CAR RENTAL SYSTEM

(by Neeharika Morla)

dao:

addCar.py

```
from util.DBConnUtil import dbConnection


def addVehicle():
    vehicleID = int(input("Enter VehicleID: "))
    make = input("Enter make: ")
    model = input("Enter model")
    year = int(input("Enter year"))
    dailyRate = float(input("Enter daily Rate: "))
    status = input("Enter status: ")
    passengerCapacity = int(input("Enter passenger capacity: "))
    engineCapacity = int(input("Enter engine capacity: "))
    try:
        conn, stmt = dbConnection.open()
        data = [(vehicleID, make, model, year, dailyRate, status, passengerCapacity, engineCapacity)]
        stmt.executemany('''insert into
Vehicle(vehicleID,make,model,year,dailyRate,status,passengerCapacity,engineCapacity)
                         values(%s,%s,%s,%s,%s,%s,%s,%s)''', data)
        conn.commit()
        print("Data inserted sucessfully")

    except Exception as E:
        print(f"ERROR DURING INSERTION {E}")
    finally:
        if conn:
            dbConnection.close(conn)
```

addCustomer.py

```
from util.DBConnUtil import dbConnection
def addCustomer():
    customerID=int(input("Enter Customer ID: "))
    firstName=input("Enter First Name: ")
    lastName=input("Enter Last Name: ")
    email=input("Enter email: ")
    phoneNumber=input("Enter Phone number: ")
    try:
        conn,stmt=dbConnection.open()
        data=[(customerID,firstName,lastName,email,phoneNumber)]
        stmt.executemany('''insert into Customer(customerID,firstName,lastName,email,phoneNumber)
                         values(%s,%s,%s,%s,%s)''',data)
        conn.commit()
        print("Data inserted Sucessfully")
    except Exception as E:
        print(f"ERROR DURING INSERTION {E}")
    finally:
        if conn:
            dbConnection.close(conn)
```

createLease.py

```
from util.DBConnUtil import dbConnection

def createLease():
    leaseID=int(input("Enter Lease ID: "))
    vehicleID=int(input("Enter VehicleID: "))
    customerID=int(input("Enter Customer ID: "))
    startDate=input("Enter Start Date: ")
    endDate=input("Enter end date: ")
```

```
    type=input("Enter (DailyLease/MonthlyLease) type: ")
    try:
        conn,stmt=dbConnection.open()
        data=[(leaseID,vehicleID,customerID,startDate,endDate,type)]
        stmt.executemany('''insert into Lease(leaseID,vehicleID,customerID,startDate,endDate,type)
                            values(%s,%s,%s,%s,%s,%s)''',data)
        conn.commit()
        print("Data inserted Successfully")
    except Exception as E:
        print(f"ERROR DURING INSERTION {E}")
    finally:
        if conn:
            dbConnection.close(conn)
```

createTables.py

```
from util.DBConnUtil import dbConnection


def createTables():
    conn = None
    try:
        conn, stmt = dbConnection.open()
        if conn:
            stmt.execute('''create table if not exists Vehicle(vehicleID int primary key,
                                make varchar(20), model varchar(30), year int, dailyRate
decimal(10,2),
                                status char(1), passengerCapacity int, engineCapacity int) ''')

            stmt.execute('''create table Customer(customerID int primary key,
                                firstName varchar(30), lastName varchar(30),
                                email varchar(30),phoneNumber char(10))''')

            stmt.execute('''create table Lease(leaseID int primary key,
                                vehicleID int,
                                customerID int,
                                startDate date,
                                endDate date,
                                types varchar(20),
                                foreign key (vehicleID) references Vehicle (vehicleID),
                                foreign key (customerID) references Customer (customerID))''')

            stmt.execute('''create table Payment(paymentID int primary key,
                                leaseID int,
                                paymentDate date,
                                Amount decimal(10,2),
                                foreign key (leaseID) references Lease (leaseID))''')
    except Exception as E:
        print(f"Error during database initialization: {E}")


    finally:
        if conn:
            dbConnection.close(conn)
```

findCarsByID.py

```
from util.DBConnUtil import dbConnection
from exception.carNotFound import CarNotFoundException
def findCarsByID():
    VehicleID=int(input("Enter Vehicle ID: "))
    try:
        conn,stmt=dbConnection.open()
        if not CarNotFoundException.isCarExists(VehicleID, conn, stmt):
            raise CarNotFoundException(VehicleID)
        data=(VehicleID,)
        stmt.execute('''select * from Vehicle where VehicleID=%s''',data)
        record=stmt.fetchall()
        for i in record:
            print(i)
    except CarNotFoundException as ce:
        print(f"ERROR: {ce}")
    except Exception as E:
```

```
        print(f"ERROR: {E}")
    finally:
        if conn:
            dbConnection.close(conn)
```

findCustomerByID.py

```
from util.DBConnUtil import dbConnection
from exception.customerNotFound import CustomerNotFoundException
def findCustomerByID():
    CustomerID=int(input("Enter Customer ID: "))
    try:
        conn,stmt=dbConnection.open()
        if not CustomerNotFoundException.isCustomerExists(CustomerID, conn, stmt):
            raise CustomerNotFoundException(CustomerID)
        data=(CustomerID,)
        stmt.execute('''select * from Customer where CustomerID=%s''',data)
        record=stmt.fetchall()
        for i in record:
            print(i)
    except CustomerNotFoundException as ce:
        print(f"ERROR: {ce}")
    except Exception as E:
        print(f"ERROR: {E}")
    finally:
        if conn:
            dbConnection.close(conn)
```

listAllCars.py

```
from util.DBConnUtil import dbConnection
def listAllCars():
    try:
        conn,stmt=dbConnection.open()
        stmt.execute('''select * from vehicle''')
        record=stmt.fetchall()
        for i in record:
            print(i)
    except Exception as E:
        print(f"ERROR: {E}")
    finally:
        if conn:
            dbConnection.close(conn)
```

listAvailableCars.py

```
from util.DBConnUtil import dbConnection
def listAvailableCars():
    try:
        conn,stmt=dbConnection.open()
        stmt.execute('''select * from vehicle where status='available' ''')
        record=stmt.fetchall()
        for i in record:
            print(i)
    except Exception as E:
        print(f"ERROR: {E}")
    finally:
        if conn:
            dbConnection.close(conn)
```

listCustomer.py

```
from util.DBConnUtil import dbConnection
def listCustomers():
    try:
        conn,stmt=dbConnection.open()
        stmt.execute('''select * from Customer''')
```

```
        record=stmt.fetchall()
        for i in record:
            print(i)
    except Exception as E:
        print(f"ERROR: {E}")
    finally:
        if conn:
            dbConnection.close(conn)
```

listLeaseHistory.py

```
from util.DBConnUtil import dbConnection
def listLeaseHistory():
    try:
        conn,stmt=dbConnection.open()
        stmt.execute('''select * from Lease''')
        record=stmt.fetchall()
        for i in record:
            print(i)
    except Exception as E:

        print(f"ERROR: {E}")
    finally:
        if conn:
            dbConnection.close(conn)
```

listRentedCars.py

```
from util.DBConnUtil import dbConnection
def listRentedCars():
    try:
        conn,stmt=dbConnection.open()
        stmt.execute('''select V.VehicleID,V.make from Vehicle as V
                        join Lease as L on L.VehicleID=V.VehicleID''')
        record=stmt.fetchall()
        for i in record:
            print(i)
    except Exception as E:
        print(f"ERROR: {E}")
    finally:
        if conn:
            dbConnection.close(conn)
```

paymentAmount.py

```
from util.DBConnUtil import dbConnection
def paymentHistory():
    try:
        conn,stmt=dbConnection.open()
        stmt.execute('''select * from Payment''')
        record=stmt.fetchall()
        for i in record:
            print(i)
    except Exception as E:
        print(f"ERROR: {E}")
    finally:
        if conn:
            dbConnection.close(conn)
```

recordPayment.py

```
from util.DBConnUtil import dbConnection
def recordPayment():
    paymentID=int(input("Enter Payment ID: "))
    leaseID=int(input("Enter Lease ID: "))
    paymentDate=input("Enter payment Date")
    Amount=float(input("Enter amount: "))
```

```
    try:
        conn,stmt=dbConnection.open()
        data=[(paymentID,leaseID,paymentDate,Amount)]
        stmt.executemany('''insert into Payment(paymentID,leaseID,paymentDate,Amount)
                            values(%s,%s,%s,%s)''',data)
        conn.commit()
        print("Data inserted Successfully")
    except Exception as E:
        print(f"ERROR DURING INSERTION {E}")
    finally:
        if conn:
            dbConnection.close(conn)
```

removeCar.py

```
from util.DBConnUtil import dbConnection
from exception.carNotFound import CarNotFoundException
def removeCar():
    VehicleID=int(input("Enter Vehicle ID: "))

    try:
        conn,stmt=dbConnection.open()
        if not CarNotFoundException.isCarExists(VehicleID, conn, stmt):
            raise CarNotFoundException(VehicleID)
        data=(VehicleID,)
        stmt.execute('''delete from Vehicle where VehicleID=%s''',data)
        conn.commit()
        print("Car Deleted")
    except CarNotFoundException as ce:
        print(f"ERROR: {ce}")
    except Exception as E:
        print(f"ERROR: {E}")
    finally:
        if conn:
            dbConnection.close(conn)
```

removeCustomer.py

```
from util.DBConnUtil import dbConnection
from exception.customerNotFound import CustomerNotFoundException
def removeCustomer():
    CustomerID=int(input("Enter Customer ID: "))
    try:
        conn,stmt=dbConnection.open()
        if not CustomerNotFoundException.isCustomerExists(CustomerID, conn, stmt):
            raise CustomerNotFoundException(CustomerID)
        data=(CustomerID,)
        stmt.execute('''delete from Customer where CustomerID=%s''',data)
        conn.commit()
        print("Customer Removed")
    except CustomerNotFoundException as ce:
        print(f"ERROR: {ce}")
    except Exception as E:
        print(f"ERROR: {E}")
    finally:
        if conn:
            dbConnection.close(conn)
```

updateCarInfo.py

```
from util.DBConnUtil import dbConnection
from exception.carNotFound import CarNotFoundException
def UpdateCarAvailability():
    VehicleID=int(input("Enter vehicle ID: "))
    status=input("Enter Updated Status: ")
    try:
        conn,stmt=dbConnection.open()
        if not CarNotFoundException.isCarExists(VehicleID, conn, stmt):
            raise CarNotFoundException(VehicleID)
        data=(status,VehicleID)
```

```
        stmt.execute('''update Vehicle set status=%s where VehicleID=%s''',data)
        conn.commit()
        print("Data updated Successfully")
    except CarNotFoundException as ce:
        print(f"ERROR: {ce}")
    except Exception as E:
        print(E)
    finally:
        if conn:
            dbConnection.close(conn)
```

entity:

customer.py

```
class Customer:
    def __init__(self,customerID,firstName,lastName,email,phoneNumber):
        self.customerID=customerID
        self.firstName=firstName
        self.lastName=lastName
        self.email=email
        self.phoneNumber=phoneNumber
```

lease.py

```
class Lease:
    def __init__(self,leaseID,vehicleID,customerID,startDate,endDate,type):
        self.leaseID=leaseID
        self.vehicleID=vehicleID
        self.customerID=customerID
        self.startDate=startDate
        self.endDate=endDate
        self.type=type
```

payment.py

```
class Payment:
    def __init__(self,paymentID,leaseID,paymentDate,amount):
        self.paymentID=paymentID
        self.leaseID=leaseID
        self.paymentDate=paymentDate
        self.amount=amount
```

vehicle.py

```
class Vehicle:
    def __init__(self,vehicleID,make,model,year,dailyRate,status,passengerCapacity,engineCapacity):
        self.vehicleID=vehicleID
        self.make=make
        self.model=model
        self.year=year
        self.dailyRate=dailyRate
        self.status=status
        self.passengerCapacity=passengerCapacity
        self.engineCapacity=engineCapacity
```

exception:

carNotFound.py

```
class CarNotFoundException(Exception):
    def __init__(self, VehicleID, message="Car not found"):
        self.VehicleID = VehicleID
        self.message = message
        super().__init__(self.message)

    @staticmethod
    def isCarExists(VehicleID, conn, stmt):
        stmt.execute('SELECT 1 FROM Vehicle WHERE VehicleID = %s', (VehicleID,))
        return bool(stmt.fetchone())
```

customerNotFound.py

```
class CustomerNotFoundException(Exception):
    def __init__(self, CustomerID, message="Customer not found"):
        self.CustomerID = CustomerID
        self.message = message
        super().__init__(self.message)

    @staticmethod
    def isCustomerExists(CustomerID, conn, stmt):
        stmt.execute('SELECT 1 FROM Customer WHERE CustomerID = %s', (CustomerID,))
        return bool(stmt.fetchone())
```

leaseNotFound.py

```
class LeaseNotFoundException(Exception):
    def __init__(self,LeaseID,message="Lease Not fund\n"):
        self.LeaseID=LeaseID
        self.message=message
        super().__init__(self.message)
    def isLeaseExists(LeaseID,conn,stmt):
        stmt.execute('SELECT 1 FROM Lease WHERE LeaseID = %s', (LeaseID,))
        return bool(stmt.fetchone())
```

main:

main.py

```
from dao.createTables import createTables
from dao.addCar import addVehicle
from dao.addCustomer import addCustomer
from dao.createLease import createLease
from dao.recordPayment import recordPayment
from dao.listAvailableCars import listAvailableCars
from dao.listRentedCars import listRentedCars
from dao.findCarsByID import findCarsByID
from dao.listCustomer import listCustomers
from dao.findCustomerByID import findCustomerByID
from dao.listLeaseHistory import listLeaseHistory
from dao.removeCar import removeCar
from dao.removeCustomer import removeCustomer
from dao.updateCarInfo import UpdateCarAvailability
from dao.paymentAmount import paymentHistory
if __name__=="__main__":
    try:
        print("\n *****  WELCOME TO CAR RENTAL SYSTEM  *****")
        print("\n----------------MAIN MENU----------------------")
        print("\nEnter 1 for car management")
        print("\nEnter 2 for customer management")
        print("\nEnter 3 for lease management")
        print("\nEnter 4 for payment management")
        print("\n Enter 5 to exit")
        while(1>0):
            ch=input("\nEnter your choice from main menu: ")
            if ch=='1':
                print("\nCAR MANAGEMENT")
                print("\nEnter 1 to add new car")
                print("\nEnter 2 to remove car")
                print("\nEnter 3 to list available cars")
                print("\nEnter 4 to find cars by ID")
                print("\nEnter 5 to Update car Availabitity")
                print("\nEnter 6 to go back")
                while(1>0):
                    c=input("\nEnter Choice: ")
                    if c=='1':
                        addVehicle()
                    elif c=='2':
                        removeCar()
                    elif c=='3':
                        listAvailableCars()
                    elif c=='4':
                        findCarsByID()
                    elif c=='5':
                        UpdateCarAvailability()
                    elif c=='6':
                        break
            elif ch=='2':
                print("\nCUSTOMER MANAGEMENT\n")
                print("\nEnter 1 to add new customer")
                print("\nEnter 2 to remove customer")
                print("\nEnter 3 to list customers")
                print("\nEnter 4 to find customer by ID")
                print("\nEnter 5 to go back")
                while(1>0):
                    c=input("\nEnter Choice: ")
                    if c=='1':
                        addCustomer()
                    elif c=='2':
                        removeCustomer()
                    elif c=='3':
                        listCustomers()
                    elif c=='4':
                        findCustomerByID()
                    elif c=='5':
                        break
            elif ch=='3':
                print("\nLEASE MANAGEMENT\n")
                print("\nEnter 1 to create new Lease")
                print("\nEnter 2 to List lease history")
                print("\nEnter 3 to go back")
                while(1>0):
                    c=input("\nEnter Choice: ")
```

```
                    if c=='1':
                        createLease()
                    elif c=='2':
                        listLeaseHistory()
                    elif c=='3':
                        break
            elif ch=='4':
                print("\nPAYMENT MANAGEMENT\n")
                print("\nEnter 1 to record new Payment")
                print("\nEnter 2 to retrive payment history")
                print("\nEnter 3 to go back")
                while(1>0):
                    c=input("\nEnter Choice: ")
                    if c=='1':
                        recordPayment()
                    elif c=='2':
                        paymentHistory()
                    elif c=='3':
                        break
            elif ch=='5':
                print("----------thank you for visiting car rental system---------------")
                break
            else:
                print("Invalid Choice")

    except Exception as E:
        print(f"\nAn error has occured: {E}")
```

util:

DBConnUtil.py

```
import mysql.connector as sql
from util.DBPropertyUtil import DBPropertyUtil

class dbConnection:
    def open():
        try:
            s=DBPropertyUtil.get_property_string()
            conn=sql.connect(host=s[0],username=s[1],database=s[2],password=s[3])
            stmt=conn.cursor()
            return conn,stmt
        except Exception as E:
            print(E)
            return None,None
    def close(conn):
        try:
            conn.close()
            print("CONNECTION CLOSED")
        except Exception as E:
            print(E)
```

DBPropertyUtil.py

```
class DBPropertyUtil:
    def get_property_string():
        host='localhost'
        username='root'
        database='carrental'
        password='sweetysmiley'
        return host,username,database,password
```

Outputs:

```
*****  WELCOME TO CAR RENTAL SYSTEM  *****

----------------MAIN MENU----------------------

Enter 1 for car management

Enter 2 for customer management

Enter 3 for lease management

Enter 4 for payment management

 Enter 5 to exit

Enter your choice from main menu:
```

```
Enter your choice from main menu: 1

CAR MANAGEMENT

Enter 1 to add new car

Enter 2 to remove car

Enter 3 to list available cars

Enter 4 to find cars by ID

Enter 5 to Update car Availabitity

Enter 6 to go back
```

```
Enter Choice: 1
Enter VehicleID: 3
Enter make: toyota
Enter modelv4
Enter year2023
Enter daily Rate: 678
Enter status: available
Enter passenger capacity: 6
Enter engine capacity: 6873
Data inserted sucessfully
```

```
Enter Choice: 2
Enter Vehicle ID: 3
Car Deleted
```

```
Enter Choice: 3
(2, 'bmw', 'b3', 2023, Decimal('678.00'), 'available', 5, 67)
```

```
Enter Choice: 4
Enter Vehicle ID: 2
(2, 'bmw', 'b3', 2023, Decimal('678.00'), 'available', 5, 67)
```

```
Enter Choice: 5
Enter vehicle ID: 2
Enter Updated Status: notavailable
Data updated Successfully
```

```
CUSTOMER MANAGEMENT



Enter 1 to add new customer

Enter 2 to remove customer

Enter 3 to list customers

Enter 4 to find customer by ID

Enter 5 to go back

Enter Choice:
```

```
Enter Choice: 1
Enter Customer ID: 3
Enter First Name: muskaan
Enter Last Name: saxena
Enter email: m@gmail.com
Enter Phone number: 5675452562
Data inserted Sucessfully
```

```
Enter Choice: 2
Enter Customer ID: 3
Customer Removed
```

```
Enter Choice: 3
(1, 'neeha', 'morla', 'neha@email.com', '4567890432')
(3, 'muskaan', 'saxena', 'm@gmail.com', '5675452562')
```

```
Enter Choice: 4
Enter Customer ID: 1
(1, 'neeha', 'morla', 'neha@email.com', '4567890432')
CONNECTION CLOSED
```

```
Enter your choice from main menu: 3

LEASE MANAGEMENT



Enter 1 to create new Lease

Enter 2 to List lease history

Enter 3 to go back

Enter Choice:
```

```
Enter Choice: 1
Enter Lease ID: 3
Enter VehicleID: 1
Enter Customer ID: 1
Enter Start Date: 2023-09-21
Enter end date: 2024-03-03
Enter (DailyLease/MonthlyLease) type: MonthlyLease
Data inserted Successfully
```

```
Enter Choice: 2
(1, 1, 1, datetime.date(2023, 7, 9), datetime.date(2024, 1, 3), 'DailyLease')
(3, 1, 1, datetime.date(2023, 9, 21), datetime.date(2024, 3, 3), 'MonthlyLease')
(6, 1, 1, datetime.date(2023, 4, 1), datetime.date(2024, 3, 1), 'DailyLease')
```

```
PAYMENT MANAGEMENT



Enter 1 to record new Payment

Enter 2 to retrive payment history

Enter 3 to go back

Enter Choice:
```

```
Enter Choice: 1
Enter Payment ID: 9
Enter Lease ID: 1
Enter payment Date2023-09-08
Enter amount: 39000
Data inserted Successfully
```

```
Enter Choice: 2
(1, 1, datetime.date(2023, 3, 7), Decimal('250000.00'))
(3, 1, datetime.date(2023, 8, 16), Decimal('250000.00'))
(9, 1, datetime.date(2023, 9, 8), Decimal('39000.00'))
```

```
Enter your choice from main menu: 5
----------thank you for visiting car rental system--------------
```