HEXAWARE PYTHON CODING CHALLENGE

NAME: AISHWARYA B
SUPERSET ID: 5006869
Project Name: PetPals
Date: 27/06/2025

Problem Statement:

PetPals, The Pet Adoption Platform scenario is a software system designed to facilitate the adoption of pets, such as dogs and cats, from shelters or rescue organisations. This platform serves as a digital marketplace where potential adopters can browse and select pets, shelters can list available pets, and donors can contribute to support animal welfare.

NOTE 1: *I have designed and created the database tables to ensure a systematic and logical flow aligned with the PetPals problem statement. Each table represents a real-world entity, such as* ***pets, users, shelters, donations, participants, adoption and adoption events****, for clear-cut functionality throughout the application.*

NOTE 2: *At the end of the project, I added a custom business logic feature called the* ***Donation Leaderboard****, which is placed at the bottom of the adopter menu flow.*

Implement OOPs:

Create a SQL Schema from the pet and user classes, and use the class attributes for table column names.

1. Create and implement the mentioned class and the structure in your application.

Pet Class:

Attributes:
• Name (string): The name of the pet.
• Age (int): The age of the pet.
• Breed (string): The breed of the pet. Methods:
• Constructor to initialise Name, Age, and Breed.
• Getters and setters for attributes.
• ToString() method to provide a string representation of the pet.

```
# entity/pet.py

class Pet:
    def __init__(self, petid, name, age, breed, pet_type,
availableforadoption, ownerid, shelterid):
        self.petid = petid
```

```
        self.name = name
        self.age = age
        self.breed = breed
        self.type = pet_type
        self.availableforadoption = availableforadoption
        self.ownerid = ownerid
        self.shelterid = shelterid

    def __str__(self):
        return f"[{self.petid}] {self.name} ({self.type}) -
{self.breed}, Age: {self.age}, Available:
{bool(self.availableforadoption)}"
```

Dog Class (Inherits from Pet):

Additional Attributes:
• DogBreed (string): The specific breed of the dog.
Additional Methods:
• Constructor to initialise DogBreed.
• Getters and setters for DogBreed.

```
# entity/dog.py

from entity.pet import Pet

class Dog(Pet):
    def __init__(self, petid, name, age, breed, availableforadoption,
ownerid, shelterid):
        super().__init__(petid, name, age, breed, "Dog",
availableforadoption, ownerid, shelterid)
```

Cat Class (Inherits from Pet):
Additional Attributes:
• CatColor (string): The colour of the cat.
Additional Methods:
• Constructor to initialise CatColor.
• Getters and setters for CatColor.

```
# entity/cat.py

from entity.pet import Pet
```

```
class Cat(Pet):
    def __init__(self, petid, name, age, breed, availableforadoption,
ownerid, shelterid):
        super().__init__(petid, name, age, breed, "Cat",
availableforadoption, ownerid, shelterid)
```

3.PetShelter Class:
Attributes:
• availablePets (List of Pet): A list to store available pets for adoption.
Methods:
• AddPet(Pet pet): Adds a pet to the list of available pets.
• RemovePet(Pet pet): Removes a pet from the list of available pets.
• ListAvailablePets(): Lists all available pets in the shelter.

```
# entity/petshelter.py

class PetShelter:
    def __init__(self):
        self.available_pets = []

    def add_pet(self, pet):
        self.available_pets.append(pet)

    def remove_pet(self, petid):
        self.available_pets = [p for p in self.available_pets if
p.petid != petid]

    def list_available_pets(self):
        for pet in self.available_pets:
            print(pet)
```

4.Donation Class (Abstract):

Attributes:
• DonorName (string): The name of the donor.
• Amount (decimal): The donation amount.
Methods:
• Constructor to initialize DonorName and Amount.
• Abstract method RecordDonation() to record the donation (to be implemented in derived classes).

```python
# entity/donation.py

from abc import ABC, abstractmethod

class Donation(ABC):
    def __init__(self, donorname, amount):
        self.donorname = donorname
        self.amount = amount

    @abstractmethod
    def record_donation(self):
        pass
```

CashDonation Class (Derived from Donation):

Additional Attributes:
• DonationDate (DateTime): The date of the cash donation.
Additional Methods:
• Constructor to initialise DonationDate.
• Implementation of RecordDonation() to record a cash donation.

```python
# entity/cash_donation.py

from entity.donation import Donation

class CashDonation(Donation):
    def __init__(self, donorname, amount, donationdate):
        super().__init__(donorname, amount)
        self.donationdate = donationdate

    def record_donation(self):
        return f"Cash donation of ₹{self.amount} by {self.donorname} on
{self.donationdate}"
```

ItemDonation Class (Derived from Donation):
Additional Attributes:
• ItemType (string): The type of item donated (e.g., food, toys).
Additional Methods:
• Constructor to initialize ItemType.

• Implementation of RecordDonation() to record an item donation.

```
# entity/item_donation.py

from entity.donation import Donation

class ItemDonation(Donation):
    def __init__(self, donorname, itemtype):
        super().__init__(donorname, 0)
        self.itemtype = itemtype

    def record_donation(self):
        return f"Item donation: {self.itemtype} by {self.donorname}"
```

5.IAdoptable Interface/Abstract Class:

Methods:
• Adopt(): An abstract method to handle the adoption process.
AdoptionEvent Class:
Attributes:
• Participants (List of IAdoptable): A list of participants (shelters and adopters) in the adoption event. Methods:
• HostEvent(): Hosts the adoption event.
• RegisterParticipant(IAdoptable participant): Registers a participant for the event.

```
# entity/adoption_event.py

class IAdoptable:
    def adopt(self):
        raise NotImplementedError("Subclasses should implement this!")

class AdoptionEvent:
    def __init__(self, eventname, eventdate, location):
        self.eventname = eventname
        self.eventdate = eventdate
        self.location = location
        self.participants = []

    def register_participant(self, participant: IAdoptable):
        self.participants.append(participant)

    def host_event(self):
```

```
        print(f"Hosting {self.eventname} on {self.eventdate} at
{self.location}")
        for p in self.participants:
            print(f"Participant: {p}")
```

6. Exception handling

Create and implement the following exceptions in your application.

• Invalid Pet Age Handling:

o In the Pet Adoption Platform, when adding a new pet to a shelter, the age of the pet should be a positive integer. Write a program that prompts the user to input the age of a pet. Implement exception handling to ensure that the input is a positive integer. If the input is not valid, catch the exception and display an error message. If the input is valid, add the pet to the shelter.

```
# exception/invalid_pet_age_exception.py

class InvalidPetAgeException(Exception):
    def __init__(self, message="Pet age must be a positive integer."):
        super().__init__(message)
```

```
=== Shelter Menu ===
1. Add New Pet
2. Register Shelter for Adoption Event
3. View Adoption Events
0. Back to Main Menu
Enter choice: 1
Pet name: Cookie
Pet age: -1
Breed: Shitzu
Type (Dog/Cat): Dog
Shelter ID: 5
Error adding pet: Invalid pet age: -1. Age cannot be negative.
```

• Null Reference Exception Handling:

o In the Pet Adoption Platform, when displaying the list of available pets in a shelter, it's important to handle situations where a pet's properties (e.g., Name, Age) might be null. Implement exception handling to catch null reference exceptions when accessing properties of pets in the shelter and display a message indicating that the information is missing.

```
# exception/null_reference_exception.py

class NullReferenceException(Exception):
    def __init__(self, message="One or more required fields are
missing."):
        super().__init__(message)
```

To prevent runtime errors while displaying pet details, the application includes null checks when accessing pet attributes like name or age. If any property is missing (i.e., None in Python), a message is shown indicating that the pet's information is incomplete. This avoids crashes and improves data reliability when displaying shelter listings.

• **Insufficient Funds Exception:**

o Suppose the Pet Adoption Platform allows users to make cash donations to shelters. Write a program that prompts the user to enter the donation amount. Implement exception handling to catch situations where the donation amount is less than the minimum allowed amount (e.g., $10). If the donation amount is insufficient, catch the exception and display an error message. Otherwise, process the donation.

```
# exception/insufficient_funds_exception.py

class InsufficientFundsException(Exception):
    def __init__(self, message="Donation amount is below the minimum
allowed."):
        super().__init__(message)
```

```
=== PetPals Platform ===
1. Adopter Menu
2. Shelter Menu
0. Exit
Enter choice: 1

=== Adopter Menu ===
1. View Available Pets
2. Adopt a Pet
3. Make a Donation
4. Register for Adoption Event
5. View Adoption Events
0. Back to Main Menu
Enter choice: 3
Enter your name: Emily
Enter donation type (Cash/Item): Cash
Enter Shelter ID: 4
Enter donation amount: 400
Error: Minimum donation is 500
```

• File Handling Exception:

o In the Pet Adoption Platform, there might be scenarios where the program needs to read data from a file (e.g., a list of pets in a shelter). Write a program that attempts to read data from a file. Implement exception handling to catch any file-related exceptions (e.g., FileNotFoundException) and display an error message if the file is not found or cannot be read.

```python
# exception/file_handling_exception.py

class FileHandlingException(Exception):
    def __init__(self, message="Error reading file or file not found."):
        super().__init__(message)
```

The application uses a custom FileHandlingException to handle file-related errors like missing files or read/write failures. It ensures the program doesn't crash and displays a clear message when file operations fail.

• Custom Exception for Adoption Errors:

o Design a custom exception class called AdoptionException that inherits from Exception. In the Pet Adoption Platform, use this custom exception to handle adoption-related errors, such as attempting to adopt a pet that is not available or adopting a pet with missing information. Create instances of AdoptionException with different error messages and catch them appropriately in your program.

```python
# exception/adoption_exception.py

class AdoptionException(Exception):
    def __init__(self, message="Error occurred during pet adoption."):
        super().__init__(message)
```

```
=== Adopter Menu ===
1. View Available Pets
2. Adopt a Pet
3. Make a Donation
4. Register for Adoption Event
5. View Adoption Events
0. Back to Main Menu
Enter choice: 2
Enter Pet ID to adopt: 20
Enter Your User ID: 5
Adoption Error: Pet with ID 20 does not exist.
```

7. Database Connectivity:

```python
# util/db_conn_util.py

import mysql.connector
from util.db_property_util import load_db_properties

def get_connection():
    props = load_db_properties("resources/db.properties")
    try:
        conn = mysql.connector.connect(
            host=props.get("host"),
            user=props.get("user"),
            password=props.get("password"),
            database=props.get("database")
        )
        return conn
    except mysql.connector.Error as err:
        raise Exception(f"Database connection failed: {err}")
```

Create and implement the following tasks in your application.

• Displaying Pet Listings:

o Develop a program that connects to the database and retrieves a list of available pets from the "pets" table. Display this list to the user. Ensure that the program handles database connectivity exceptions gracefully, including cases where the database is unreachable.

```
1. Adopter Menu
2. Shelter Menu
0. Exit
Enter choice: 1

=== Adopter Menu ===
1. View Available Pets
2. Adopt a Pet
3. Make a Donation
4. Register for Adoption Event
5. View Adoption Events
0. Back to Main Menu
Enter choice: 1
[1] Oliver (Dog) - Labrador Retriever, Age: 2, Available: True
[3] Bimbo (Cat) - Persian, Age: 1, Available: True
[4] Tusky (Dog) - Beagle, Age: 4, Available: True
[6] Lulu (Cat) - Ragdoll, Age: 2, Available: True
[7] Tiger (Dog) - Pug, Age: 3, Available: True
[8] Meeko (Cat) - British Shorthair, Age: 2, Available: True
[9] Caser (Dog) - Doberman, Age: 4, Available: True
[10] Mochi (Cat) - Himalayan, Age: 1, Available: True
[11] Tommy (Dog) - Golden Retriver, Age: 2, Available: True
```

• Donation Recording:

o Create a program that records cash donations made by donors. Allow the user to input donor information and the donation amount, and insert this data into the "donations" table in the database. Handle exceptions related to database operations, such as database errors or invalid inputs.

Mithali Shah donated cash of Rs. 5000/-

```
PS C:\Users\Aishwarya\OneDrive\Desktop\hexaware\ASSIGNMENT 1\PetPals> python -m main.main_module

=== PetPals Platform ===
1. Adopter Menu
2. Shelter Menu
0. Exit
Enter choice: 1

=== Adopter Menu ===
1. View Available Pets
2. Adopt a Pet
3. Make a Donation
4. Register for Adoption Event
5. View Adoption Events
0. Back to Main Menu
Enter choice: 3
Enter your name: Mitali Shah
Enter donation type (Cash/Item): Cash
Enter Shelter ID: 8
Enter donation amount: 5000.00
✅ Cash donation recorded.
```

Mithali Shah also donated few items to the shelter:

```
=== Adopter Menu ===
1. View Available Pets
2. Adopt a Pet
3. Make a Donation
4. Register for Adoption Event
5. View Adoption Events
0. Back to Main Menu
Enter choice: 3
Enter your name: Mitali Shah
Enter donation type (Cash/Item): Item
Enter Shelter ID: 4
Enter item description: Pet Shampoo and Food
✅ Item donation recorded.
```

Field recorded successfully in the database:

```
mysql> select * from donations;
+------------+--------------+--------------+----------------+---------------------+---------------------+-----------+
| donationid | donorname    | donationtype | donationamount | itemdescription     | donationdate        | shelterid |
+------------+--------------+--------------+----------------+---------------------+---------------------+-----------+
|          1 | Neha Gupta   | Cash         |        1500.00 | NULL                | 2025-06-01 10:00:00 |         1 |
|          2 | Rohan Das    | Item         |           NULL | Cat Food Pack       | 2025-06-02 12:00:00 |         2 |
|          3 | Tanya Mehta  | Cash         |        2000.00 | NULL                | 2025-06-03 11:30:00 |         3 |
|          4 | Amit Sinha   | Item         |           NULL | Dog Leash Set       | 2025-06-04 14:00:00 |         4 |
|          5 | Priya Anand  | Cash         |        1000.00 | NULL                | 2025-06-05 09:45:00 |         5 |
|          6 | Manav Bhatia | Item         |           NULL | Pet Beds            | 2025-06-06 16:00:00 |         6 |
|          7 | Sakshi Yadav | Cash         |        2500.00 | NULL                | 2025-06-07 13:30:00 |         7 |
|          8 | Deepak Rao   | Item         |           NULL | Vaccination Kits    | 2025-06-08 10:15:00 |         8 |
|          9 | Kriti Jain   | Cash         |        1800.00 | NULL                | 2025-06-09 15:20:00 |         9 |
|         10 | Jay Mehra    | Item         |           NULL | Pet Shampoo         | 2025-06-10 12:10:00 |        10 |
|         11 | Mitali Shah  | Cash         |        5000.00 | NULL                | 2025-06-27 12:21:33 |         8 |
|         12 | Mitali Shah  | Item         |           NULL | Pet Shampoo and Food | 2025-06-27 12:21:51 |         4 |
+------------+--------------+--------------+----------------+---------------------+---------------------+-----------+
12 rows in set (0.01 sec)
```

EXCEPTION IS TRIGGERED WHEN THE DONATION AMOUNT IS LESS THAN Rs. 500/-

```
=== PetPals Platform ===
1. Adopter Menu
2. Shelter Menu
0. Exit
Enter choice: 1

=== Adopter Menu ===
1. View Available Pets
2. Adopt a Pet
3. Make a Donation
4. Register for Adoption Event
5. View Adoption Events
0. Back to Main Menu
Enter choice: 3
Enter your name: Emily
Enter donation type (Cash/Item): Cash
Enter Shelter ID: 4
Enter donation amount: 400
Error: Minimum donation is 500
```

• Adoption Event Management:

o Build a program that connects to the database and retrieves information about upcoming adoption events from the "adoption_events" table. Allow the user to register for an event by adding their details to the "participants" table. Ensure that the program handles database connectivity and insertion exceptions properly.

VIEW ADOPTION EVENTS BY THE USER;

```
=== Adopter Menu ===
1. View Available Pets
2. Adopt a Pet
3. Make a Donation
4. Register for Adoption Event
5. View Adoption Events
0. Back to Main Menu
Enter choice: 5
1: Tails of Joy Expo on 2025-07-10 11:00:00 at Chennai Community Center
2: Whisker Wonderland on 2025-07-15 14:00:00 at Mumbai Expo Grounds
3: Pawsitive Vibes Fair on 2025-07-20 10:00:00 at Delhi Canopy Hall
4: Snuggle Fest 2025 on 2025-07-25 09:30:00 at Bangalore Central Park
5: Tailspin Carnival on 2025-08-01 12:00:00 at Hyderabad Pet Square
6: Bark & Bond Bash on 2025-08-05 13:00:00 at Pune Stadium Arena
7: Furball Fiesta on 2025-08-10 15:30:00 at Jaipur City Hall
8: The Adoption Junction on 2025-08-15 11:45:00 at Kolkata Riverside Walk
9: Meow Meet 2025 on 2025-08-20 10:15:00 at Ahmedabad Pet Ground
10: Cuddle & Care Carnival on 2025-08-25 14:45:00 at Lucknow Adoption Pavilion
```

REGISTERING FOR THE EVENT IS SUCCESSFUL

```
=== Adopter Menu ===
1. View Available Pets
2. Adopt a Pet
3. Make a Donation
4. Register for Adoption Event
5. View Adoption Events
0. Back to Main Menu
Enter choice: 4
Enter your name: Bhavna Shasha
Enter Event ID to register: 7
✅ Participant registered successfully.
```

REFLECTION TO THE DATABASE IS SUCCESSFUL;

```
mysql> select * from participants;
+---------------+----------------------+-----------------+---------+
| participantid | participantname      | participanttype | eventid |
+---------------+----------------------+-----------------+---------+
|             1 | Silver Whiskers Haven | Shelter         |       1 |
|             2 | Happy Pawstice       | Shelter         |       2 |
|             3 | Naina Verma          | Adopter         |       1 |
|             4 | Yashwant Rao         | Adopter         |       2 |
|             5 | Canine Cosmos        | Shelter         |       3 |
|             6 | Karan Joshi          | Adopter         |       3 |
|             7 | Meow Manor           | Shelter         |       4 |
|             8 | Sneha Patil          | Adopter         |       4 |
|             9 | Barkshire Retreat    | Shelter         |       5 |
|            10 | Diya Malhotra        | Adopter         |       5 |
|            11 | Aanya Bedi           | Adopter         |       3 |
|            12 | Laura Jacob          | Adopter         |       9 |
|            13 | Meow Manor           | Shelter         |       9 |
|            14 | Bhavna Shasha        | Adopter         |       7 |
+---------------+----------------------+-----------------+---------+
14 rows in set (0.03 sec)
```

MY BUSINESS LOGIC ADDITION :

- As part of improving user engagement and community involvement, I implemented a custom Donation Leaderboard feature.

- This ranks the top 3 donors based on total cash contributions and displays their names and donation amounts.

- The goal was to motivate donors by recognising their generosity and encouraging more contributions through friendly visibility. This logic adds value by making the donation system more interactive and impactful.
- This functionality uses SQL aggregation functions like SUM() to calculate the total cash donations by each donor, GROUP BY to group records by donor name, and ORDER BY to rank them in descending order. The query also uses LIMIT 3 to display only the top 3 contributors.

```
=== Adopter Menu ===
1. View Available Pets
2. Adopt a Pet
3. Make a Donation
4. Register for Adoption Event
5. View Adoption Events
6. Donation Leaderboard
0. Back to Main Menu
Enter choice: 6

--- Donation Leaderboard (Top Cash Donors) ---
1. Mitali Shah - ₹5000.00
2. Sakshi Yadav - ₹2500.00
3. Tanya Mehta - ₹2000.00
```