Mini project report on

PET ADOPTION MANAGEMENT SYSTEM

Submitted in partial fulfilment of the requirements for the award of degree of

Bachelor of Technology in Computer Science & Engineering UE23CS351A – DBMS Project

Submitted by:

Aishwarya	**PES2UG23AM008**
Deepika N	**PES2UG23AM028**

under the guidance of

Dr. Geetha D

Associate Professor

PES University

AUG - DEC 2025

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

FACULTY OF ENGINEERING

PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)

Electronic City, Hosur Road, Bengaluru – 560 100, Karnataka, India

(

DECLARATION

We hereby declare that the DBMS Project entitled **Pet Adoption Management System** has been carried out by us under the guidance of **Dr Geetha D,** and submitted in partial fulfilment of the course requirements for the award of degree of **Bachelor of Technology** in **Computer Science and Engineering** of **PES University, Bengaluru** during the academic semester AUG – DEC 2025.

Aishwarya	**PES2UG23AM008**
Deepika N	**PES2UG23AM028**

ACKNOWLEDGEMENT

I would like to express my gratitude to Dr. Geetha D, Department of Computer Science and Engineering, PES University, for her continuous guidance, assistance, and encouragement throughout the development of this UE23CS351A - DBMS Project.

I am deeply grateful to Dr. M. R. Doreswamy, Chancellor, PES University, Prof. Jawahar Doreswamy, Pro Chancellor – PES University, Dr. Suryaprasad J, Vice Chancellor, PES University for providing to me various opportunities and enlightenment every step of the way. Finally, this DBMS Project could not have been completed without the continual support and encouragement I have received from my family and friends.

ABSTRACT

The Pet Adoption Management System is a database-driven application designed to simplify and digitize the process of managing pet adoptions in animal shelters. The system maintains detailed information about pets, adopters, staff, shelters, medical records, and adoption activities in a structured and efficient manner. It provides a user-friendly Graphical User Interface (GUI) built using Python's Tkinter and connected to a MySQL database, enabling easy data entry, update, and retrieval. The project ensures accurate tracking of each pet's status

— whether available or adopted — and maintains a record of which adopter has adopted which pet.

Through features such as automated adoption status updates, dynamic record display, and easy-to-use CRUD (Create, Read, Update, Delete) operations, the system minimizes manual errors and paperwork. It also includes triggers, stored procedures, and functions to automate backend tasks like updating medical records or marking pets as adopted. The overall objective of the system is to create a reliable, efficient, and scalable digital platform for animal shelters to manage adoption records, improve transparency, and encourage responsible pet adoption.

TABLE OF CONTENTS

TITLE: PET ADOPTION MANAGEMENT SYSTEM

Team details:

SRN	NAME	SEC
PES2UG23AM008	AISHWARYA	A
PES2UG23AM028	DEEPIKA N	A

1) Introduction:

The **Pet Adoption Management System** is a computerized application developed to simplify and organize the operations of pet adoption centers. It replaces traditional paper-based record-keeping with a reliable and efficient digital database system.

The project is implemented using **Python (Tkinter)** for the GUI and **MySQL** for database management.

It stores and manages data related to **pets, adopters, shelters, medical records, and staff members**.

The system allows users to easily add, update, delete, and view records through a simple user interface.

It also includes automated backend features such as **triggers, stored procedures, and functions**.

When an adopter adopts a pet, the pet's status is automatically updated to "Adopted."

This ensures accurate and real-time tracking of available and adopted pets.

The system reduces human errors, saves time, and enhances efficiency in data handling.

Overall, it provides a modern and reliable platform for managing pet adoption processes effectively.

2. Scope of the Project

The scope of this project includes designing and developing a database system that can handle operations related to pet management, adopter registration, adoption tracking, and medical history maintenance.
The system provides features for:

- Registering new pets and adopters.
- Recording details of adoptions, including adopter and pet mapping.
- Managing medical records for pets (vaccinations, treatments, and checkups).
- Keeping track of shelters, their capacity, and assigned staff.
- Managing visitor information and their interaction with pets.

This system can be used by *animal shelters, NGOs, veterinary clinics,* and *government adoption centers* to streamline their operations. In the future, it can be integrated with a web or mobile interface for easy public access.

3. Detailed Description

The *Pet Adoption Database* manages all entities and relationships involved in the pet adoption process.
The system contains multiple entities such as:

- **Pet:** Stores details like pet ID, name, species, breed, age, and gender. Each pet is associated with a particular shelter.
- **Adopter:** Stores adopter details such as name, contact number, email, and address.
- **Shelter:** Contains information about shelters like shelter ID, name, location, and capacity.
- **Staff:** Maintains details of staff members working in shelters, including their roles and assigned shelters.
- **Medical_Record:** Keeps track of each pet's health information, including vaccinations, treatments, and last checkup date.
- **Adopts:** Represents the relationship between *Adopter* and *Pet*, recording which adopter adopted which pet, along with the adoption date.
- **Visits:** Records information about potential adopters visiting shelters and interacting with pets before adoption.

These tables are linked using *primary and foreign key relationships* ensuring referential integrity.
The database supports *insertion, updation, deletion,* and *retrieval* of information efficiently.
Triggers, stored procedures, and functions are used for automation—such as updating pet availability status after adoption or recording a medical entry automatically.

4. Functional Requirements System Functionality 1: Manage Pet Records

- The system allows adding, updating, and deleting pet details.
- Each pet record includes attributes like name, species, age, breed, and associated shelter.

System Functionality 2: Manage Adopter Information

- The system maintains adopter data such as contact details and addresses.
- Allows viewing adopter history and adoption count.

System Functionality 3: Handle Adoptions

- Records each adoption linking a pet to an adopter with date and status.
- Automatically updates the pet's availability once adopted.

System Functionality 4: Manage Shelters

- Keeps details of all shelters including capacity and location.
- Links shelters to staff and pets, ensuring capacity constraints.

System Functionality 5: Staff Management

- Stores staff data like name, role, and assigned shelter.
- Helps in tracking which staff is responsible for a given shelter.

System Functionality 6: Medical Record Management

- Stores vaccination and treatment details for pets.
- Supports health tracking with vaccination type and last checkup date.

System Functionality 7: Visitor Tracking

- Records visitors' information who come to shelters.
- Useful for analyzing visitor interest and adoption conversion rate.

System Functionality 8: Data Retrieval & Reporting

- Allows generating queries to get detailed reports — such as pets available for adoption, adopters by city, shelter occupancy, or vaccination status.

System Functionality 9: Triggers and Automation

- Uses triggers to auto-update pet availability when adopted.
- Automatically logs timestamps or alerts for medical updates.

System Functionality 10: Security & Integrity

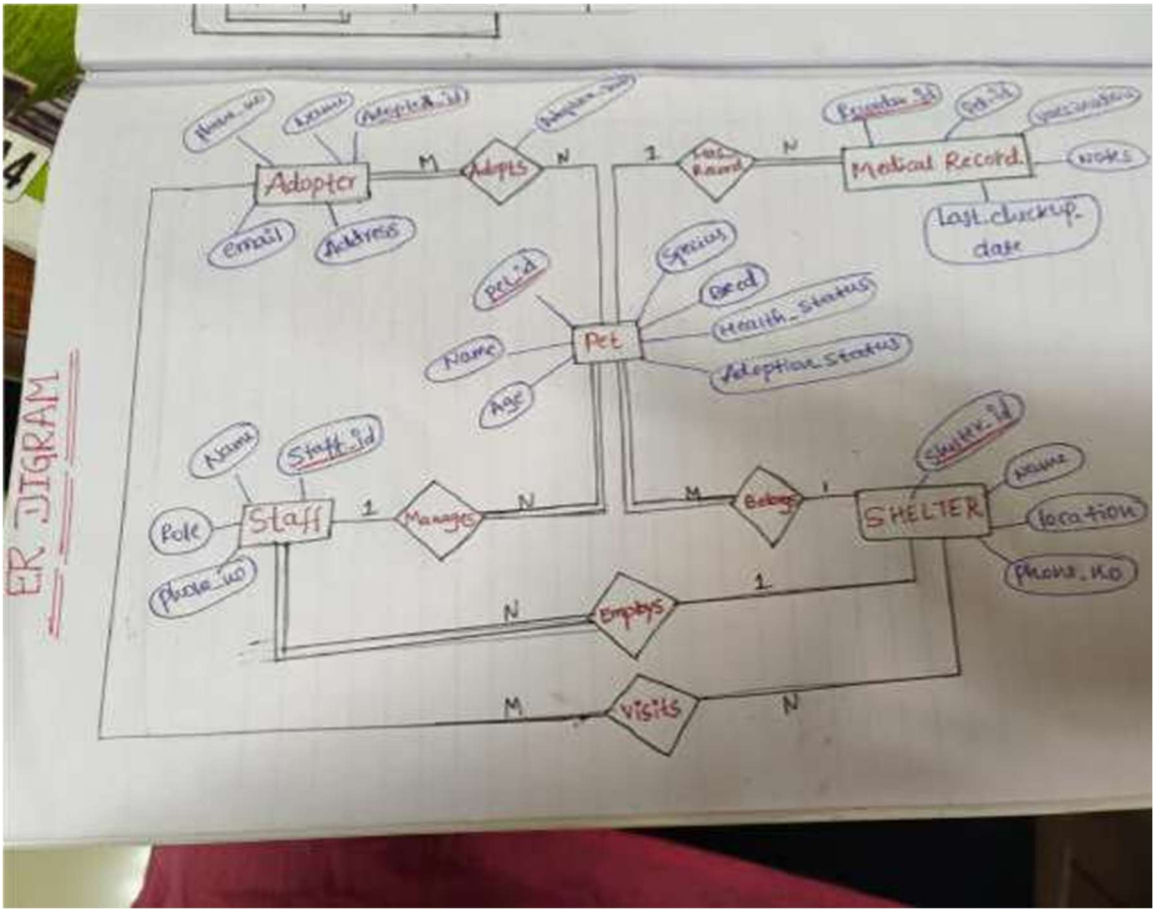
- Enforces constraints like primary keys, foreign keys, and not-null conditions.
- Ensures data accuracy and prevents duplicate entries.

List of Softwares / Tools / Programming Languages Used

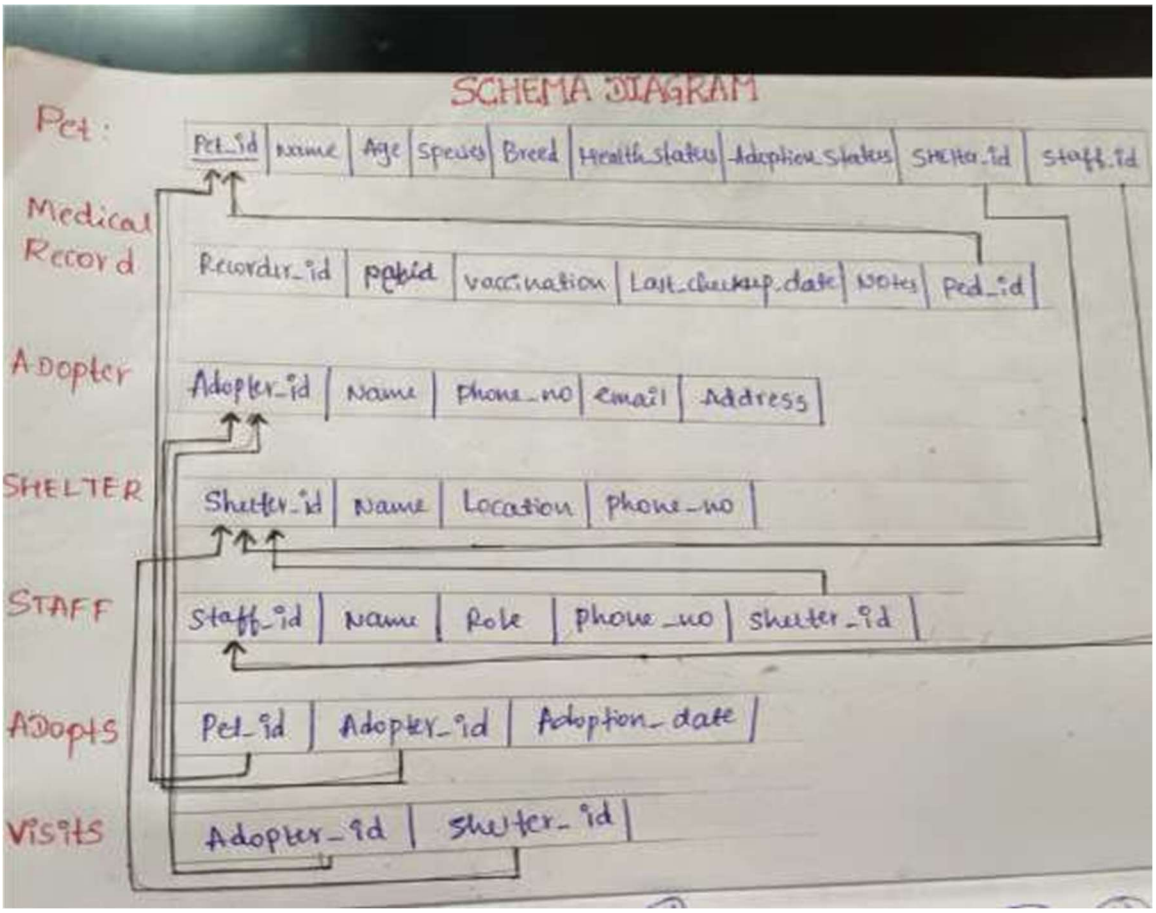
1. Programming Language: Python
2. Database: MySQL Server

3. GUI Library: Tkinter (for graphical user interface)
4. IDE / Code Editor: Visual Studio Code (VS Code)
5. Command-Line Tool: MySQL Command Line Client / Terminal
6. Database Connector: mysql-connector-python
   (Python library for connecting MySQL with Python)

ER Diagram:



Relational Schema



DDL COMMANDS:

-- **Create Adopter table** CREATE TABLE adopter (
adopter_id INT AUTO_INCREMENT PRIMARY KEY,
name VARCHAR(100) NOT NULL, phone_no
VARCHAR(15), email VARCHAR(100),
address VARCHAR(255)
);

-- **Create Shelter table** CREATE TABLE shelter (
shelter_id INT AUTO_INCREMENT PRIMARY KEY,
name VARCHAR(100) NOT NULL, location VARCHAR(255),
capacity INT
);

-- **Create Pet table** CREATE TABLE pet (pet_id INT
AUTO_INCREMENT PRIMARY KEY, name
VARCHAR(100) NOT NULL,

```
   species VARCHAR(50),
breed VARCHAR(50),     age
INT,     gender
VARCHAR(10),     shelter_id
INT,
   FOREIGN KEY (shelter_id) REFERENCES shelter(shelter_id)
);

-- Create Staff table CREATE TABLE
staff (
   staff_id INT AUTO_INCREMENT PRIMARY KEY,     name
 VARCHAR(100) NOT NULL,
role     VARCHAR(50),
phone_no
         VARCHAR(15),
shelter_id INT,
   FOREIGN KEY (shelter_id) REFERENCES shelter(shelter_id)
);

-- Create Medical Record table CREATE TABLE medical_record
(     record_id INT AUTO_INCREMENT
PRIMARY KEY,     pet_id INT,     vaccination_status
VARCHAR(100),     last_checkup DATE,      notes
TEXT,
   FOREIGN KEY (pet_id) REFERENCES pet(pet_id)
);

-- Create Adoption table CREATE TABLE
adoption (
   adoption_id INT AUTO_INCREMENT PRIMARY KEY,
   adopter_id       INT,
pet_id               INT,
adoption_date DATE,
   FOREIGN KEY (adopter_id) REFERENCES adopter(adopter_id),
   FOREIGN KEY (pet_id) REFERENCES pet(pet_id)
);

-- Create Visits table (adopter visiting pets before adoption)  CREATE
TABLE visits (
   visit_id INT AUTO_INCREMENT PRIMARY KEY,
adopter_id INT,     pet_id INT,     visit_date
DATE,     notes TEXT,
```

FOREIGN KEY (adopter_id) REFERENCES adopter(adopter_id), FOREIGN KEY (pet_id) REFERENCES pet(pet_id)
);

DML COMMANDS

-- Insert into Shelter

```
INSERT INTO shelter (name, location, capacity) VALUES
('Happy Paws Shelter', 'Bangalore', 50),
('Safe Haven Shelter', 'Mysore', 30);
```

-- Insert into Adopter

```
INSERT INTO adopter (name, phone, email, address) VALUES
('Ramesh Kumar', '9876543210', 'ramesh@example.com', 'Bangalore'),
('Sita Devi', '9123456780', 'sita@example.com', 'Mysore');
```

-- Insert into Pet

```
INSERT INTO pet (name, species, breed, age, gender, shelter_id) VALUES
('Tommy', 'Dog', 'Labrador', 3, 'Male', 1),
('Mittu', 'Bird', 'Parrot', 2, 'Female', 2),
('Kitty', 'Cat', 'Persian', 1, 'Female', 1);
```

-- Insert into Staff

```
INSERT INTO staff (name, role, phone, shelter_id) VALUES
('Arjun Rao', 'Manager', '9988776655', 1),
('Priya Sharma', 'Vet', '9876501234', 2);
```

-- Insert into Medical Record

```
INSERT INTO medical_record (pet_id, vaccination_status, last_checkup, notes)
VALUES
(1, 'Vaccinated', '2025-08-01', 'Healthy and active'),
(2, 'Not Vaccinated', '2025-07-15', 'Needs vaccination'),
(3, 'Vaccinated', '2025-09-10', 'Minor skin allergy');
```

-- Insert into Visits

```
INSERT INTO visits (adopter_id, pet_id, visit_date, notes) VALUES
(1, 1, '2025-08-20', 'Adopter liked the pet'),
(2, 3, '2025-08-25', 'Adopter considering adoption');
```

-- Insert into Adoption

```
INSERT INTO adoption (adopter_id, pet_id, adoption_date) VALUES
(1, 1, '2025-09-01'), (2,
3, '2025-09-05');
```

DML

AND CRUD
OPERATION:

```
mysql> INSERT INTO adopter (name, phone_no, email, address)
    -> VALUES ('Riya Sharma', 9876543210, 'riya@example.com', 'Bangalore');
Query OK, 1 row affected (0.01 sec)

mysql> SELECT * FROM adopter;
+------------+-----------------+------------+-----------------------+-----------------------+
| adopter_id | name            | phone_no   | email                 | address               |
+------------+-----------------+------------+-----------------------+-----------------------+
|          1 | Renuka          | 9123456789 | renuka@gmail.com      | Bangalore             |
|          2 | Adarsha         | 9876543210 | adarsha@gmail.com     | Delhi                 |
|          3 | dineshwari      | 9123456789 | dineshwari@gmail.com  | Bangalore             |
|          4 | Renuka Hosamani | 9876543210 | renuka@gmail.com      | Hubballi, Karnataka   |
|          5 | Riya Sharma     | 9812345678 | riya@gmail.com        | Bengaluru, Karnataka  |
|          6 | Aarav Patil     | 9123456789 | aarav@gmail.com       | Belagavi, Karnataka   |
|          7 | Priya Nair      | 9098765432 | priya@gmail.com       | Mysuru, Karnataka     |
|          8 | Kiran Desai     | 9988776655 | kiran@gmail.com       | Mangaluru, Karnataka  |
|          9 | ruhan           | 8985673456 | ruhan@pes.com         | bangalore             |
|         10 | Riya Sharma     | 9876543210 | riya@example.com      | Bangalore             |
+------------+-----------------+------------+-----------------------+-----------------------+
10 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM adopter;
+------------+-----------------+------------+-----------------------+-----------------------+
| adopter_id | name            | phone_no   | email                 | address               |
+------------+-----------------+------------+-----------------------+-----------------------+
|          1 | Renuka          | 9123456789 | renuka@gmail.com      | Bangalore             |
|          2 | Adarsha         | 9876543210 | adarsha@gmail.com     | Delhi                 |
|          3 | dineshwari      | 9123456789 | dineshwari@gmail.com  | Bangalore             |
|          4 | Renuka Hosamani | 9876543210 | renuka@gmail.com      | Hubballi, Karnataka   |
|          5 | Riya Sharma     | 9812345678 | riya@gmail.com        | Bengaluru, Karnataka  |
|          6 | Aarav Patil     | 9123456789 | aarav@gmail.com       | Belagavi, Karnataka   |
|          7 | Priya Nair      | 9098765432 | priya@gmail.com       | Mysuru, Karnataka     |
|          8 | Kiran Desai     | 9988776655 | kiran@gmail.com       | Mangaluru, Karnataka  |
|          9 | ruhan           | 8985673456 | ruhan@pes.com         | bangalore             |
|         10 | Riya Sharma     | 9876543210 | riya@example.com      | Bangalore             |
+------------+-----------------+------------+-----------------------+-----------------------+
10 rows in set (0.00 sec)

mysql> UPDATE adopter
    -> SET phone_no = 9112233445,
    ->     email = 'dineshwari_new@gmail.com',
    ->     address = 'Mysore, Karnataka'
    -> WHERE adopter_id = 3;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM adopter;
+------------+-----------------+------------+-----------------------+-----------------------+
| adopter_id | name            | phone_no   | email                 | address               |
+------------+-----------------+------------+-----------------------+-----------------------+
|          1 | Renuka          | 9123456789 | renuka@gmail.com      | Bangalore             |
|          2 | Adarsha         | 9876543210 | adarsha@gmail.com     | Delhi                 |
|          3 | dineshwari      | 9112233445 | dineshwari_new@gmail.com | Mysore, Karnataka  |
|          4 | Renuka Hosamani | 9876543210 | renuka@gmail.com      | Hubballi, Karnataka   |
|          5 | Riya Sharma     | 9812345678 | riya@gmail.com        | Bengaluru, Karnataka  |
|          6 | Aarav Patil     | 9123456789 | aarav@gmail.com       | Belagavi, Karnataka   |
|          7 | Priya Nair      | 9098765432 | priya@gmail.com       | Mysuru, Karnataka     |
|          8 | Kiran Desai     | 9988776655 | kiran@gmail.com       | Mangaluru, Karnataka  |
|          9 | ruhan           | 8985673456 | ruhan@pes.com         | bangalore             |
|         10 | Riya Sharma     | 9876543210 | riya@example.com      | Bangalore             |
+------------+-----------------+------------+-----------------------+-----------------------+
10 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM adopter;
+------------+-----------------+------------+-------------------------+-------------------+
| adopter_id | name            | phone_no   | email                   | address           |
+------------+-----------------+------------+-------------------------+-------------------+
|          1 | Renuka          | 9123456789 | renuka@gmail.com        | Bangalore         |
|          2 | Adarsha         | 9876543210 | adarsha@gmail.com       | Delhi             |
|          3 | dineshwari      | 9112233445 | dineshwari_new@gmail.com| Mysore, Karnataka |
|          4 | Renuka Hosamani | 9876543210 | renuka@gmail.com        | Hubballi, Karnataka |
|          5 | Riya Sharma     | 9812345678 | riya@gmail.com          | Bengaluru, Karnataka |
|          6 | Aarav Patil     | 9123456789 | aarav@gmail.com         | Belagavi, Karnataka |
|          7 | Priya Nair      | 9098765432 | priya@gmail.com         | Mysuru, Karnataka |
|          8 | Kiran Desai     | 9988776655 | kiran@gmail.com         | Mangaluru, Karnataka |
|          9 | ruhan           | 8905673456 | ruhan@pes.com           | bangalore         |
|         10 | Riya Sharma     | 9876543210 | riya@example.com        | Bangalore         |
+------------+-----------------+------------+-------------------------+-------------------+
10 rows in set (0.00 sec)

mysql> DELETE FROM adopter
    -> WHERE adopter_id = 9;
Query OK, 1 row affected (0.01 sec)

mysql> SELECT * FROM adopter;
+------------+-----------------+------------+-------------------------+-------------------+
| adopter_id | name            | phone_no   | email                   | address           |
+------------+-----------------+------------+-------------------------+-------------------+
|          1 | Renuka          | 9123456789 | renuka@gmail.com        | Bangalore         |
|          2 | Adarsha         | 9876543210 | adarsha@gmail.com       | Delhi             |
|          3 | dineshwari      | 9112233445 | dineshwari_new@gmail.com| Mysore, Karnataka |
|          4 | Renuka Hosamani | 9876543210 | renuka@gmail.com        | Hubballi, Karnataka |
|          5 | Riya Sharma     | 9812345678 | riya@gmail.com          | Bengaluru, Karnataka |
|          6 | Aarav Patil     | 9123456789 | aarav@gmail.com         | Belagavi, Karnataka |
|          7 | Priya Nair      | 9098765432 | priya@gmail.com         | Mysuru, Karnataka |
|          8 | Kiran Desai     | 9988776655 | kiran@gmail.com         | Mangaluru, Karnataka |
|         10 | Riya Sharma     | 9876543210 | riya@example.com        | Bangalore         |
+------------+-----------------+------------+-------------------------+-------------------+
9 rows in set (0.00 sec)
```

Nested query:

```
mysql> SELECT name
    -> FROM pet
    -> WHERE pet_id IN (
    ->     SELECT pet_id
    ->     FROM adopts
    ->     WHERE adopter_id IN (
    ->         SELECT adopter_id
    ->         FROM adopter
    ->         WHERE address LIKE '%Bangalore%'
    ->     )
    -> );
+--------+
| name   |
+--------+
| Kitty |
+--------+
1 row in set (0.01 sec)
```

```
mysql> SELECT name
    -> FROM pet
    -> WHERE pet_id NOT IN (
    ->     SELECT pet_id
    ->     FROM adopts
    -> );
+--------+
| name   |
+--------+
| Milo   |
| Rocky  |
| dany   |
| Rocky  |
| gsdfsh |
| gsdfsh |
| gsdfsh |
+--------+
7 rows in set (0.00 sec)
```

```
mysql> SELECT name
    -> FROM pet
    -> WHERE pet_id IN (
    ->     SELECT pet_id
    ->     FROM medical_record
    ->     GROUP BY pet_id
    ->     HAVING COUNT(record_id) > 1
    -> );
Empty set (0.00 sec)

mysql> SELECT name
    -> FROM adopter
    -> WHERE adopter_id IN (
    ->     SELECT adopter_id
    ->     FROM adopts
    ->     WHERE pet_id IN (
    ->         SELECT pet_id
    ->         FROM pet
    ->         WHERE age < 2
    ->     )
    -> );
Empty set (0.00 sec)
```

```
mysql> SELECT name
    -> FROM shelter
    -> WHERE shelter_id IN (
    ->     SELECT shelter_id
    ->     FROM pet
    ->     GROUP BY shelter_id
    ->     HAVING COUNT(pet_id) = (
    ->         SELECT MAX(pet_count)
    ->         FROM (
    ->             SELECT COUNT(pet_id) AS pet_count
    ->             FROM pet
    ->             GROUP BY shelter_id
    ->         ) AS counts
    ->     )
    -> );
+--------------+
| name         |
+--------------+
| Green Shelter |
+--------------+
1 row in set (0.00 sec)

mysql>
```

AGRREGATE FUNCTIONS:

```
mysql> select*from pet;
+--------+---------+------+---------+----------------+---------------+----------------+------------+----------+
| pet_id | name    | age  | species | breed          | health_status | adoption_status | shelter_id | staff_id |
+--------+---------+------+---------+----------------+---------------+----------------+------------+----------+
|      1 | Buddy   |    3 | Dog     | Labrador       | Healthy       | Adopted        |          1 |        1 |
|      2 | Kitty   |    2 | Cat     | Persian        | Healthy       | Adopted        |          1 |        2 |
|      3 | Bruno   |    4 | Dog     | Beagle         | Injured       | Adopted        |          2 |        3 |
|      4 | Charlie |    2 | Dog     | Pug            | Healthy       | Adopted        |          1 |        2 |
|      5 | Milo    |    1 | Cat     | Siamese        | Healthy       | Available      |          2 |        1 |
|      6 | Rocky   |    5 | Dog     | Bulldog        | Injured       | Available      |          2 |        3 |
|      8 | dany    |   10 | cat     | persian        | helathy       | available      |          1 |        1 |
|      9 | Rocky   |    0 | NULL    | NULL           | NULL          | Available      |       NULL |     NULL |
|     10 | Rocky   |    2 | Dog     | German Shepherd | Healthy       | Adopted        |          1 |        2 |
|     11 | gsdfsh  |   54 | dsf     | szdf           | fgh           | Available      |          2 |     NULL |
|     12 | gsdfsh  |   54 | dsf     | szdf           | fgh           | Adopted        |          2 |     NULL |
|     13 | gsdfsh  |   54 | dsf     | szdf           | fgh           | Available      |          2 |     NULL |
+--------+---------+------+---------+----------------+---------------+----------------+------------+----------+
12 rows in set (0.00 sec)
```

```
mysql> SELECT shelter_id, COUNT(*) AS total_pets
    -> FROM pet
    -> GROUP BY shelter_id;
+------------+-------------+
| shelter_id | total_pets |
+------------+-------------+
|       NULL |           1 |
|          1 |           5 |
|          2 |           3 |
+------------+-------------+
3 rows in set (0.01 sec)
```

```
mysql> SELECT a.name AS adopter_name, COUNT(ad.pet_id) AS total_adopted
    -> FROM adopter a
    -> JOIN adopts ad ON a.adopter_id = ad.adopter_id
    -> GROUP BY a.adopter_id;
+----------------+---------------+
| adopter_name   | total_adopted |
+----------------+---------------+
| Adarsha        |             1 |
| Renuka         |             1 |
| dineshwari     |             1 |
| Renuka Hosamani |             1 |
| Aarav Patil    |             1 |
+----------------+---------------+
5 rows in set (0.01 sec)
```

```
mysql> SELECT species, ROUND(AVG(age), 2) AS average_age
    -> FROM pet
    -> GROUP BY species;
+---------+-------------+
| species | average_age |
+---------+-------------+
| Dog     |        3.20 |
| Cat     |        4.33 |
| NULL    |        0.00 |
+---------+-------------+
3 rows in set (0.01 sec)
```

Triggers:

```
mysql> CREATE TRIGGER before_pet_insert
    -> BEFORE INSERT ON pet
    -> FOR EACH ROW
    -> BEGIN
    ->     -- Check if the age is negative
    ->     IF NEW.age < 0 THEN
    ->         SET NEW.age = 0;   -- Correct the age
    ->     END IF;
    ->
    ->     -- Set default adoption status if not provided
    ->     IF NEW.adoption_status IS NULL THEN
    ->         SET NEW.adoption_status = 'Available';
    ->     END IF;
    -> END $$
Query OK, 0 rows affected (0.03 sec)

mysql>
mysql> DELIMITER ;
mysql> INSERT INTO pet (name, age)
    -> VALUES ('Rocky', -5);
Query OK, 1 row affected (0.02 sec)

mysql> select *from pet;
+--------+---------+------+---------+----------+---------------+-----------------+------------+----------+
| pet_id | name    | age  | species | breed    | health_status | adoption_status | shelter_id | staff_id |
+--------+---------+------+---------+----------+---------------+-----------------+------------+----------+
|      1 | Buddy   |    3 | Dog     | Labrador | Healthy       | Adopted         |          1 |        1 |
|      2 | Kitty   |    2 | Cat     | Persian  | Healthy       | Adopted         |          1 |        2 |
|      3 | Bruno   |    4 | Dog     | Beagle   | Injured       | Adopted         |          2 |        3 |
|      4 | Charlie |    2 | Dog     | Pug      | Healthy       | Adopted         |          1 |        2 |
|      5 | Milo    |    1 | Cat     | Siamese  | Healthy       | Available       |          2 |        1 |
|      6 | Rocky   |    5 | Dog     | Bulldog  | Injured       | Available       |          2 |        3 |
|      8 | dany    |   10 | cat     | persian  | helathy       | available       |          1 |        1 |
|      9 | Rocky   |    0 | NULL    | NULL     | NULL          | Available       |       NULL |     NULL |
+--------+---------+------+---------+----------+---------------+-----------------+------------+----------+
8 rows in set (0.00 sec)
```

```
mysql> DELIMITER $$
mysql>
mysql> CREATE TRIGGER add_medical_record_after_pet
    -> AFTER INSERT ON pet
    -> FOR EACH ROW
    -> BEGIN
    ->     INSERT INTO medical_record (pet_id, vaccination, notes, last_checkup_date)
    ->     VALUES (NEW.pet_id, 'Not vaccinated', 'Record created automatically', CURDATE());
    -> END $$
Query OK, 0 rows affected (0.02 sec)

mysql>
mysql> DELIMITER ;
mysql> INSERT INTO pet (name, age, species, breed, health_status, adoption_status, shelter_id, staff_id)
    -> VALUES ('Rocky', 2, 'Dog', 'German Shepherd', 'Healthy', 'Available', 1, 2);
Query OK, 1 row affected (0.02 sec)

mysql> select *from pet;
+--------+---------+------+---------+----------------+---------------+-----------------+------------+----------+
| pet_id | name    | age  | species | breed          | health_status | adoption_status | shelter_id | staff_id |
+--------+---------+------+---------+----------------+---------------+-----------------+------------+----------+
|      1 | Buddy   |    3 | Dog     | Labrador       | Healthy       | Adopted         |          1 |        1 |
|      2 | Kitty   |    2 | Cat     | Persian        | Healthy       | Adopted         |          1 |        2 |
|      3 | Bruno   |    4 | Dog     | Beagle         | Injured       | Adopted         |          2 |        3 |
|      4 | Charlie |    2 | Dog     | Pug            | Healthy       | Adopted         |          1 |        2 |
|      5 | Milo    |    1 | Cat     | Siamese        | Healthy       | Available       |          2 |        1 |
|      6 | Rocky   |    5 | Dog     | Bulldog        | Injured       | Available       |          2 |        3 |
|      8 | dany    |   10 | cat     | persian        | helathy       | available       |          1 |        1 |
|      9 | Rocky   |    0 | NULL    | NULL           | NULL          | Available       |       NULL |     NULL |
|     10 | Rocky   |    2 | Dog     | German Shepherd | Healthy       | Available       |          1 |        2 |
+--------+---------+------+---------+----------------+---------------+-----------------+------------+----------+
9 rows in set (0.00 sec)
```

```
mysql> DELIMITER $$
mysql>
mysql> CREATE TRIGGER update_pet_adoption_status
    -> AFTER INSERT ON adopts
    -> FOR EACH ROW
    -> BEGIN
    ->     UPDATE pet
    ->     SET adoption_status = 'Adopted'
    ->     WHERE pet_id = NEW.pet_id;
    -> END $$
ERROR 1359 (HY000): Trigger already exists
mysql>
mysql> DELIMITER ;
```

```
mysql> INSERT INTO adopts (adopter_id, pet_id, adoption_date)
    -> VALUES (6, 10, CURDATE());
Query OK, 1 row affected (0.01 sec)
```

STORED_PROCEDURE:

```
mysql> DELIMITER $$
mysql>
mysql> CREATE PROCEDURE update_staff_role(
    ->     IN s_id INT,
    ->     IN new_role VARCHAR(50)
    -> )
    -> BEGIN
    ->     UPDATE staff
    ->     SET role = new_role
    ->     WHERE staff_id = s_id;
    ->
    ->     SELECT CONCAT('✅ Staff ID ', s_id, ' role updated to "', new_role, '".') AS message;
    -> END $$
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql> CALL update_staff_role(3, 'Manager');
+---------------------------------------+
| message                               |
+---------------------------------------+
| ? Staff ID 3 role updated to "Manager". |
+---------------------------------------+
1 row in set (0.01 sec)

Query OK, 0 rows affected (0.02 sec)

mysql> select *from staff;
+----------+-------------+--------------+------------+------------+
| staff_id | name        | role         | phone_no   | shelter_id |
+----------+-------------+--------------+------------+------------+
|        1 | Arjun Rao   | Manager      | 9988771122 |          1 |
|        2 | Deepika     | Veterinarian | 9988772233 |          1 |
|        3 | Ramesh      | Manager      | 9988773344 |          2 |
|        4 | Ravi Kumar  | Caretaker    | 9876501234 |          1 |
|        5 | Priya Sharma | Manager     | 9876543210 |          2 |
+----------+-------------+--------------+------------+------------+
5 rows in set (0.00 sec)
```

```
mysql> DELIMITER $$
mysql>
mysql> CREATE PROCEDURE update_staff_role(
    ->     IN s_id INT,
    ->     IN new_role VARCHAR(50)
    -> )
    -> BEGIN
    ->     UPDATE staff
    ->     SET role = new_role
    ->     WHERE staff_id = s_id;
    ->
    ->     SELECT CONCAT('✅ Staff ID ', s_id, ' role updated to "', new_role, '".') AS message;
    -> END $$
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql> CALL update_staff_role(3, 'Manager');
+---------------------------------------+
| message                               |
+---------------------------------------+
| ? Staff ID 3 role updated to "Manager". |
+---------------------------------------+
1 row in set (0.01 sec)

Query OK, 0 rows affected (0.02 sec)

mysql> select *from staff;
+----------+--------------+--------------+------------+------------+
| staff_id | name         | role         | phone_no   | shelter_id |
+----------+--------------+--------------+------------+------------+
|        1 | Arjun Rao    | Manager      | 9988771122 |          1 |
|        2 | Deepika      | Veterinarian | 9988772233 |          1 |
|        3 | Ramesh       | Manager      | 9988773344 |          2 |
|        4 | Ravi Kumar   | Caretaker    | 9876501234 |          1 |
|        5 | Priya Sharma | Manager      | 9876543210 |          2 |
+----------+--------------+--------------+------------+------------+
5 rows in set (0.00 sec)
```

```
mysql> DELIMITER $$
mysql>
mysql> CREATE PROCEDURE delete_staff_member(IN s_id INT)
    -> BEGIN
    ->     DELETE FROM staff
    ->     WHERE staff_id = s_id;
    ->
    ->     SELECT CONCAT('🗑️ Staff ID ', s_id, ' deleted successfully!') AS message;
    -> END $$
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql> CALL delete_staff_member(4);
+-----------------------------------+
| message                           |
+-----------------------------------+
| ?? Staff ID 4 deleted successfully! |
+-----------------------------------+
1 row in set (0.02 sec)

Query OK, 0 rows affected (0.03 sec)

mysql> select *from staff;
+----------+--------------+--------------+------------+------------+
| staff_id | name         | role         | phone_no   | shelter_id |
+----------+--------------+--------------+------------+------------+
|        1 | Arjun Rao    | Manager      | 9988771122 |          1 |
|        2 | Deepika      | Veterinarian | 9988772233 |          1 |
|        3 | Ramesh       | Manager      | 9988773344 |          2 |
|        5 | Priya Sharma | Manager      | 9876543210 |          2 |
+----------+--------------+--------------+------------+------------+
4 rows in set (0.00 sec)
```

FUNCTIONS:

```
mysql> DELIMITER $$
mysql>
mysql> CREATE FUNCTION get_pet_name(p_pet_id INT)
    -> RETURNS VARCHAR(100)
    -> DETERMINISTIC
    -> BEGIN
    ->     DECLARE p_name VARCHAR(100);
    ->     SELECT name INTO p_name
    ->     FROM pet
    ->     WHERE pet_id = p_pet_id;
    ->     RETURN p_name;
    -> END $$
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql> SELECT get_pet_name(3);
+------------------+
| get_pet_name(3) |
+------------------+
| Bruno            |
+------------------+
1 row in set (0.01 sec)
```

```
mysql> DELIMITER $$
mysql>
mysql> CREATE FUNCTION days_since_last_checkup(p_pet_id INT)
    -> RETURNS INT
    -> DETERMINISTIC
    -> BEGIN
    ->     DECLARE days_diff INT;
    ->
    ->     SELECT DATEDIFF(CURDATE(), last_checkup_date)
    ->     INTO days_diff
    ->     FROM medical_record
    ->     WHERE pet_id = p_pet_id
    ->     ORDER BY last_checkup_date DESC
    ->     LIMIT 1;
    ->
    ->     RETURN days_diff;
    -> END $$
ERROR 1304 (42000): FUNCTION days_since_last_checkup already exists
mysql>
mysql> DELIMITER ;
mysql> SELECT days_since_last_checkup(2);
+-----------------------------+
| days_since_last_checkup(2) |
+-----------------------------+
|                          76 |
+-----------------------------+
1 row in set (0.00 sec)
```

```
mysql> DELIMITER $$
mysql>
mysql> CREATE FUNCTION get_shelter_location(p_shelter_id INT)
    -> RETURNS VARCHAR(100)
    -> DETERMINISTIC
    -> BEGIN
    ->     DECLARE s_location VARCHAR(100);
    ->     SELECT location INTO s_location
    ->     FROM shelter
    ->     WHERE shelter_id = p_shelter_id;
    ->     RETURN s_location;
    -> END $$
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql> SELECT get_shelter_location(1);
+--------------------------+
| get_shelter_location(1) |
+--------------------------+
| Bangalore                |
+--------------------------+
1 row in set (0.00 sec)
```

```
mysql> select*from pet;
+--------+---------+------+---------+-----------------+---------------+-----------------+------------+-----+
| pet_id | name    | age  | species | breed           | health_status | adoption_status | shelter_id | staf |
+--------+---------+------+---------+-----------------+---------------+-----------------+------------+-----+
|      1 | Buddy   |    3 | Dog     | Labrador        | Healthy       | Adopted         |          1 |     |
|      2 | Kitty   |    2 | Cat     | Persian         | Healthy       | Adopted         |          1 |     |
|      3 | Bruno   |    4 | Dog     | Beagle          | Injured       | Adopted         |          2 |     |
|      4 | Charlie |    2 | Dog     | Pug             | Healthy       | Adopted         |          1 |     |
|      5 | Milo    |    1 | Cat     | Siamese         | Healthy       | Available       |          2 |     |
|      6 | Rocky   |    5 | Dog     | Bulldog         | Injured       | Available       |          2 |     |
|      8 | dany    |   10 | cat     | persian         | helathy       | available       |          1 |     |
|      9 | Rocky   |    0 | NULL    | NULL            | NULL          | Available       |       NULL |     |
|     10 | Rocky   |    2 | Dog     | German Shepherd | Healthy       | Adopted         |          1 |     |
|     11 | gsdfsh  |   54 | dsf     | szdf            | fgh           | Available       |          2 |     |
|     12 | gsdfsh  |   54 | dsf     | szdf            | fgh           | Adopted         |          2 |     |
|     13 | gsdfsh  |   54 | dsf     | szdf            | fgh           | Available       |          2 |     |
+--------+---------+------+---------+-----------------+---------------+-----------------+------------+-----+
12 rows in set (0.00 sec)
```

FOR GUI:

pet_adoption_gui.py × | pet_adoption_gui1.py | ~$_review1.docx | 32_review1.dc

pet_adoption_gui.py > ...

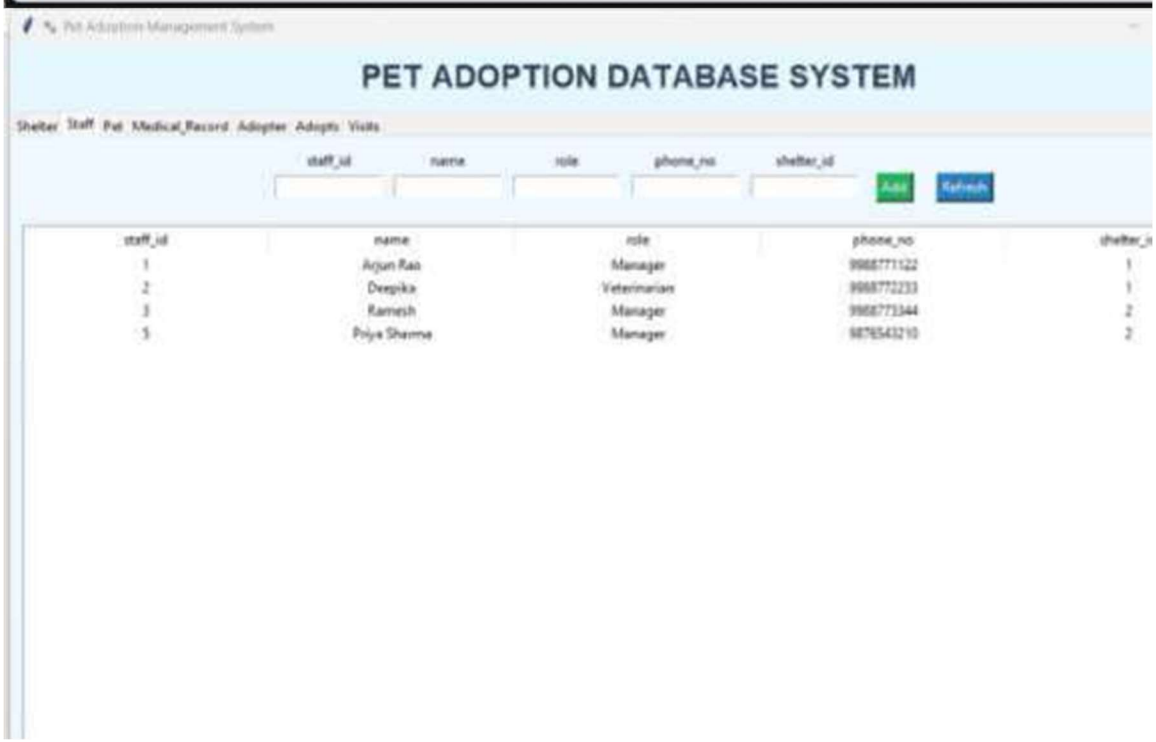
```python
import mysql.connector
from tkinter import *
from tkinter import ttk, messagebox

# ----------------------------------
# DB Connection
# ----------------------------------
def connect_db():
    return mysql.connector.connect(
        host="localhost",
        user="root",
        password="rk123",

        database="DB"  # use your DB name
    )

# ----------------------------------
# Generic Table Display Function
# ----------------------------------
def fetch_data(table, tree):
    for i in tree.get_children():
        tree.delete(i)
    try:
        con = connect_db()
        cur = con.cursor()
        cur.execute(f"SELECT * FROM {table}")
        rows = cur.fetchall()
        for row in rows:
            tree.insert("", END, values=row)
        con.close()
    except Exception as e:
        messagebox.showerror("Error", str(e))

```

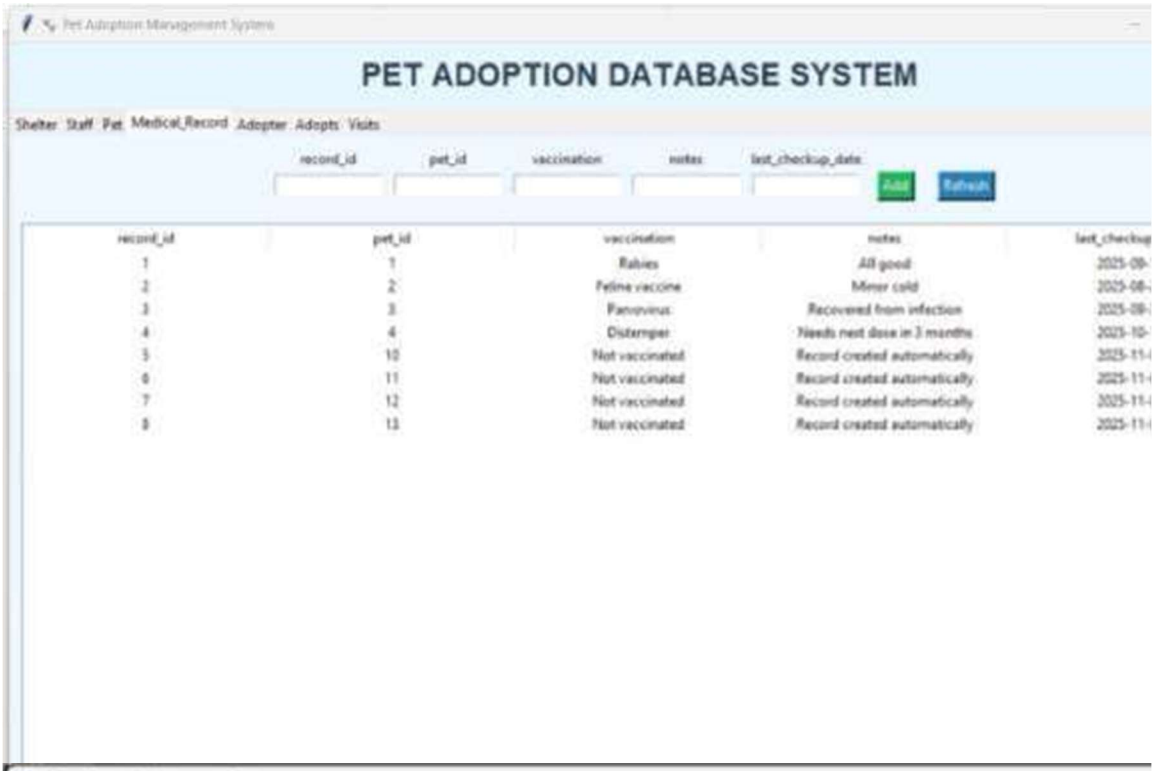
```python
# ----------------------------------
# Insert Function
# ----------------------------------
def insert_data(table, entries, tree):
    try:
        con = connect_db()
        cur = con.cursor()
        cols = ", ".join(entries.keys())
        values = tuple(e.get() for e in entries.values())
        placeholders = ", ".join(["%s"] * len(entries))
        cur.execute(f"INSERT INTO {table} ({cols}) VALUES ({placeholders})", values)
        con.commit()
        con.close()
        fetch_data(table, tree)
        messagebox.showinfo("Success", f"Record added to {table}!")
    except Exception as e:
        messagebox.showerror("Error", str(e))

# ----------------------------------
# GUI Setup
# ----------------------------------
root = Tk()
root.title("🐾 Pet Adoption Management System")
root.geometry("1100x700")
root.configure(bg="#e6f7ff")

Label(root, text="PET ADOPTION DATABASE SYSTEM", font=("Arial", 20, "bold"), bg="#e6f7ff", fg="#2c3e50").

# Notebook for tabs
tab_control = ttk.Notebook(root)
tab_control.pack(expand=1, fill="both")
```

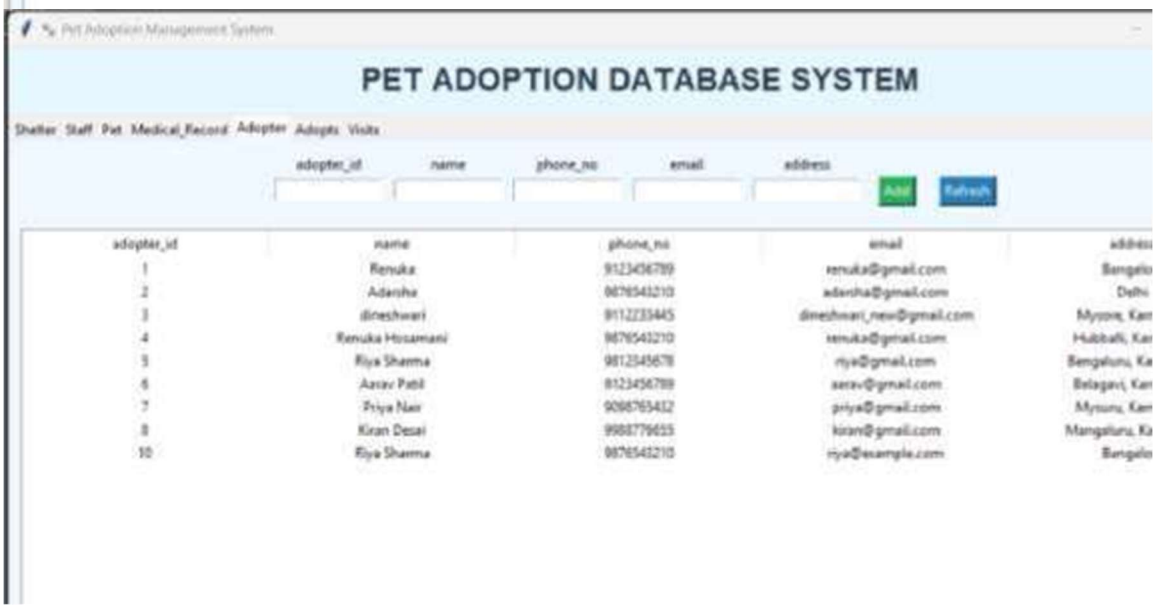
```python
# ----------------------------------
# Helper function to create a tab
# ----------------------------------
def create_tab(name, columns):
    tab = Frame(tab_control, bg="#f0f8ff")
    tab_control.add(tab, text=name)

    # Input frame
    form_frame = Frame(tab, bg="#f0f8ff")
    form_frame.pack(pady=10)

    entries = {}
    for idx, col in enumerate(columns):
        Label(form_frame, text=col, bg="#f0f8ff").grid(row=0, column=idx)
        e = Entry(form_frame, width=15)
        e.grid(row=1, column=idx, padx=5)
        entries[col] = e

    # Buttons
    Button(form_frame, text="Add", bg="#27ae60", fg="white",
           command=lambda: insert_data(name.lower(), entries, tree)).grid(row=1, column=len(columns), pa

    Button(form_frame, text="Refresh", bg="#298069", fg="white",
           command=lambda: fetch_data(name.lower(), tree)).grid(row=1, column=len(columns)+1, padx=10)

    # Treeview
    tree = ttk.Treeview(tab, columns=columns, show="headings")
    for col in columns:
        tree.heading(col, text=col)
        tree.column(col, width=150, anchor=CENTER)
    tree.pack(fill="both", expand=True, padx=10, pady=10)

    fetch_data(name.lower(), tree)
    return tab
```

```
# ---------------------------------
# Tabs for each entity
# ---------------------------------
create_tab("Shelter", ["shelter_id", "name", "location", "phone_no"])
create_tab("Staff", ["staff_id", "name", "role", "phone_no", "shelter_id"])
create_tab("Pet", ["pet_id", "name", "age", "species", "breed", "health_status", "adoption_status", "shelter_id",
create_tab("Medical_Record", ["record_id", "pet_id", "vaccination", "notes", "last_checkup_date"])
create_tab("Adopter", ["adopter_id", "name", "phone_no", "email", "address"])
create_tab("Adopts", ["adopter_id", "pet_id", "adoption_date"])
create_tab("Visits", ["visit_id", "staff_id", "shelter_id", "visit_date"])

root.mainloop()

```


Pet Adoption Management System
PET ADOPTION DATABASE SYSTEM
Shelter Staff Pet Medical_Record Adopter Adopts Visits
shelter_id
name
location
phone_no
Add
Refresh
shelter_id name location phone_no
1 City Shelter Bangalore 9988776655
2 Green Shelter Delhi 9876543210
3 aadiyogi chikkaballapur 9988776655
4 m benglore 1234567890


Pet Adoption Management System
PET ADOPTION DATABASE SYSTEM
Shelter Staff Pet Medical_Record Adopter Adopts Visits
staff_id
name
role
phone_no
shelter_id
Add
Refresh
staff_id name role phone_no
1 Arjun Rao Manager 9988771122
2 Deepika Veterinarian 9988772233
3 Ramesh Manager 9988773344
5 Priya Sharma Manager 9876543210


Pet Adoption Management System
PET ADOPTION DATABASE SYSTEM
Shelter Staff Pet Medical_Record Adopter Adopts Visits
record_id
pet_id
vaccination
notes
last_checkup_date
Add
Refresh
record_id pet_id vaccination notes
1 1 Rabies All good
2 2 Feline vaccine Minor cold
3 3 Parvovirus Recovered from infection
4 4 Distemper Needs next dose in 3 months
5 10 Not vaccinated Record created automatically
6 11 Not vaccinated Record created automatically
7 12 Not vaccinated Record created automatically
8 13 Not vaccinated Record created automatically


Pet Adoption Management System
PET ADOPTION DATABASE SYSTEM
Shelter Staff Pet Medical_Record Adopter Adopts Visits
adopter_id
name
phone_no
email
address
Add
Refresh
adopter_id name phone_no email
1 Renuka 9123456789 renuka@gmail.com
2 Adarsha 9876543210 adarsha@gmail.com
3 dineshwari 9112233445 dineshwari_new@gmail.com
4 Renuka Hosamani 9876543210 renuka@gmail.com
5 Riya Sharma 9812345678 riya@gmail.com
6 Aarav Patil 8123456789 aarav@gmail.com
7 Priya Nair 9088765432 priya@gmail.com
8 Kiran Desai 9988776655 kiran@gmail.com
10 Riya Sharma 9876543210 riya@example.com