|  |  |  |
| --- | --- | --- |
|  | 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**

Chapter 1 : Introduction and Motivation [Purpose of the problem statement (societal benefit)

Chapter 2: Modules Description

Chapter 3 : Proposed ER Diagram

Chapter 4: Schema Conversion

Chapter 4: Implementation requirements

Chapter 5: Output Screenshots

Conclusion

References

Appendix A – GitHub Profile and Link for the Project

Appendix B – Source Code

**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

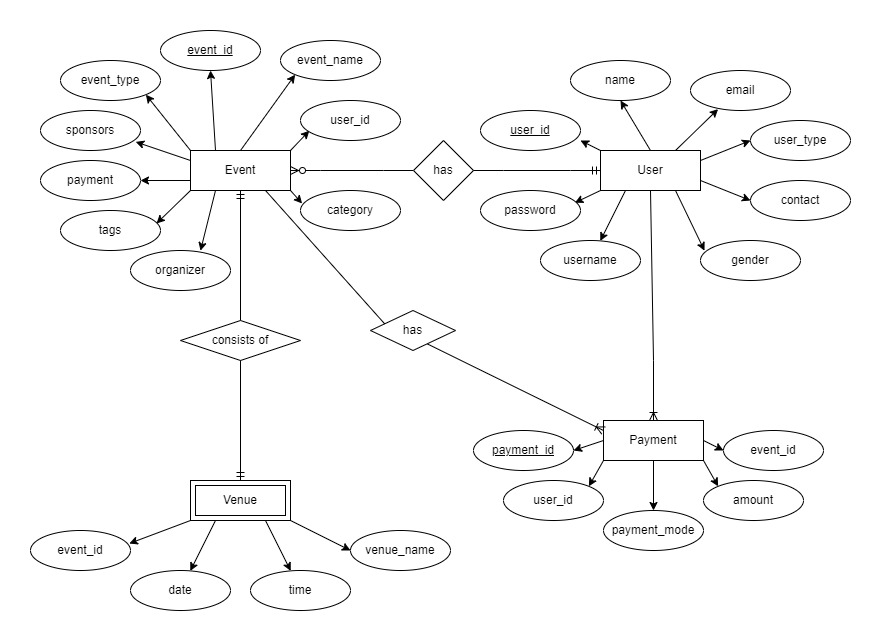
MODULE 2: Schema Conversion

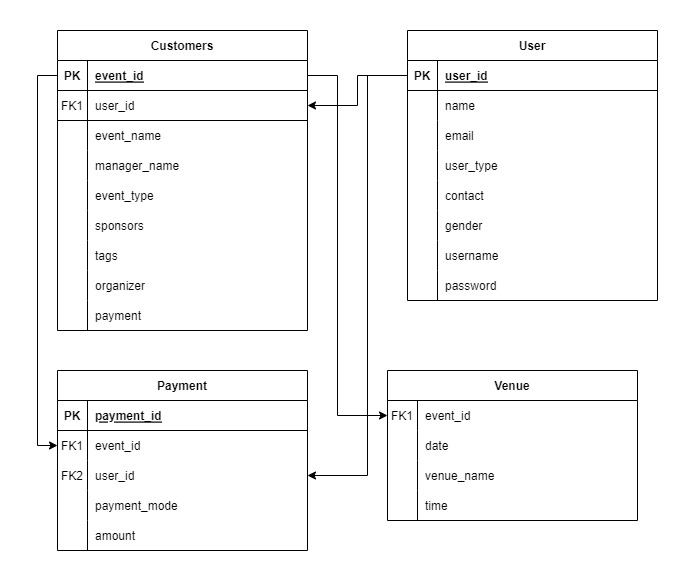
COMPLETED DURING: 7th April to 10th April

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.



**Events**

**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)

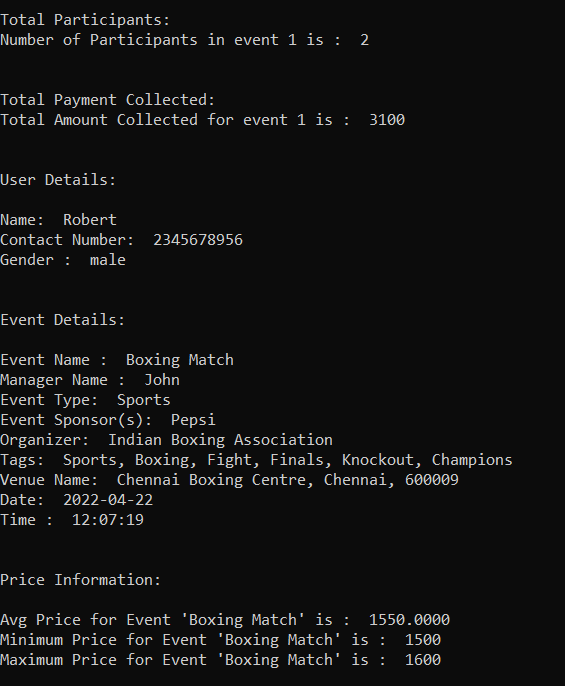
price\_info(event\_id)

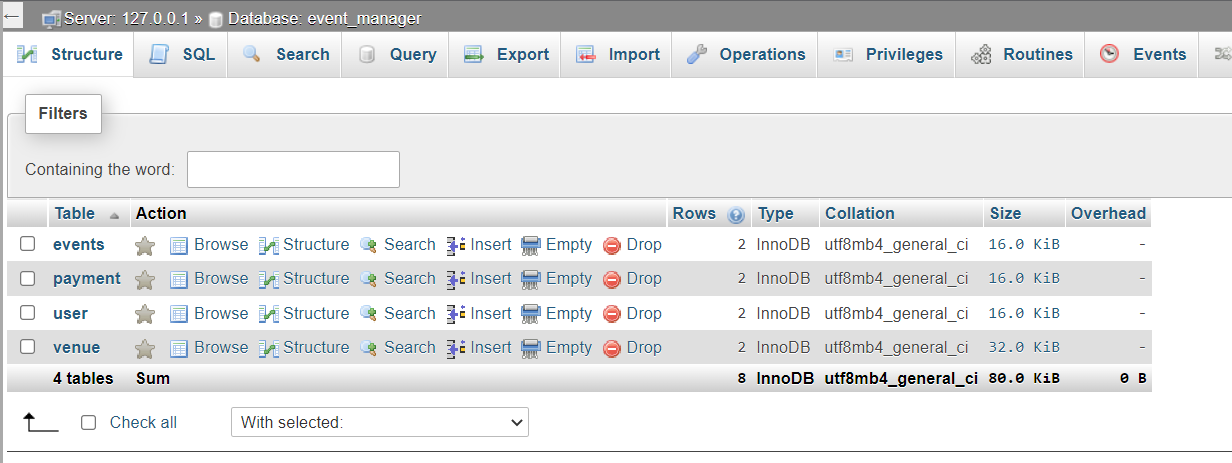
buy\_ticket(event\_id,user\_id ,payment\_mode, amount)

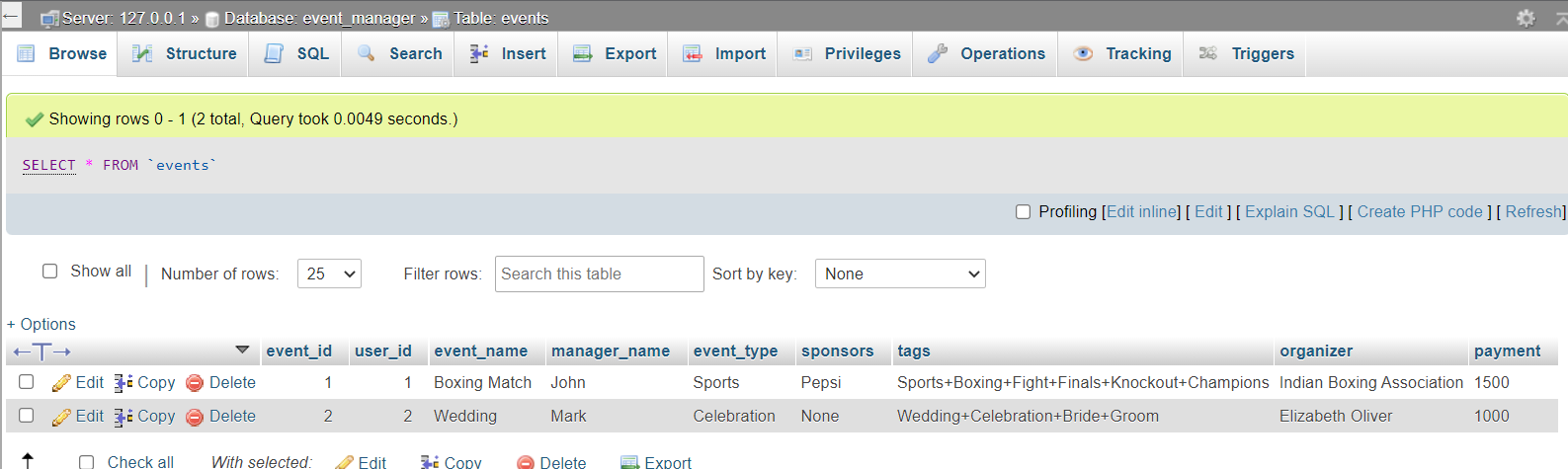
**Implementation Requirements**

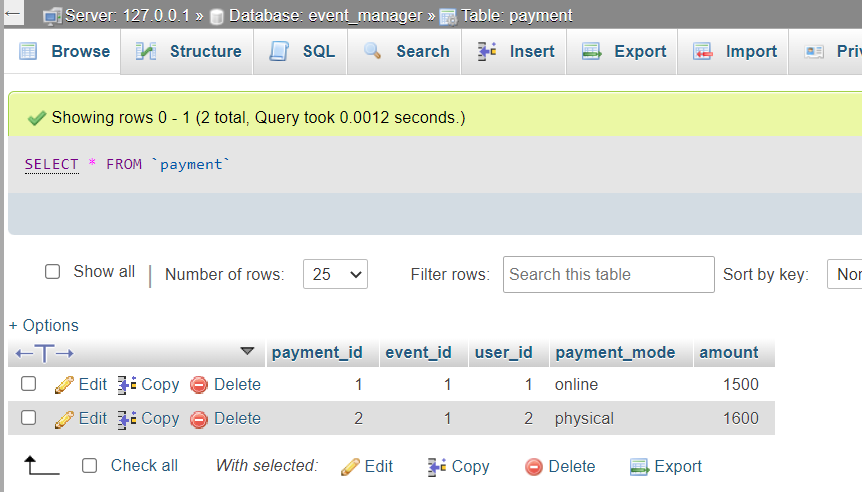
* Python
* mysql connector
* xampp
* command line

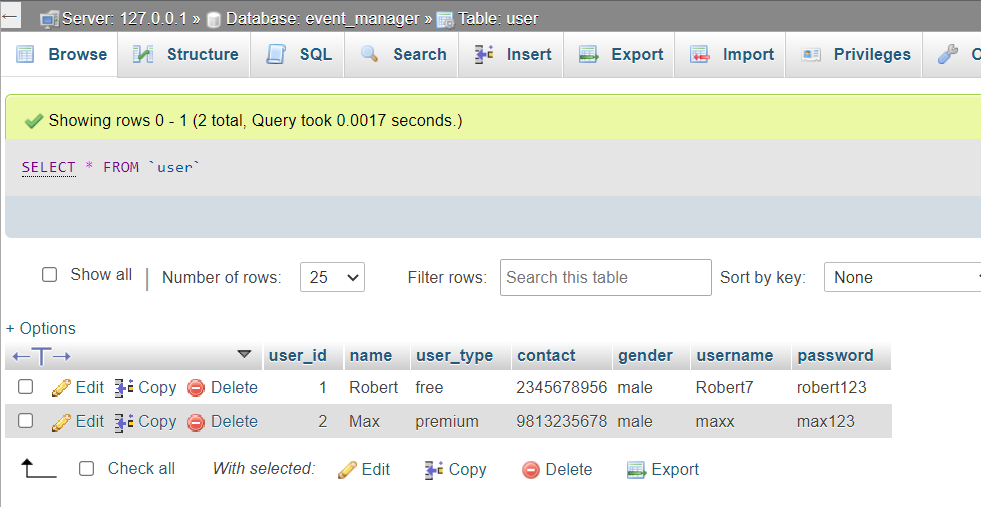
**Output Screenshots:**

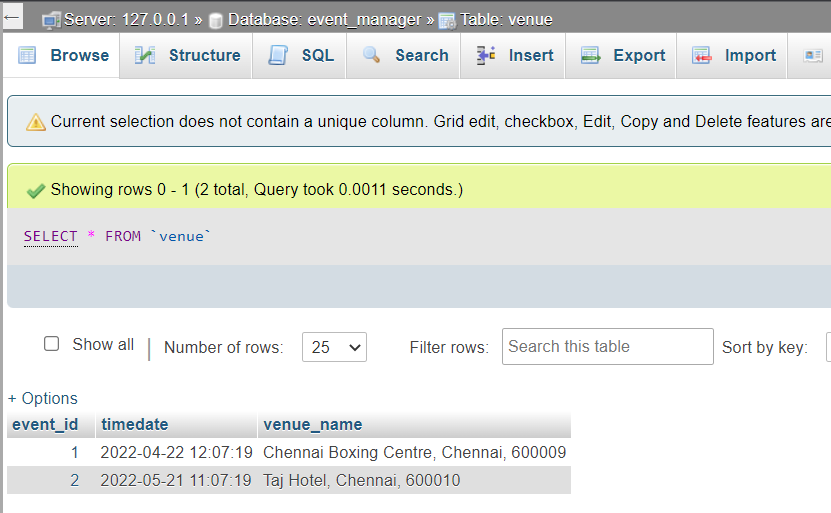
****

****

****

****

****

****

**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.w3schools.com/python/python_mysql_getstarted.asp)

[**https://www.mysql.com/products/connector/**](https://www.mysql.com/products/connector/)

[**https://dev.mysql.com/doc/connector-python/en/**](https://dev.mysql.com/doc/connector-python/en/)

[**https://stackoverflow.com/**](https://stackoverflow.com/)

**Github Source Code link:-**

[**https://github.com/Aryan7Mohan/Event\_Manager**](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,sponsors,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\_type,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+Celebration+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,sponsors,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