursor.execute(price)
```

```
  result=cursor.fetchall()

  event_name = (""" SELECT event_name from events WHERE
event_id="""+str(event_id)+""" """)
  cursor.execute(event_name)
  result2=cursor.fetchall()
  print("\nAvg Price for Event '"+str(result2[0][0])+"' is : ",result[0][0])
  print("Minimum Price for Event '"+str(result2[0][0])+"' is : ",result[0][1])
  print("Maximum Price for Event '"+str(result2[0][0])+"' is : ",result[0][2])
  cnx.commit()

  cursor.close()
  cnx.close()

price_info(1)



def buy_ticket(event_id,user_id ,payment_mode, amount):
  cnx = mysql.connector.connect(user='root', database='event_manager')
  cursor = cnx.cursor()

  #tomorrow = datetime.now().date() + timedelta(days=1)
  #event_id   user_id event_name  manager_name    event_type  sponsors
tags    organizer   payment

  buy = ("INSERT INTO payment "
              "(event_id,user_id ,payment_mode, amount) "
              "VALUES (%s, %s, %s, %s)")
  cursor.execute(buy,(event_id,user_id ,payment_mode, amount))
  cnx.commit()
```

```
  cursor.close()
  cnx.close()

#buy_ticket(1,2,'physical',1600)
```

GITHUB LINK:

https://github.com/shubhadityasingh/Event-Management-System