1. INTRODUCTION

Movies have a unique way of taking us on incredible adventures, making us laugh, cry, and sparking our imagination. They're not just about passing the time; they're windows into different worlds and a reflection of our thoughts and emotions at various stages of life.

Today, the world of entertainment, especially movies, is a thriving business, thus, I would like to present my project, the 'Movie Rent Library Management' software. This program offers an exciting opportunity for movie enthusiasts to dive into the world of cinema. It empowers customers to rent movies from the library with ease, eliminating the need for manual processes and reducing the workload of employees.

But that's not all; this program is more than just a rental platform. It's a personalized movie assistant. It automatically calculates due payments and adds fines for overdue rentals, ensuring a hassle-free experience for both customers and staff.

Moreover, it takes into account the unique preferences of each customer. By analyzing age and gender categories, it suggests movies that are tailored to individual tastes and moods. It's like having a friendly movie expert at your fingertips, ready to guide you through a vast collection of films.!

2. HARDWARE AND SOFTWARE REQUIREMENTS

2.1 HARDWARE REQUIREMENTS

I. INTEL OR AMD PROCESSOR WITH ATLEAST 1GHZ SPEED

II. RAM : 521 MB+

III. PEN DRIVE :(IF ANY REQUIRED)

IV. PRINTER : (IF ANY REQUIRED-(HARD COPY))

2.2 SOFTWARE REQUIREMENTS

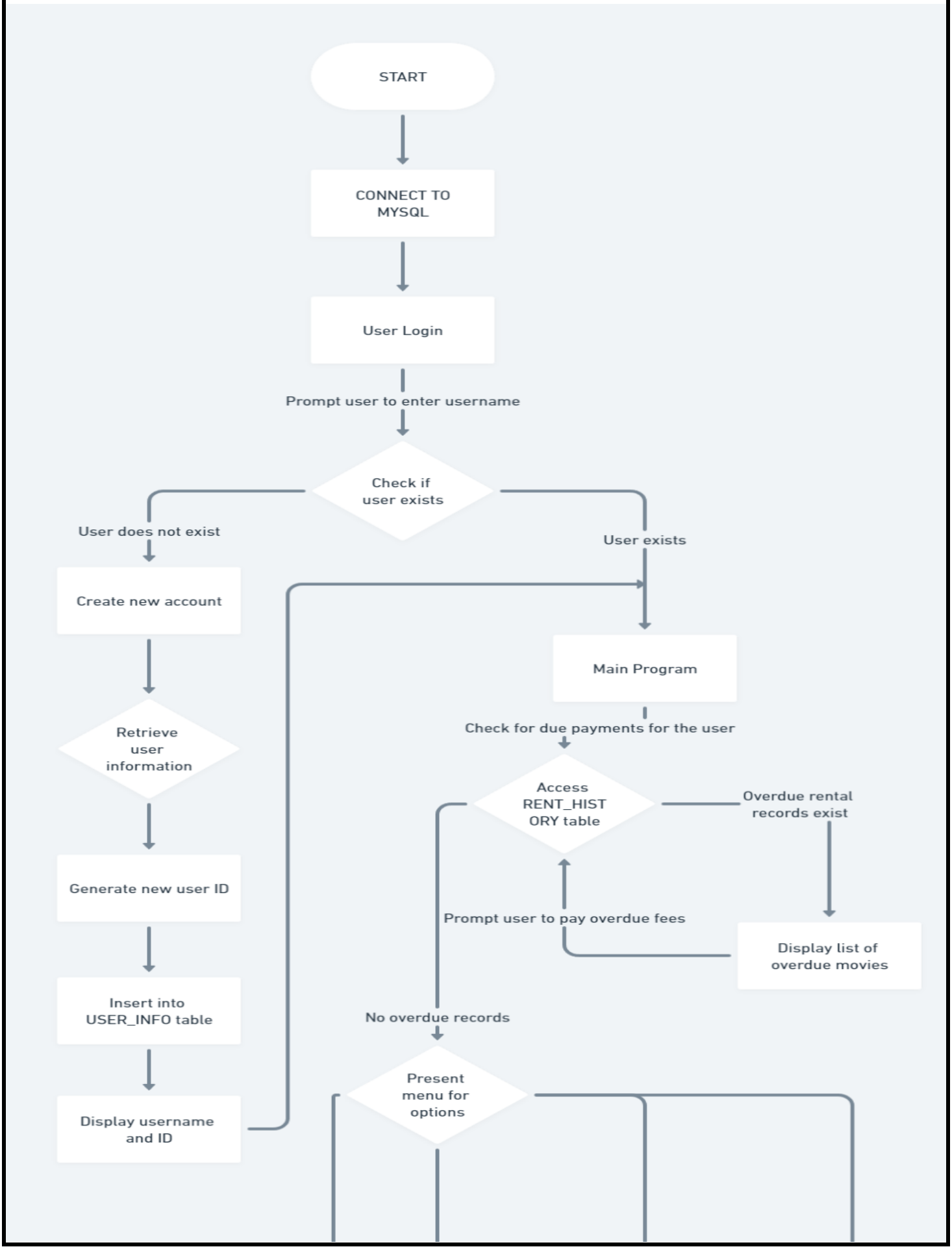
1. WINDOWS OS

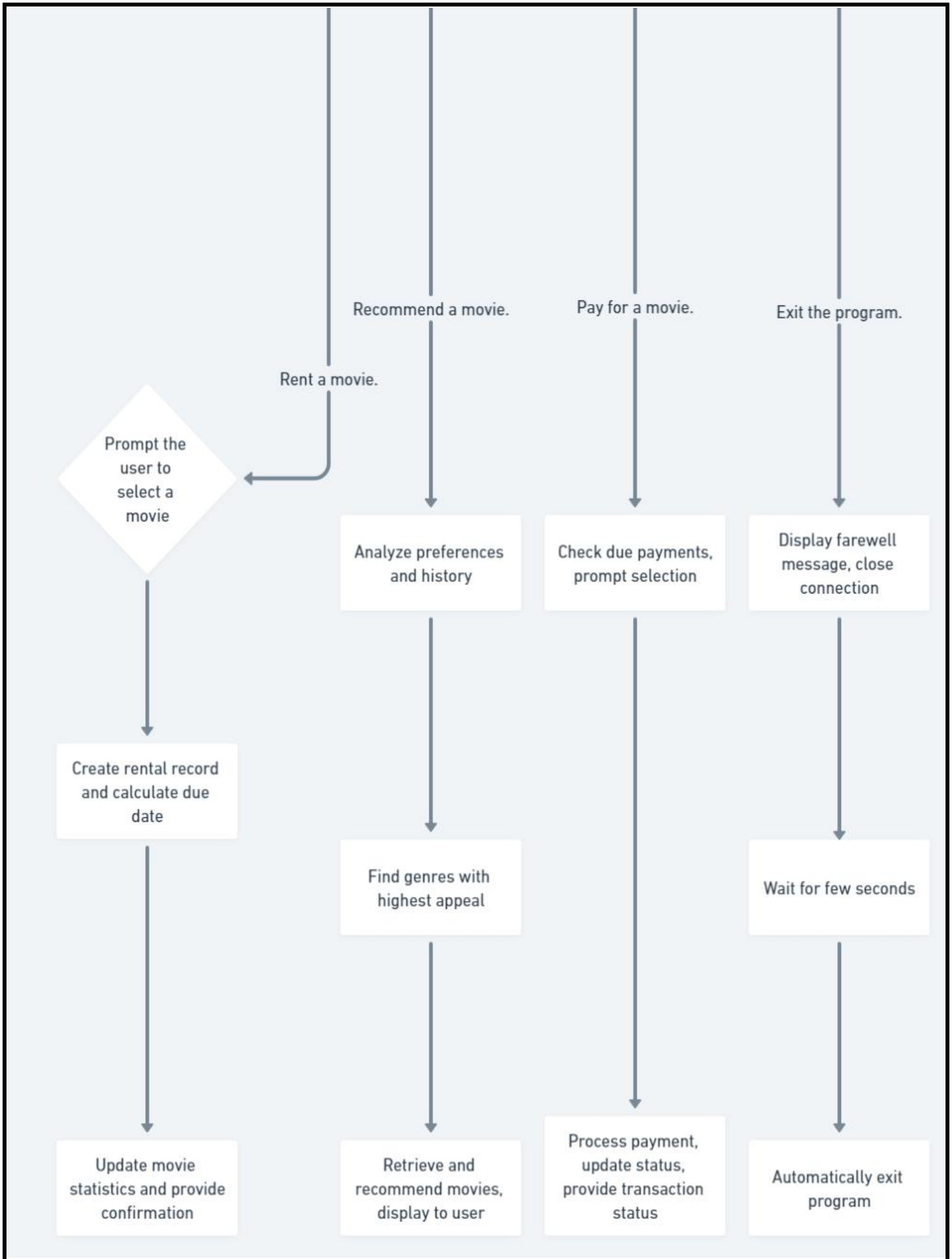
II. PYTHON 3.7

III. MYSQL CONNECTOR 5.0

IV. MICROSOFT WORD

3.FLOW OF CONTROL




Rent a movie.
Recommend a movie.
Pay for a movie.
Exit the program.
Prompt the user to select a movie
Analyze preferences and history
Check due payments, prompt selection
Display farewell message, close connection
Create rental record and calculate due date
Find genres with highest appeal
Wait for few seconds
Update movie statistics and provide confirmation
Retrieve and recommend movies, display to user
Process payment, update status, provide transaction status
Automatically exit program

4. PROJECT DESCRIPTION

Movie Rent Library Management is a python computerized application for maintaining, accessing, providing information movie rental platforms.

FUNCTIONS OF THE PROGRAM

4.1. CHECKDUE

CHECKS FOR EXPIRED DUES

4.2. RENT A MOVIE

i. SEARCHES THE MOVIE AND GIVE A LIST OF MOVIES WITH ITS CODE FROM GIVEN MOVIE NAME.
ii. AFTER ENTERING THE MOVIE CODE, ITS STORE THE CURRENT DATE ASWELL AS DUE DATE WHICH IS 60 DAY AFTER RENTED DATE I.E. CURRENT DATE.

4.3. RECOMMEND A MOVIE

i. BASED ON COMMON FEED
ii. BASED ON USERS FEED

4.4. PAY FOR THE MOVIE RENT

i. SEARCHES AND RETURNS THE LIST OF DUE MOVIES WITH THEIR CODE.
ii. IF THE DUE DATE IS OVER THEN ITS ADD UP FINE OF 10% OF MOVIE PRICE EACH DAY.
iii. AFTER COMPLETING TRANSACTION IT WILL CLEAR DUE FROM DATABASE.

4.5. EXIT

EXIT FROM THE PROGRAM

5. SOURCE CODE

```
##################################-IMPORTING MODULE AND PACKAGES-##############################
import mysql.connector as sql
import time


#########################################-CONNECTING TO MYSQL-
#################################
user='root'#{user_name}
password='root'#{user_password}
def CONNECT():

connection_object=sql.connect(host='localhost',user=user,passwd=password)
    return connection_object
connection_object=CONNECT()
if connection_object.is_connected():
    print('SUCCESSFULLY CONNECTED TO MYSQL..')
else:
    print('ERROR CONNECTING TO MYSQL, PLEASE TRY AGAIN')
    connection_object=CONNECT()
    if connection_object.is_connected():
        print('SUCCESSFULLY CONNECTED TO MYSQL..')
    else:
        print('ERROR CONNECTING TO MYSQL, PLEASE TRY LATER')
        time.sleep(20)
        exit()
```

```
cursor_object=connection_object.cursor()


##################################-
CREATING DATABASE AND TABLES-#####################

#CREATING DATABASES AND ACCESSING IT

cursor_object.execute('CREATE DATABASE IF NOT EXISTS MOVIES;')

cursor_object.execute('USE MOVIES;')



#CREATING TABLES:-

#TABLES:-

#1.MOVIES_INFO TABLE: CONTAINS INFORMATION ABOUT MOVIES
(i.e)MOVIE CODE, MOVIES NAME,YEAR RELEASED,

#  MOVIE GENERE,DURATION, ACTORS NAME, IMDB RATING.  ******

#2.MOVIES_ANALYSING TABLE: CONTAINS DATA OF MOVIES SUCH AS
WHICH AGE CATAGEROY PEOPLE WOULD

#  WATCH, WHETHER ITS MALE OR FEMALE.

#3.USER_INFO TABLE: CONTAINS INFORMATION OF USER SUCH AS
USER ID, NAME, AGE, GENDER,

#  FAVORITE ACTOR.

#4.USER_ANALYSING TABLE: CONTAINS STATSTICS OF USER ABOUT
HIS/HER PREFERENCES IN GENERE

#5.AGE_ANALYSING TABLE: CONTAINS STATSTICS OF MOVIES
GENERE ACCORDING TO AGE

#6.RENT_HISTORY TABLE: CONTAIN RENT HISTORY. ******


tempvar='CREATE TABLE IF NOT EXISTS MOVIE_INFO(MOVIE_CODE
INT PRIMARY KEY,MOVIE_NAME VARCHAR(60),\
```

```
YEAR_RELEASED YEAR(4),GENERE VARCHAR(50),ACTORS
VARCHAR(50),DURATION TIME,IMDB_RATING FLOAT, PRICE INT
DEFAULT 0);'

cursor_object.execute(tempvar)


tempvar='CREATE TABLE IF NOT EXISTS
MOVIE_ANALYSING(MOVIE_CODE INT  REFERENCES \

MOVIE_INFO(MOVIE_CODE),1_5 INT DEFAULT 0, 5_10 INT DEFAULT 0,
10_15 INT DEFAULT 0, 15_20\

 INT DEFAULT 0, 20_25 INT DEFAULT 0, 25_30 INT DEFAULT 0, 30_40
INT DEFAULT 0, 40_50 INT\

 DEFAULT 0, 50_70 INT DEFAULT 0,70PLUS INT DEFAULT 0,MALE INT
DEFAULT 0,FEMALE INT DEFAULT 0 ,NIL INT  );'

cursor_object.execute(tempvar)


tempvar='CREATE TABLE IF NOT EXISTS USER_INFO(USER_ID INT
PRIMARY KEY,USER_NAME VARCHAR(20),AGE INT,\

GENDER ENUM("MALE","FEMALE","NIL"),FAVORITE_ACTOR
VARCHAR(50));'

cursor_object.execute(tempvar)


tempvar='CREATE TABLE IF NOT EXISTS USER_ANALYSING(USER_ID
INT REFERENCES \

 USER_INFO(USER_ID),SCI_FI  INT DEFAULT 0,ACTION  INT DEFAULT
0, THRILLER  INT DEFAULT\

 0, HORROR  INT DEFAULT 0, ANIMATION  INT DEFAULT 0, FAMILY
INT DEFAULT 0, FANTASY  INT \

 DEFAULT 0, COMEDY  INT DEFAULT 0, DRAMA  INT DEFAULT 0,
FICTION  INT DEFAULT 0,\

 ROMANCE  INT DEFAULT 0, ADVENTURE  INT DEFAULT 0);'
```

```
cursor_object.execute(tempvar)


tempvar='CREATE TABLE IF NOT EXISTS AGE_ANALYSING(GENERE
VARCHAR(20) PRIMARY KEY,1_5\
 INT DEFAULT 0, 5_10 INT DEFAULT 0, 10_15 INT DEFAULT 0, 15_20
INT DEFAULT 0, 20_25 \
 INT DEFAULT 0, 25_30 INT DEFAULT 0, 30_40 INT DEFAULT 0, 40_50
INT DEFAULT 0 ,\
 50_70 INT DEFAULT 0 ,70PLUS INT DEFAULT 0 ,MALE INT DEFAULT 0
,FEMALE INT DEFAULT 0,NIL INT  );'
cursor_object.execute(tempvar)


tempvar='CREATE TABLE IF NOT EXISTS RENT_HISTORY(USER_ID INT
REFERENCES USER_INFO(USER_ID),USER_NAME\
 VARCHAR(20),MOVIE_CODE INT REFERENCES
MOVIE_INFO(MOVIE_CODE),\
MOVIE_NAME VARCHAR(60),RENTED_DATE DATE, DUE_DATE
DATE,PAYMENT_STATUS \
ENUM("DUE","PAYED"));'
cursor_object.execute(tempvar)


cursor_object.fetchall()
tempvar=('SELECT COUNT(*) FROM AGE_ANALYSING ;')
cursor_object.execute(tempvar)
if not int(cursor_object.fetchone()[0])==12:
    tempvar="INSERT INTO AGE_ANALYSING (GENERE)
VALUES('SCI_FI'),('ACTION'),('THRILLER'),('HORROR'),('ANIMATION'),('
FAMILY'),('FANTASY'),\
    ('COMEDY'),('DRAMA'),('FICTION'),('ROMANCE'),('ADVENTURE');"
```

```
    cursor_object.execute(tempvar)
    connection_object.commit()



##########################################################
#####-FUNCTIONS-
##########################################3
#1.checkdue(userid): CHECKS FOR PENDING DUES. +
#2.agegap(age): GROUPS AGE.
#3.pay(moviecode,userid):PAYS THE RENT.
#4.payment(userid):FIND AND PAY THE RENT OR DUE.
#5.movievote(moviecode,age,gender):COLLECTS THE INFORMATION
ABOUT MOVIE AND USER ON EACH PURCHASE.
#6.rent(moviecode,moviename,userid,username):CREATE A RENT
HISTORY.
#7.movieselect(userid):SELECTS THE MOVIE FOR PURCHASING.
++++++++++++++
#8.recommend(userid):RECOMMEND THE MOVIES ON COMMONFEED
AS WELL AS USERFEED.


def tableconnection():
    cursor_object.fetchall()
    tempvar=('SELECT COUNT(*) FROM MOVIE_INFO ;')
    cursor_object.execute(tempvar)
    movieinfo=int(cursor_object.fetchone()[0])
    cursor_object.fetchall()
    tempvar=('SELECT COUNT(*) FROM MOVIE_ANALYSING ;')
    cursor_object.execute(tempvar)
```

```
    movieanalyse=int(cursor_object.fetchone()[0])
    if not movieinfo==movieanalyse:
            for i in range(movieanalyse+1,movieinfo+1):
                tempvar=("INSERT INTO MOVIE_ANALYSING
(MOVIE_CODE) VALUES ({moviecode});")\
.format(moviecode=i)
                cursor_object.execute(tempvar)
                connection_object.commit()
    cursor_object.fetchall()
    tempvar=('SELECT COUNT(*) FROM USER_INFO ;')
    cursor_object.execute(tempvar)
    userinfo=int(cursor_object.fetchone()[0])
    cursor_object.fetchall()
    tempvar=('SELECT COUNT(*) FROM USER_ANALYSING ;')
    cursor_object.execute(tempvar)
    useranalyse=int(cursor_object.fetchone()[0])
    if not userinfo==useranalyse:
            for i in range(useranalyse+1,userinfo+1):
                tempvar=("INSERT INTO USER_ANALYSING (USER_ID)
VALUES ({userid});")\
.format(userid=i)
                cursor_object.execute(tempvar)
                connection_object.commit()




def checkdue(userid):
```

```
    cursor_object.fetchall()
    tempvar=('SELECT MOVIE_NAME,MOVIE_CODE FROM
RENT_HISTORY WHERE USER_ID={userid} AND\
 DUE_DATE < CURDATE() AND
PAYMENT_STATUS="DUE";').format(userid=userid)
    cursor_object.execute(tempvar)
    duemovie=cursor_object.fetchall()
    if len(duemovie)==0:
        return True
    else:
        print('YOUR RENT ON THE MOVIES:-',end="")
        for i in duemovie:
            print(i[0],'code=',i[1],end=',')
        print('HAVE NOT RETURNED AND YOUR DUE DATE HAD BEEN
OVER.PLEASE COMPLETE YOUR\
 DUE TO CLEAR ALL RESTRICTION')
        return False



def agegap(age):
    if age>=70:
        return '70PLUS'
    elif age>=50:
        return '50_70'
    elif age>=40:
        return '40_50'
    elif age>=30:
```

```
        return '30_40'
    elif age>=25:
        return '25_30'
    elif age>=20:
        return '20_25'
    elif age>=15:
        return '15_20'
    elif age>=10:
        return '10_15'
    elif age>=5:
        return '5_10'
    else:
        return '1_5'


def pay(moviecode,userid):
    tempvar=("SELECT TIMESTAMPDIFF(DAY , CURDATE(),DUE_DATE)
FROM RENT_HISTORY\
 WHERE USER_ID={userid} AND MOVIE_CODE={moviecode} AND
PAYMENT_STATUS='DUE';").\
format(userid=userid,moviecode=moviecode)
    cursor_object.execute(tempvar)
    datediff=int(cursor_object.fetchone()[0])
    cursor_object.fetchall()
    tempvar=("SELECT PRICE FROM MOVIE_INFO WHERE
MOVIE_CODE={moviecode};").\
            format(moviecode=moviecode)
    cursor_object.execute(tempvar)
```

```
    price=(cursor_object.fetchone())
    if price==(None,):
        price=[1000]
    price=int(price[0])
    cursor_object.fetchall()
    if datediff >=0:
        print('YOUR PRICE IS:',price)
        finalprice=price
    else:
        fine=-1*(5/100)*price*datediff
        finalprice=price+fine
        print('YOUR FINAL AMOUNT:',price,'+',fine,'(FINE)=',finalprice)
        print('YOU HAVE FINED BECAUSE YOUR DUE DATE IS OVER
BEFORE:',datediff*-1,'DAYS')
    temp=True
    while temp:
        try:
            var=int(input('ENTER 1 TO PROCCED TRANSACTION..'))
            temp=False
        except:
            print('INVALID INPUT, PLESE TRY AGAIN')
    if var==1:
        tempvar=("UPDATE RENT_HISTORY SET
PAYMENT_STATUS='PAYED' WHERE USER_ID={userid}\
 AND
MOVIE_CODE={moviecode};").format(userid=userid,moviecode=moviec
ode)
        cursor_object.execute(tempvar)
```

```
        connection_object.commit()
        print('YOUR TRANSACTION IS SUCCESSFUL')
    else:
        print('TRANSACTION FAILED.. PLEASE TRY AGAIN')


def payment(userid):
    tempvar=("SELECT MOVIE_NAME, MOVIE_CODE FROM
RENT_HISTORY WHERE USER_ID={userid}\
 AND PAYMENT_STATUS='DUE';").format(userid=userid)
    cursor_object.execute(tempvar)
    var=cursor_object.fetchall()
    if len(var)==0:
        print('YOU HAVE NO DUE PAYMENT')
        return None
    print('YOU HAVE DUE ON THE MOVIES:')
    moviecode1=[]
    for i in var:
        print(i[0],'MOVIE_CODE=',i[1])
        moviecode1.append(i[1])
    print('HAVE NOT PAYED YET')
    var=int(input('ENTER THE MOVIE CODE FOR PAYMENT...\
NOTE YOU CAN PAY ONLY ONE MOVIE AT A TIME'))
    if var not in moviecode1:
        print('INVALID MOVIE CODE, TRY AGAIN')
        return None
    pay(var,userid)
```

```
def movievote(moviecode,age,gender):
    tempvar=("UPDATE MOVIE_ANALYSING  SET {age}={age}+1
WHERE MOVIE_CODE={moviecode};").\
            format(moviecode=moviecode,age=age)
    cursor_object.execute(tempvar)
    connection_object.commit()
    tempvar=("UPDATE MOVIE_ANALYSING  SET {gender}={gender}+1
WHERE MOVIE_CODE={moviecode};").\
            format(moviecode=moviecode,gender=gender)
    cursor_object.execute(tempvar)
    connection_object.commit()
    tempvar=("SELECT GENERE FROM MOVIE_INFO WHERE
MOVIE_CODE={moviecode};").\
            format(moviecode=moviecode)
    cursor_object.execute(tempvar)
    genere=cursor_object.fetchone()[0].split(',')
    for i in genere:
        tempvar=("UPDATE AGE_ANALYSING  SET {age}={age}+1
WHERE GENERE='{genere}';").format(genere=i,age=age)
        cursor_object.execute(tempvar)
        connection_object.commit()
        tempvar=("UPDATE AGE_ANALYSING  SET {gender}={gender}+1
WHERE GENERE='{genere}';").format(genere=i,gender=gender)
        cursor_object.execute(tempvar)
        connection_object.commit()
        tempvar=("UPDATE USER_ANALYSING  SET
{genere}={genere}+1 WHERE
USER_ID={userid};").format(genere=i,userid=userid)
        cursor_object.execute(tempvar)
```

```
        connection_object.commit()


def rent(moviecode,moviename,userid,username):
    tempvar=("INSERT INTO
RENT_HISTORY(MOVIE_CODE,MOVIE_NAME,USER_NAME,USER_ID,DU
E_DATE,RENTED_DATE,PAYMENT_STATUS)\

VALUES({moviecode},'{moviename}','{username}',{userid},DATE_ADD(
CURDATE() ,INTERVAL 60 DAY) ,CURDATE(),'DUE' );")\

.format(userid=userid,moviecode=moviecode,username=username,mov
iename=moviename)
    cursor_object.execute(tempvar)
    connection_object.commit()
    tempvar='SELECT DATE_ADD(CURDATE() ,INTERVAL 60 DAY)'
    cursor_object.execute(tempvar)
    date=cursor_object.fetchone()[0]
    cursor_object.fetchall()
    print('MOVIE PURCHASED.. \n  NOTE: YOUR DUE DATE IS AFTER 60
DAYS FROM NOW ON..THAT IS',date)



def movieselect(userid):
    tempvar=("SELECT MOVIE_NAME,MOVIE_CODE FROM
MOVIE_INFO;")
    cursor_object.execute(tempvar)
    movie=cursor_object.fetchall()
    var=input('ENTER THE MOVIENAME')
    var=var.lower().split()
```

```
    x=0
    if x>=0:
        print('MATCHES FOUND:')
    for i in movie:
        i1=i[0].lower().split()+[str(i[1])]
        if any(x in var for x in i1):
            x=x+1
            print(i[0],'CODE=',i[1])
    if x==0:
        print('NONE , TRY AGAIN')
        return False
    tempvar=("SELECT AGE, GENDER, USER_NAME FROM USER_INFO
WHERE USER_ID={userid};").format(userid=userid)
    cursor_object.execute(tempvar)
    var=cursor_object.fetchone()
    cursor_object.fetchall()
    age=agegap(int(var[0]))
    gender=var[1]
    username=var[2]
    temp=True
    while temp:
        try:
            var=int(input('ENTER THE MOVIE CODE:'))
            temp=False
        except:
            print('INVALID INPUT, PLESE TRY AGAIN')
    temp=var
```

```
    tempvar=("SELECT MOVIE_NAME FROM MOVIE_INFO WHERE
MOVIE_CODE={moviecode};").format(moviecode=temp)
    cursor_object.execute(tempvar)
    var3=cursor_object.fetchone()
    cursor_object.fetchall()
    if var3==None:
            print('NO RESULT FOUND, TRY AGAIN')
            return False
    moviename=var3[0]
    moviecode=temp
    rent(moviecode,moviename,userid,username)
    movievote(moviecode,age,gender)
    print('HOPE YOU ENJOYED OUR SERVICE')
    return True


def recommend(userid):
    print('LET US ANALYSE FOR YOU')
    tempvar=("SELECT AGE, GENDER, USER_NAME FROM USER_INFO
WHERE USER_ID={userid};").format(userid=userid)
    cursor_object.execute(tempvar)
    var=cursor_object.fetchone()
    cursor_object.fetchall()
    age=agegap(int(var[0]))
    gender=var[1]
    username=var[2]
    tempvar=("SELECT GENERE FROM AGE_ANALYSING ORDER BY
{age} desc;").format(age=age)
```

```
    cursor_object.execute(tempvar)
    genere=cursor_object.fetchall()
    if genere==None :
        genere=[]
    if  len(genere)>=2:
        genere1=genere[0][0]
        genere2=genere[1][0]
        tempvar=("SELECT MOVIE_NAME,MOVIE_INFO.MOVIE_CODE
FROM MOVIE_INFO, MOVIE_ANALYSING WHERE
MOVIE_INFO.MOVIE_CODE=\
    MOVIE_ANALYSING.MOVIE_CODE AND (GENERE LIKE
'%{genere1}%' OR GENERE LIKE '%{genere2}%') ORDER BY\

({age}+{gender});").format(genere1=genere1,genere2=genere2,gende
r=gender,age=age)
        cursor_object.execute(tempvar)
        movies=cursor_object.fetchmany(5)
        cursor_object.fetchall()
        x=0
        print(('BASED ON COMMON FEED, MOST {gender}S AT YOUR AGE
WOULD LIKE TO WATCH {genere1} AND {genere2}\
 THUS I
RECOMMEND:').format(gender=gender,genere1=genere1,genere2=gen
ere2))
        for i in movies:
            x=x+1
            print(x,'.',i[0],'CODE=',i[1])
    else:
        print('INSUFFICENT DATA FOR COMMON FEED')
```

```
    print()
    tempvar=("SELECT * FROM USER_ANALYSING WHERE
USER_ID={userid}").format(userid=userid)
    cursor_object.execute(tempvar)
    var=['SCI_FI', 'ACTION', 'THRILLER', 'HORROR', 'ANIMATION',
'FAMILY', 'FANTASY', 'COMEDY']
    var1=(cursor_object.fetchone())
    if var1 == (None) :
        var1=[0]
    var1=list(var1)
    cursor_object.fetchall()
    var1.pop(0)
    var2=sorted(var1,reverse=True)
    var2=list(set(var2))
    if len(var2)>=2:
        genere1=var[var1.index(var2[0])]
        genere2=var[var1.index(var2[1])]
        tempvar=("SELECT MOVIE_NAME,MOVIE_INFO.MOVIE_CODE
FROM MOVIE_INFO, MOVIE_ANALYSING WHERE
MOVIE_INFO.MOVIE_CODE=\
    MOVIE_ANALYSING.MOVIE_CODE AND (GENERE LIKE
'%{genere1}%' OR GENERE LIKE '%{genere2}%') ORDER BY\

({age}+{gender});").format(genere1=genere1,genere2=genere2,gende
r=gender,age=age)
        cursor_object.execute(tempvar)
        movies=cursor_object.fetchmany(5)
        cursor_object.fetchall()
```

```
        x=0
        print(('BASED ON YOUR FEED, YOU WOULD LIKE TO WATCH
{genere1} AND {genere2}\
 THUS I RECOMMEND:').format(genere1=genere1,genere2=genere2))
        for i in movies:
            x=x+1
            print(x,'.',i[0],'CODE=',i[1])
    else:
        print('INSUFFICENT DATA FOR USER FEED')




##########################################################
#######-LOG IN-
##########################################################
############


def login():
    username=input('ENTER THE USER NAME:')
    tableconnection()
    tempvar=('SELECT USER_ID FROM USER_INFO WHERE
USER_NAME="{username}" ;').format(username=username)
    cursor_object.execute(tempvar)
    userexists=cursor_object.fetchone()


    if userexists==None:
        print('WELCOME IT SEEMS LIKE YOU ARE NEW TO OUR
SHOP..LETS CREATE AN ACCOUNT FOR YOU')
```

```
        temp=True
        while temp:
            try:
                age=int(input('YOUR AGE:'))
                temp=False
            except:
                print('INVALID INPUT, PLESE TRY AGAIN')
        gender=input('ARE YOU MALE OR FEMALE:').upper()
        if not(gender ==  'MALE' or gender == 'FEMALE'):
           gender='NIL'
        tempvar='SELECT COUNT(*)FROM USER_INFO;'
        cursor_object.execute(tempvar)
        userid=userexists=cursor_object.fetchone()[0]+1
        tempvar=('INSERT INTO USER_INFO
VALUES({userid},"{username}",{age},"{gender}",NULL);')\

.format(userid=userid,username=username,age=age,gender=gender.up
per())
        cursor_object.execute(tempvar)
        connection_object.commit()
        print(('YOUR USERNAME:{username} \n
USERID:{userid}').format(username=username,userid=userid))
    tempvar=('SELECT USER_ID FROM USER_INFO WHERE
USER_NAME="{username}" ;').format(username=username)
    cursor_object.fetchall()
    cursor_object.execute(tempvar)
    userid=cursor_object.fetchone()[0]
    temp=True
```

```
while temp:
    try:
        varuserid=int(input('ENTER YOUR ID:'))
        temp=False
    except:
        print('INVALID INPUT, PLESE TRY AGAIN')
if varuserid==userid:
    print('WELCOME',username.upper())
else:
    print('WRONG USER ID..TRY AGAIN')
    temp=True
    while temp:

        try:
            varuserid=int(input('ENTER YOUR ID:'))
            temp=False
        except:
            print('INVALID INPUT, PLESE TRY AGAIN')

    if varuserid==userid:
        print('WELCOME',username.upper())
    else:
        print('WRONG USER ID..TRY LATER')
        time.sleep(5)
        exit()
return userid
```

```
#username-username
#userid-userid
########################################-START-########################################

def start(userid):
    print('WHAT CAN I DO FOR YOU:')
    for i in range(10):
        print()

print('########################################################################################')
        print('1.RENT A MOVIE','2.RECOMMEND A MOVIE','3.PAY FOR THE MOVIE RENT','4.EXIT',sep='\n')
        print('ENTER THE CORRECT SNO FOR GIVEN OPTIONS')
        temp=True
        while temp:
            try:
                var=int(input('SNO:'))
                temp=False
            except:
                print('INVALID INPUT, PLESE TRY AGAIN')
        print()
        if var==1:
            for i in range(5):
```

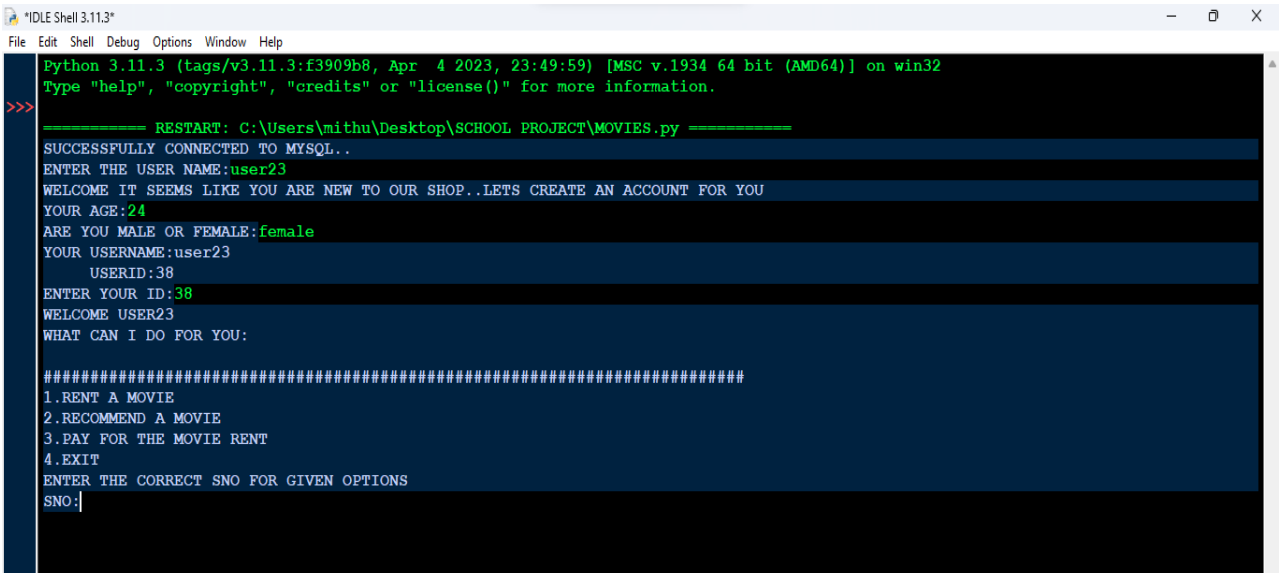
```
            x=movieselect(userid)
            if x==True:
                break
        else:
            print('PLEASE TRY LATER..')
    elif var==2:
        recommend(userid)
    elif var==3:
        payment(userid)
    elif var==4:
        print('HOPE WE MEET AGAIN THANK YOU')
        print('YOU WOULD AUTOMATICALLY EXIT IN 10 SEC')
        connection_object.close()
        time.sleep(10)
        exit()##################END
        break
        time.sleep(10)
    else:
       if i==4:
            print('CANT UNDERSTAND YOU TRY LATER..')
            exit()
       else:
            print('CANT UNDERSTAND YOU TRY AGAIN..')
    print()
    print()
```

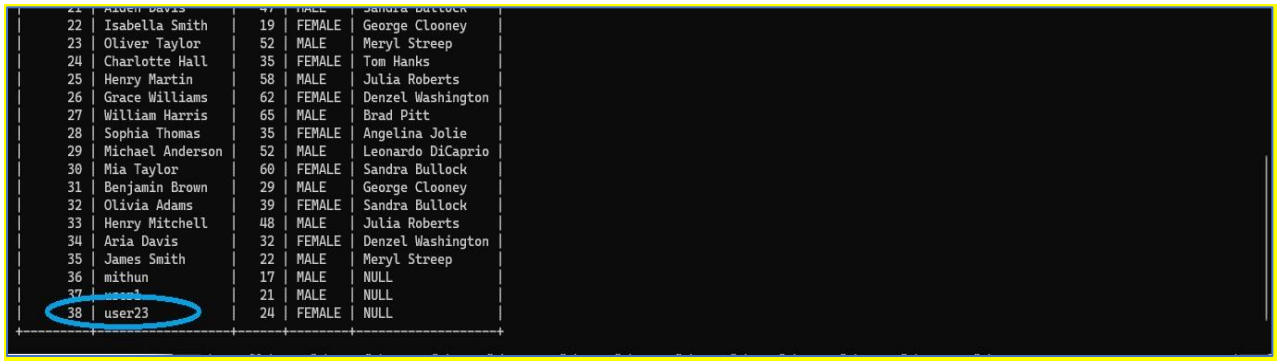
```
    print('YOU WOULD AUTOMATICALLY EXIT IN 10 SEC')
    connection_object.close()
    time.sleep(10)
    exit()#############END



temp1=True
while temp1:
    try:
        userid=login()
        temp1=False
    except:
        print("THERE IS AN ERROR CAUSED DURING LOGIN PLEASE TRY
AGAIN..")

tableconnection()
varv1=checkdue(userid)
while not varv1:
    print('YOU CANT ACCSESS ANY FUNCTION WITHOUT CLEARING
YOUR EXPIRY DUE..')
    payment(userid)
    varv1=checkdue(userid)



temp1=True
while temp1:
    try:
```

```
        start(userid)
        temp1=False
    except:
        print("THERE IS AN ERROR CAUSED ON OUR SIDE PLEASE TRY
AGAIN..")
```

6. OUTPUT SCREENSHOTS

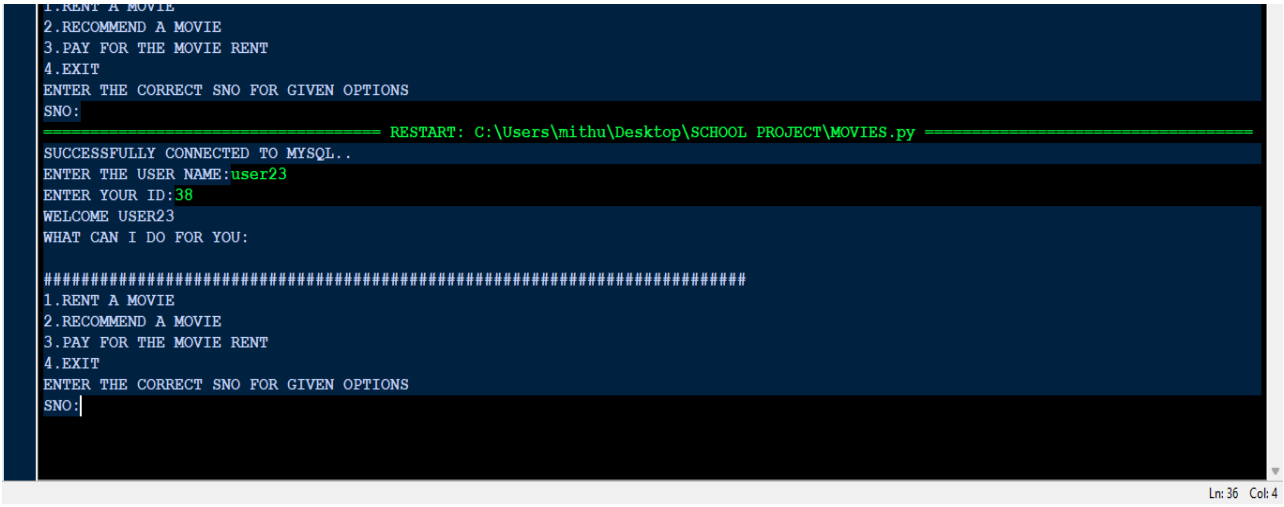
6.1 ADDING NEW USER





NOTE: NEW USER (user23) has been added in user_info table.

6.2 LOGGING IN USER



6.3 RENT A MOVIE

```
########################################################################
1.RENT A MOVIE
2.RECOMMEND A MOVIE
3.PAY FOR THE MOVIE RENT
4.EXIT
ENTER THE CORRECT SNO FOR GIVEN OPTIONS
SNO:1

ENTER THE MOVIENAMEstar wars
MATCHES FOUND:
Star Wars: Episode IV - A New Hope CODE= 13
Star Wars: Episode V - The Empire Strikes Back CODE= 26
Star Wars: Episode VI - Return of the Jedi CODE= 27
Star Wars: Episode I - The Phantom Menace CODE= 28
Star Wars: Episode II - Attack of the Clones CODE= 29
Star Wars: Episode III - Revenge of the Sith CODE= 30
Star Wars: The Force Awakens CODE= 31
Star Wars: The Last Jedi CODE= 32
Star Wars: The Rise of Skywalker CODE= 33
ENTER THE MOVIE CODE:13
MOVIE PURCHASED..
  NOTE: YOUR DUE DATE IS AFTER 60 DAYS FROM NOW ON..THAT IS 2024-01-08
HOPE YOU ENJOYED OUR SERVICE



########################################################################
1.RENT A MOVIE
```

Ln: 62 Col: 4

4	Interstellar	2014	SCI_FI,DRAMA	Matthew McConaughey, Anne Hathaway	02:49:00	8.6	4300
5	Avengers: Endgame	2019	ACTION,SCI_FI	Robert Downey Jr., Chris Evans	03:01:00	8.4	4200
6	Spider Man Across the Universe	2007	ANIMATION,ACTION	Evan Rachel Wood, Jim Sturgess	02:13:00	7.4	3700
7	How to Train Your Dragon	2010	ANIMATION,FAMILY	Jay Baruchel, Gerard Butler	01:38:00	8.1	4050
8	Annabelle	2014	HORROR,THRILLER	Annabelle Wallis, Ward Horton	01:39:00	5.4	2700
9	The Nun	2018	HORROR,THRILLER	Taissa Farmiga, Demián Bichir	01:36:00	5.3	2650
10	Free Guy	2021	ACTION,COMEDY	Ryan Reynolds, Jodie Comer	01:55:00	7.4	3700
11	Self/less	2015	SCI_FI,THRILLER	Ryan Reynolds, Ben Kingsley	01:57:00	6.5	3250
12	Inception	2010	ACTION,SCI_FI	Leonardo DiCaprio,Cillian Murphy	02:28:00	8.8	4400
13	Star Wars: Episode IV - A New Hope	1977	SCI_FI,FANTASY	k Hamill, Harrison Ford	02:01:00	8.6	4300
14	WALL-E	2008	ANIMATION,FAMILY	Ben Burtt, Elissa Knight	01:38:00	8.4	4200
15	It	2017	HORROR,THRILLER	Bill Skarsgård, Jaeden Martell	02:15:00	7.3	3650
16	Harry Potter and the Sorcerers Stone	2001	FANTASY,ADVENTURE	Daniel Radcliffe, Rupert Grint	02:32:00	7.6	3800
17	Captain America: The First Avenger	2011	ACTION,ADVENTURE	Chris Evans, Hugo Weaving	02:04:00	6.9	3450
18	Iron Man	2008	ACTION,SCI_FI	Robert Downey Jr., Gwyneth Paltrow	02:06:00	7.9	3950
19	Thor	2011	ACTION,FANTASY	Chris Hemsworth, Natalie Portman	01:55:00	7	3500
20	Mission: Impossible	1996	ACTION,THRILLER	Tom Cruise, Jon Voight	01:50:00	7.1	3550
21	Pirates of the Caribbean: The Curse of the Black Pearl	2003	ACTION,ADVENTURE	Johnny Depp, Geoffrey Rush	02:23:00	8	4000
22	Pirates of the Caribbean: Dead Mans Chest	2006	ACTION,ADVENTURE	Johnny Depp, Orlando Bloom	02:31:00	7.3	3650
23	Pirates of the Caribbean: At Worlds End	2007	ACTION,ADVENTURE	Johnny Depp, Keira Knightley	02:49:00	7.1	3550
24	Pirates of the Caribbean: On Stranger Tides	2011	ACTION,ADVENTURE	Johnny Depp, Penélope Cruz	02:16:00	6.6	3300
25	Pirates of the Caribbean: Dead Men Tell No Tales	2017	ACTION,ADVENTURE	Johnny Depp, Javier Bardem	02:09:00	6.5	3250

THUS, ACCORDING TO USER AGE, USER GENDER, MOVIE GENRE IT ADD UP POINT IN GIVEN FIELD FOR FOLLOWING TABLES:

6.3.1. MOVIE_ANALYSING TABLE

MOVIE_CODE	1_5	5_10	10_15	15_20	20_25	25_30	30_40	40_50	50_70	70PLUS	MALE	FEMALE	NIL
1	0	0	0	0	0	0	0	0	0	0	0	0	NULL
2	0	0	0	0	0	0	0	0	0	0	0	0	NULL
3	0	0	0	0	0	0	0	0	0	0	0	0	NULL
4	0	0	0	0	0	0	0	0	0	0	0	0	NULL
5	0	0	0	0	0	0	0	0	0	0	0	0	NULL
6	0	0	0	0	0	0	0	0	0	0	0	0	NULL
7	0	0	0	0	0	0	0	0	0	0	0	0	NULL
8	0	0	0	0	0	0	0	0	0	0	0	0	NULL
9	0	0	0	0	0	0	0	0	0	0	0	0	NULL
10	0	0	0	0	0	0	0	0	0	0	0	0	NULL
11	0	0	0	0	0	0	0	0	0	0	0	0	NULL
12	0	0	0	0	0	0	0	0	0	0	0	0	NULL
13	0	0	0	1	1	0	0	0	0	0	1	1	NULL
14	0	0	0	0	0	0	0	0	0	0	0	0	NULL
15	0	0	0	0	0	0	0	0	0	0	0	0	NULL
16	0	0	0	0	0	0	0	0	0	0	0	0	NULL
17	0	0	0	0	0	0	0	0	0	0	0	0	NULL
18	0	0	0	0	0	0	0	0	0	0	0	0	NULL
19	0	0	0	0	0	0	0	0	0	0	0	0	NULL
20	0	0	0	0	0	0	0	0	0	0	0	0	NULL
21	0	0	0	0	0	0	0	0	0	0	0	0	NULL
22	0	0	0	0	0	0	0	0	0	0	0	0	NULL
23	0	0	0	0	0	0	0	0	0	0	0	0	NULL
24	0	0	0	0	0	0	0	0	0	0	0	0	NULL
25	0	0	0	0	0	0	0	0	0	0	0	0	NULL

6.3.2. AGE_ANALYSING TABLE

GENERE	1_5	5_10	10_15	15_20	20_25	25_30	30_40	40_50	50_70	70PLUS	MALE	FEMALE	NIL
ACTION	0	0	0	0	0	0	0	0	0	0	0	0	NULL
ADVENTURE	0	0	0	0	0	0	0	0	0	0	0	0	NULL
ANIMATION	0	0	0	0	0	0	0	0	0	0	0	0	NULL
COMEDY	0	0	0	0	0	0	0	0	0	0	0	0	NULL
DRAMA	0	0	0	0	0	0	0	0	0	0	0	0	NULL
FAMILY	0	0	0	0	0	0	0	0	0	0	0	0	NULL
FANTASY	0	0	0	1	1	0	0	0	0	0	1	1	NULL
FICTION	0	0	0	0	0	0	0	0	0	0	0	0	NULL
HORROR	0	0	0	0	0	0	0	0	0	0	0	0	NULL
ROMANCE	0	0	0	0	0	0	0	0	0	0	0	0	NULL
SCI_FI	0	0	0	1	1	0	0	0	0	0	1	1	NULL
THRILLER	0	0	0	0	0	0	0	0	0	0	0	0	NULL

6.3.3. USER_ANALYSING TABLE

USER_ID	SCI_FI	ACTION	THRILLER	HORROR	ANIMATION	FAMILY	FANTASY	COMEDY	DRAMA	FICTION	ROMANCE	ADVENTURE
1	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0	0	0	0
33	0	0	0	0	0	0	0	0	0	0	0	0
34	0	0	0	0	0	0	0	0	0	0	0	0
35	0	0	0	0	0	0	0	0	0	0	0	0
36	1	0	0	0	0	0	1	0	0	0	0	0
37	0	0	0	0	0	0	0	0	0	0	0	0
38	1	0	0	0	0	0	1	0	0	0	0	0

NOTE: CURRENT USER user23 ‘S USER CODE IS 38.

6.3.4. RENT_HISTORY TABLE

```
mysql> SELECT * FROM RENT_HISTORY;
+---------+-----------+------------+-----------------------------------+-------------+------------+----------------+
| USER_ID | USER_NAME | MOVIE_CODE | MOVIE_NAME                        | RENTED_DATE | DUE_DATE   | PAYMENT_STATUS |
+---------+-----------+------------+-----------------------------------+-------------+------------+----------------+
|      36 | mithun    |         13 | Star Wars: Episode IV - A New Hope | 2023-11-09  | 2024-01-08 | PAYED          |
|      38 | user23    |         13 | Star Wars: Episode IV - A New Hope | 2023-11-09  | 2024-01-08 | DUE            |
+---------+-----------+------------+-----------------------------------+-------------+------------+----------------+
2 rows in set (0.00 sec)
```

NOTE: PREVIOUS USER ALSO BROUGHT THE SAME MOVIE, THUS YOU MAY NOTICE SOME POINTS IN DIFFERENT AGE COLUMN FOR SAME MOVIE .

6.4 RECOMMEND A MOVIE

```
##############################################################################
1.RENT A MOVIE
2.RECOMMEND A MOVIE
3.PAY FOR THE MOVIE RENT
4.EXIT
ENTER THE CORRECT SNO FOR GIVEN OPTIONS
SNO:2

LET US ANALYSE FOR YOU
BASED ON COMMON FEED, MOST FEMALES AT YOUR AGE WOULD LIKE TO WATCH FANTASY AND SCI_FI THUS I RECOMMEND:
1 . Passenger CODE= 1
2 . Interstellar CODE= 4
3 . Avengers: Endgame CODE= 5
4 . Self/less CODE= 11
5 . Inception CODE= 12

BASED ON YOUR FEED, YOU WOULD LIKE TO WATCH ACTION AND SCI_FI THUS I RECOMMEND:
1 . Passenger CODE= 1
2 . Interstellar CODE= 4
3 . Avengers: Endgame CODE= 5
4 . Spider Man Across the Universe CODE= 6
5 . Free Guy CODE= 10



##############################################################################
1.RENT A MOVIE
2.RECOMMEND A MOVIE
3.PAY FOR THE MOVIE RENT
```

BASED ON POINTS ON ABOVE TABLES IT RECOMMENDS A MOVIE.

6.5 PAY FOR THE MOVIE RENT

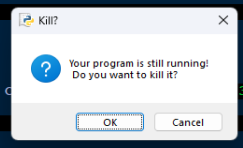
```
5 . Free Guy CODE= 10



##############################################################################
1.RENT A MOVIE
2.RECOMMEND A MOVIE
3.PAY FOR THE MOVIE RENT
4.EXIT
ENTER THE CORRECT SNO FOR GIVEN OPTIONS
SNO:3

YOU HAVE DUE ON THE MOVIES:
Star Wars: Episode IV - A New Hope MOVIE_CODE= 13
HAVE NOT PAYED YET
ENTER THE MOVIE CODE FOR PAYMENT...NOTE YOU CAN PAY ONLY ONE MOVIE AT A TIME13
YOUR PRICE IS: 4300
ENTER 1 TO PROCCED TRANSACTION..1
YOUR TRANSACTION IS SUCCESSFUL



##############################################################################
1.RENT A MOVIE
2.RECOMMEND A MOVIE
3.PAY FOR THE MOVIE RENT
4.EXIT
ENTER THE CORRECT SNO FOR GIVEN OPTIONS
SNO:
```

```
6 rows in set (0.24 sec)

mysql> SELECT * FROM RENT_HISTORY;
+---------+-----------+------------+-----------------------------------+-------------+------------+----------------+
| USER_ID | USER_NAME | MOVIE_CODE | MOVIE_NAME                        | RENTED_DATE | DUE_DATE   | PAYMENT_STATUS |
+---------+-----------+------------+-----------------------------------+-------------+------------+----------------+
|      36 | mithun    |         13 | Star Wars: Episode IV - A New Hope | 2023-11-09  | 2024-01-08 | PAYED          |
|      38 | user23    |         13 | Star Wars: Episode IV - A New Hope | 2023-11-09  | 2024-01-08 | PAYED          |
+---------+-----------+------------+-----------------------------------+-------------+------------+----------------+
2 rows in set (0.03 sec)
```

NOTE: THE PAYMENT_STATUS HAS BEEN UPDATED FROM “DUE” TO “PAYED”

6.6 EXIT

```
*IDLE Shell 3.11.3*
File  Edit  Shell  Debug  Options  Window  Help

BASED ON YOUR FEED, YOU WOULD LIKE TO WATCH ACTION AND SCI_FI THUS I RECOMMEND:
1 . Passenger CODE= 1
2 . Interstellar CODE= 4
3 . Avengers: Endgame CODE= 5
4 . Spider Man Across the Universe CODE= 6
5 . Free Guy CODE= 10



################################################################################
1.RENT A MOVIE
2.RECOMMEND A MOVIE
3.PAY FOR THE MOVIE RENT
4.EXIT
ENTER THE CORRECT SNO FOR GIVEN OPTIONS
SNO:3

YOU HAVE DUE ON THE MOVIES:
Star Wars: Episode IV - A New Hope MOVIE_CODE= 13
HAVE NOT PAYED YET
ENTER THE MOVIE CODE FOR PAYMENT...NOTE YOU CAN PAY O                    3
YOUR PRICE IS: 4300
ENTER 1 TO PROCCED TRANSACTION..1
YOUR TRANSACTION IS SUCCESSFUL



################################################################################
1.RENT A MOVIE
2.RECOMMEND A MOVIE
3.PAY FOR THE MOVIE RENT
4.EXIT
ENTER THE CORRECT SNO FOR GIVEN OPTIONS
SNO:4

HOPE WE MEET AGAIN THANK YOU
YOU WOULD AUTOMATICALLY EXIT IN 10 SEC
                                                                    Ln: 109  Col: 0
```



6.7 CHECKDUE, FINE

```
+---------+-----------+------------+-----------------------------------+-------------+------------+----------------+
| USER_ID | USER_NAME | MOVIE_CODE | MOVIE_NAME                        | RENTED_DATE | DUE_DATE   | PAYMENT_STATUS |
+---------+-----------+------------+-----------------------------------+-------------+------------+----------------+
|      36 | mithun    |         13 | Star Wars: Episode IV - A New Hope | 2023-11-09  | 2024-01-08 | PAYED          |
|      38 | user23    |         13 | Star Wars: Episode IV - A New Hope | 2024-01-09  | 2023-01-09 | DUE            |
+---------+-----------+------------+-----------------------------------+-------------+------------+----------------+
```

```
ENTER THE CORRECT SNO FOR GIVEN OPTIONS
SNO:
========== RESTART: C:\Users\mithu\Desktop\SCHOOL PROJECT\MOVIES.py ==========
SUCCESSFULLY CONNECTED TO MYSQL..
ENTER THE USER NAME:user23
ENTER YOUR ID:38
WELCOME USER23
YOUR RENT ON THE MOVIES:-Star Wars: Episode IV - A New Hope code= 13,HAVE NOT RETURNED AND YOUR DUE DATE HAD BEEN OVER.PLEASE COM
PLETE YOUR DUE TO CLEAR ALL RESTRICTION
YOU CANT ACCSESS ANY FUNCTION WITHOUT CLEARING YOUR EXPIRY DUE..
YOU HAVE DUE ON THE MOVIES:
Star Wars: Episode IV - A New Hope MOVIE_CODE= 13
HAVE NOT PAYED YET
ENTER THE MOVIE CODE FOR PAYMENT...NOTE YOU CAN PAY ONLY ONE MOVIE AT A TIME13
YOUR FINAL AMOUNT: 4300 + 65360.0 (FINE)= 69660.0
YOU HAVE FINED BECAUSE YOUR DUE DATE IS OVER BEFORE: 304 DAYS
ENTER 1 TO PROCCED TRANSACTION..1
YOUR TRANSACTION IS SUCCESSFUL
WHAT CAN I DO FOR YOU:

################################################################################
1.RENT A MOVIE
2.RECOMMEND A MOVIE
3.PAY FOR THE MOVIE RENT
```

NOTE: SOME EDITED FIELD HAVE TAKEN PLACE FOR CHECKING PURPOSES.

7. CONCLUSION:

In summary, the Movie Database Management System stands as a user-friendly and adaptable solution for seamless movie rentals. Its efficiency in simplifying selection, reservation, and updates makes it a valuable asset for diverse applications. The system's ease of modification ensures flexibility, allowing it to cater to the specific needs of different environments. Whether implemented in entertainment venues or community spaces, its versatility shines through. This system not only facilitates a hassle-free movie-watching experience but also serves as a scalable solution ready for customization in response to evolving requirements. Its potential to enhance user convenience and streamline movie library management makes it a valuable asset in varied settings, promising an improved and accessible cinematic experience for all.

8. BIBLIOGRAPHY

8.1 BOOKS REFERRED:

SUMITA ARORA COMPUTER SCIENCE WITH PYTHON(CLASS 11,12)

8.2 WEBSITES REFERRED:

https://www.w3schools.com/

https://www.geeksforgeeks.org/