PET ADOPTION MANAGEMENT SYSTEM

COURSE PROJECT REPORT

18CSC303J – DATABASE MANAGEMENT SYSTEMS

III Year/ VI Semester

Academic Year: 2023 -2024 (EVEN)

By

Shaik Mohammed Fahad (RA2111026010331)

Under the guidance of

Dr. M Dhilsath Fathima

Assistant Professor

Department of Computational Intelligence

DEPARTMENT OF COMPUTATIONAL INTELLIGENCE

FACULTY OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

Kattankulathur, Kancheepuram

APRIL 2024

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR - 603203

BONAFIDE CERTIFICATE

Certified that this project report titled "**PET ADOPTION MANAGEMENT SYSTEM**" is a bonafide work done by **SHAIK MOHAMMED FAHAD (RA2111026010331)** who carried out the project work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other project work or dissertation.

SIGNATURE
Faculty In-Charge
Dr. M.Dhilsath Fathima
Assistant Professor
Department of Computational Intelligence
SRM Institute of Science and Technology
Kattankulathur

SIGNATURE
Head of Department
Dr. R. Annie Uthra
Professor and Head
Department of Computational Intelligence
SRM Institute of Science and Technology
Kattankulathur

ABSTRACT

The Pet Adoption Management System (PAMS) serves as a comprehensive platform facilitating the adoption, distribution, and management of pets through online channels and networks. In recent years, the emergence of online platforms has transformed the way pet adoption operates, providing convenience and accessibility to potential adopters. PAMS aims to capitalize on this trend by offering a robust, flexible, and secure backend infrastructure to support its operations. As pet adoption becomes increasingly ingrained in modern life, the success and efficacy of an online system like PAMS hinge upon its backend architecture. It must possess qualities such as robustness, flexibility, security, efficiency, scalability, and reliability to meet the diverse needs of both users and administrators. Unlike traditional adoption processes, PAMS enables users to browse available pets, compare their profiles, and initiate adoption proceedings from the comfort of their homes using their internet-enabled devices.In today's digital age, the advancement of pet adoption through e-commerce platforms is crucial for societal development. Local shelters and adoption agencies face the challenge of adapting to digital platforms to remain competitive and relevant. By embracing digitalization and e-marketing strategies, these entities can elevate their services and reach a wider audience of potential adopters.

INDEX

CONTENTS		
S.no	**Particulars**	**Page no**
1.	INTRODUCTION	1
2.	PROJECT FEATURES AND OBJECTIVES	3
3.	BACK-END DESIGN, FRONT-END DESIGN AND CONNECTIVITY	6
4.	MODULE IMPLEMENTATION	6
5.	APPLICATIONS	23
6.	CONCLUSION	24
7.	BIBILOGRAPHY	25

LIST OF FIGURES

CONTENTS		
S.no	**Particulars**	**Page no**
3.1.1	ER DIAGRAM	6
4.1	TABLES LIST	8
4.1.1	PRODUCT TABLE	9
4.1.2	INBUILT FUNCTIONS	12
4.1.3	NESTED QUERIES	13
4.1.4	SET OPERATORS AND VIEWS	14
4.1.5	PROCEDURES AND FUNCTIONS	15
4.1.6	CURSORS AND EXCEPTION HANDLING	17
4.1.1	LOGIN	19
4.1.2	HOME PAGE	19
4.1.3	CHOOSE A PET	20
4.1.3	INSTRUCTION PAGE	20
4.1.4	APLICATION FORM FOR ADOPTION	21
4.1.5	CONTACT PAGE	21

CHAPTER I

INTRODUCTION

The Pet Adoption Management System (PAMS) serves as a pivotal tool in today's evolving pet adoption landscape, providing a seamless online platform for prospective pet owners to navigate through available pets, initiate adoption processes, and securely complete transactions. Leveraging advanced encryption and secure transaction protocols, PAMS ensures the safety and confidentiality of adoption transactions, while its robust backend infrastructure, powered by MySQL, facilitates efficient management of pet profiles, adoption requests, and shelter operations. Through intuitive frontend development using HTML, CSS, and JavaScript, PAMS offers a user-friendly interface, while its Entity-Relationship (ER) diagram and conceptual schema aid in visualizing and maintaining the database structure. With its broad applicability across various sectors, PAMS represents a comprehensive solution for managing the pet adoption process, empowering shelters and adoption agencies to streamline operations and enhance the overall adoption experience for both pets and adopters.

1.1 Software

VS CODE (FrontEnd)

Visual Studio Code, often abbreviated as VS Code, is a popular and versatile source-code editor developed by Microsoft. Launched in 2015, it quickly gained traction among developers due to its lightweight yet powerful features, extensive customization options, and wide support for various programming languages and frameworks. Visual Studio Code has emerged as a go-to choice for developers worldwide, offering a powerful and customizable development environment that caters to a wide range of programming languages and workflows. Its simplicity, extensibility, and robust feature set make it a favourite among developers for building applications of all types and sizes.

Database – MySQL

The back-end database used in this project is **MySQL.**

It is a language used to interrogate and process data in a relational database. Originally developed by IBM for its mainframes, SQL commands can be used to interactively work with a database or can be embedded within a script or programming language as an interface to a database. Programming extensions to SQL have turned it into a full-blown database programming language, and all major database management systems (DBMSs) support it.

ANSI standardized SQL.

But most DBMSs have some proprietary enhancement, which if used, makes SQL non-standard. Moving an application from one SQL database to another sometimes requires tweaking, the age-old problem in this business!

1.2 Advantages of MySQL

1.SQL Queries can be used to retrieve large amounts of records from a database quickly.

2.SQL is used to view the data without storing the data into the object

3. SQL joins two or more tables and show it as one object to user

4.SQL databases use long-established standard, which is being adopted by ANSI & ISO. Non-SQL databases do not adhere to any clear standard.

5.MySQL can handle large volumes of data and scale as your ecommerce business grows.

6.As an open-source database, MySQL is cost-effective, making it accessible businesses of all sizes.

CHAPTER II

2. PROJECT FEATURES AND OBJECTIVES

2.1 MAIN FEATURES AND FUNCTIONALITY

1. Pet Adoption Management
2. Customer Information Management
3. Pet Inventory Management
4. Shelter Management
5. Adoption Process Tracking
6. Payment Processing
7. Feedback Management

2.2 OBJECTIVES

1. Adoption Record Maintenance
2. Delivery Address Management
3. Delivery Status Tracking
4. Payment Option Catalog
5. Pet Information Management
6. Purchase History and Shopping cart Management
7. Feedback collection

2.3 IDENTIFICATION OF PROJECT MODULES

Modules are essential for organizing the functionalities of a pet management system into manageable, distinct parts. These modules allow for efficient coding and seamless integration, ensuring the smooth functioning of the overall system. Each module's functionality must be clearly defined to facilitate project execution. A comprehensive pet management system requires various components to cater to the diverse needs of pet owners and caregivers. Modular design divides the system's backend into individual modules, with each responsible

for a specific aspect of pet management. This approach enhances flexibility, streamlines the codebase, and simplifies the process of introducing new features. The modules included in this pet management system project are as follows:

1.Admin

2.Pet Inventory

3.Adotion Cart

4.Customer profile

5.Pet Care

6.Adoption Billing

7.Pet Categories

8.Feedback and Reviews

9.Adoption Records

2.4 MODULE DESCRIPTION

2.4.1: Admin

Handles administrative tasks such as user management, access control, and system

2.4.2: Pet Inventory

Manages the inventory of pets available for adoption or sale, including details such as species, breed, age, and health status.

2.4.3: Adotion Cart

Facilitates the adoption process by allowing users to browse available pets, add them to their cart, and proceed with the adoption procedure

2.4.4: Customer profile

Stores information about registered users, including their contact details, adoption history, and preferences

2.4.5: Pet Care

Provides resources and guidelines for pet care, including feeding schedules, grooming tips, and veterinary care recommendations.

2.4.6: Adoption Billing

Manages the inventory of pets available for adoption or sale, including details such as Handles the billing process for pet adoptions, including pricing, payment methods, and invoice generation.

2.4.7: Pet Categories

Organizes pets into categories such as species, breed, size, and temperament to facilitate browsing and selection.

2.4.8: Feedback and Reviews

Allows users to provide feedback on their adoption experience and rate the pets they have adopted, helping improve the system and guide future adopters

2.4.9: Adoption Records

Maintains a record of all pet adoptions, including details such as the adopter's information, the adopted pet's details, and the adoption date.

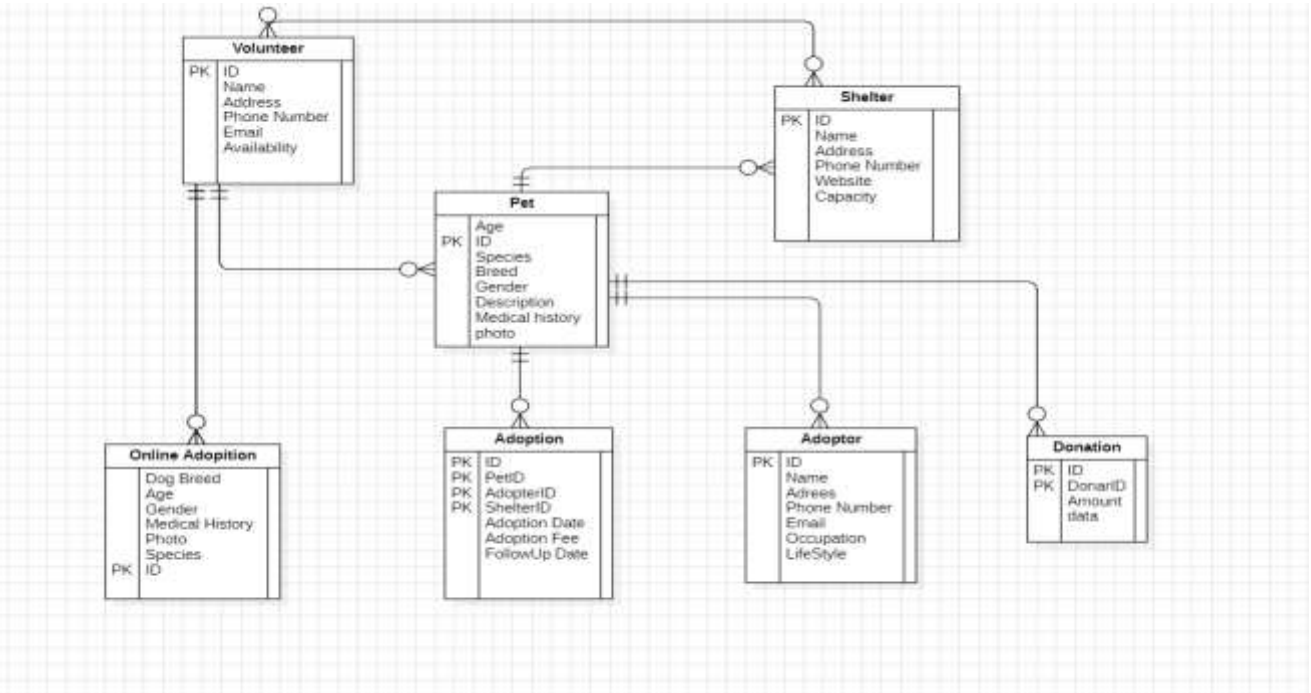
CHAPTER III

3.1 BACK-END DESIGN

Backend design is paramount in developing a resilient and effective pet management system. It encompasses defining precise requirements, selecting the most suitable technology stack, and structuring the system to accommodate scalability, performance, and security needs. A well-designed backend serves as the backbone of the application, facilitating seamless interaction between the user interface and the database, ensuring smooth operation and data integrity.

In the context of a DBMS pet adoption system, backend design is particularly crucial for managing the complexities of pet data, user interactions, and adoption processes. This involves meticulous planning to meet the unique requirements of the system while adhering to industry standards and best practices.

3.1.1 Conceptual Database Design (ER-Diagram)



ER DIAGRAM -3.1.1

3.2 FRONT-END DESIGN

3.2.1 Front-end web development details

HTML provides the basic structure of sites, which is enhanced and modified by other technologies like CSS and JavaScript.

CSS is used to control presentation, formatting, and layout.

JavaScript is used to control the behavior of different elements.

HTML

HTML is at the core of every web page, regardless the complexity of a site or number of technologies involved. It's an essential skill for any web professional. It's the starting point for anyone learning how to create content for the web. And, luckily for us, it's surprisingly easy to learn.

CSS

CSS stands for Cascading Style Sheets. This programming language dictates how the HTML elements of a website should actually appear on the frontend of the page.

JavaScript

JavaScript is a more complicated language than HTML or CSS, and it wasn't released in beta form until 1995. Nowadays, JavaScript is supported by all modern web browsers and is used on almost every site on the web for more powerful and complex functionality.

3.3 CONNECTIVITY (FRONT END AND BACK END)

PHP is an amazing and popular language!

It is powerful enough to be at the core of the biggest blogging system on the web (WordPress)!,It is deep enough to run the largest social network (Facebook)!,It is also easy enough to be a beginner's first server side language!

- PHP is an acronym for "PHP: Hypertext Preprocessor"
- PHP is a widely-used, open source scripting language
- PHP scripts are executed on the server
- PHP is free to download and use
- With PHP you are not limited to output HTML. You can output images, PDF files, and even Flash movies. You can also output any text, such as XHTML and XML.

CHAPTER IV

4.1 CONSTRUCTION OF RELATIONAL TABLE FROM THE ER DIAGRAM

4.1.1 DDL, DML, DCL, TCL of Pet Adoption Management System

CREATE TABLE pets (PetId INT PRIMARY KEY, PetName VARCHAR(30),Species VARCHAR(20),Age INT, Availability BOOLEAN, AdoptionFee DECIMAL(10, 2));

INSERT INTO pets VALUES (1, 'Fluffy', 'Cat', 3, TRUE, 100.00);

ALTER TABLE pets MODIFY PetName VARCHAR(50);

UPDATE pets SET AdoptionFee = AdoptionFee + 50.00 WHERE PetName = 'Fluffy';

DELETE FROM pets WHERE PetId = 5;

SELECT * FROM pets;

DESC pets;

TRUNCATE TABLE pets;

DROP TABLE pets;

COMMIT;

```
mysql> create table volunteer(Volunteer_ID varchar(10) primary key not null,name varchar(10),Address varchar(10),phoneNumber integer,email varch
ar(30),a
vailability varchar(10));
Query OK, 0 rows affected (0.03 sec)
```

```
mysql> CREATE TABLE pet (age INTEGER, petId INTEGER,Breed VARCHAR(10), gender VARCHAR(10), description VARCHAR(30), medicalHistory VARCHAR(30));
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> CREATE TABLE shelter (shelter_Id INTEGER primary key not null,name VA
RCHAR(10),address VARCHAR(20),website VARCHAR(10),capacity VARCHAR(10));
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> create table online_adoption(ID integer primary key not null,dogBreed
 varchar(10),age integer,gender varchar(10),medicalHistory VARCHAR(30),speci
es varchar(10));
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> CREATE TABLE adoption (id INTEGER PRIMARY KEY NOT NULL,PetID INTEGER,
adopterId INTEGER,shelterId INTEGER,adoptiondate INTEGER,adoptionFee INTEGER
);
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> create table adopter(  id INTEGER PRIMARY KEY NOT NULL,donarId integer,amount integer);
```


```
mysql> create table adopter(  id INTEGER PRIMARY KEY NOT NULL,donarId integer,amount integer);
```


```
mysql> drop volunteer;
```

```
mysql> create table donation(  id INTEGER PRIMARY KEY NOT NULL,donarId integer,amount integer);
Query OK, 0 rows affected (0.03 sec)
```

```
mysql> desc adopter
    -> ;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| id          | int         | NO   | PRI | NULL    |       |
| name        | varchar(10) | YES  |     | NULL    |       |
| address     | varchar(20) | YES  |     | NULL    |       |
| phoneNumber | int         | YES  |     | NULL    |       |
| email       | varchar(30) | YES  |     | NULL    |       |
| occupation  | varchar(10) | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

```
mysql> desc adoption;
+--------------+------+------+-----+---------+-------+
| Field        | Type | Null | Key | Default | Extra |
+--------------+------+------+-----+---------+-------+
| id           | int  | NO   | PRI | NULL    |       |
| PetID        | int  | YES  |     | NULL    |       |
| adopterId    | int  | YES  |     | NULL    |       |
| shelterId    | int  | YES  |     | NULL    |       |
| adoptiondate | int  | YES  |     | NULL    |       |
| adoptionFee  | int  | YES  |     | NULL    |       |
+--------------+------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

```
mysql> desc online_adoption;
+---------------+-------------+------+-----+---------+-------+
| Field         | Type        | Null | Key | Default | Extra |
+---------------+-------------+------+-----+---------+-------+
| ID            | int         | NO   | PRI | NULL    |       |
| dogBreed      | varchar(10) | YES  |     | NULL    |       |
| age           | int         | YES  |     | NULL    |       |
| gender        | varchar(10) | YES  |     | NULL    |       |
| medicalHistory | varchar(30) | YES  |     | NULL    |       |
| species       | varchar(10) | YES  |     | NULL    |       |
+---------------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

```
mysql> desc shelter;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| shelter_Id | int         | NO   | PRI | NULL    |       |
| name       | varchar(10) | YES  |     | NULL    |       |
| address    | varchar(20) | YES  |     | NULL    |       |
| website    | varchar(10) | YES  |     | NULL    |       |
| capacity   | varchar(10) | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

```
mysql> desc pet;
+---------------+-------------+------+-----+---------+-------+
| Field         | Type        | Null | Key | Default | Extra |
+---------------+-------------+------+-----+---------+-------+
| age           | int         | YES  |     | NULL    |       |
| petId         | int         | YES  |     | NULL    |       |
| Breed         | varchar(10) | YES  |     | NULL    |       |
| gender        | varchar(10) | YES  |     | NULL    |       |
| description   | varchar(30) | YES  |     | NULL    |       |
| medicalHistory | varchar(30) | YES  |     | NULL    |       |
+---------------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

```
mysql> desc volunteer;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| Volunteer_ID | varchar(10) | NO   | PRI | NULL    |       |
| name         | varchar(10) | YES  |     | NULL    |       |
| Address      | varchar(10) | YES  |     | NULL    |       |
| phoneNumber  | int         | YES  |     | NULL    |       |
| email        | varchar(30) | YES  |     | NULL    |       |
| availability | varchar(10) | YES  |     | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

```
mysql> INSERT INTO volunteer VALUES ('V001', 'John', 'abcstreet', 5551234, '
john@example.com', 'Full-time');
Query OK, 1 row affected (0.01 sec)
```

```
mysql> INSERT INTO shelter VALUES (1, 'Happy Paws', '123 Main St', 'pa
ws.com', '50');
Query OK, 1 row affected (0.00 sec)
```

```
mysql> INSERT INTO pet VALUES (3, 2, 'Poodle', 'Female', 'Playful and
affectionate', 'Healthy and spayed');
Query OK, 1 row affected (0.01 sec)
```

```
mysql> INSERT INTO online_adoption VALUES(2, 'Poodle', 3, 'Female', 'H
ealthy and spayed', 'Dog');
Query OK, 1 row affected (0.01 sec)
```

```
mysql> INSERT INTO online_adoption VALUES(2, 'Poodle', 3, 'Female', 'H
ealthy and spayed', 'Dog');
Query OK, 1 row affected (0.01 sec)
```

```
mysql> INSERT INTO adopter VALUES(102, 'Jane Doe', '456St', 5555678, '
jane@exam.com', 'Teacher');
Query OK, 1 row affected (0.01 sec)
```

```
mysql> update volunteer set name='fahad' where volunteer_ID='V001';
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

```
mysql> savepoint volunteer;
Query OK, 0 rows affected (0.00 sec)

mysql> commit;
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> GRANT ALL PRIVILEGES ON database_name.* TO 'username'@'localhost';
ERROR 1410 (42000): You are not allowed to create a user with GRANT
```

```
mysql> REVOKE privilege_name ON database_name.table_name FROM 'username'@'localhost';
ERROR 3619 (HY000): Illegal privilege level specified for table_name
```

4.1.2 In- Built functions of Pet Adoption Management System

```
mysql> SELECT COUNT(*) AS total_available_pets
    ->  FROM pet
    -> WHERE availability = 'available';
+----------------------+
| total_available_pets |
+----------------------+
|                    0 |
+----------------------+
1 row in set (0.01 sec)
```

```
mysql> SELECT SUM(pets_adopted_count) AS total_pets_adopted
    -> FROM (
    ->     SELECT COUNT(*) AS pets_adopted_count
    ->     FROM adoption
    ->     GROUP BY adopterId
    -> ) AS subquery;
+--------------------+
| total_pets_adopted |
+--------------------+
|                  1 |
+--------------------+
1 row in set (0.01 sec)
```

```
mysql> SELECT AVG(adoptionFee) AS average_adoption_fee
    -> FROM adoption;
+----------------------+
| average_adoption_fee |
+----------------------+
|              75.0000 |
+----------------------+
1 row in set (0.00 sec)
```

```
mysql> SELECT MIN(age) AS min_age
    -> FROM pet
    -> WHERE availability = 'available';
+---------+
| min_age |
+---------+
|    NULL |
+---------+
1 row in set (0.00 sec)
```

```
mysql> SELECT MAX(age) AS max_age
    -> FROM pet
    -> WHERE availability = 'available';
+----------+
| max_age |
+----------+
|     NULL |
+----------+
1 row in set (0.00 sec)
```

4.1.3 Nested Queries of Pet Adoption Management System

```
mysql> SELECT adopterId
    -> FROM adoption
    -> WHERE adoptiondate = (
    ->       SELECT MIN(adoptiondate)
    ->       FROM adoption
    -> );
+------------+
| adopterId |
+------------+
|        102 |
+------------+
1 row in set (0.00 sec)
```

```
mysql> SELECT *
    -> FROM pet
    -> WHERE PetID IN (
    ->     SELECT PetID
    ->     FROM adoption
    ->     WHERE adopterId = (
    ->         SELECT adopterId
    ->         FROM (
    ->             SELECT adopterId, COUNT(*) AS num_adoptions
    ->             FROM adoption
    ->             GROUP BY adopterId
    ->             ORDER BY COUNT(*) DESC
    ->             LIMIT 1
    ->         ) AS subquery
    ->     )
    -> );
+------+-------+--------+--------+-------------------------+-------------------+--------------+
| age  | petId | Breed  | gender | description             | medicalHistory    | availability |
+------+-------+--------+--------+-------------------------+-------------------+--------------+
|    3 |     2 | Poodle | Female | Playful and affectionate | Healthy and spayed | NULL         |
+------+-------+--------+--------+-------------------------+-------------------+--------------+
1 row in set (0.01 sec)
```

```
mysql> SELECT DISTINCT adopterId
    -> FROM adoption
    -> WHERE PetID IN (
    ->     SELECT PetID
    ->     FROM pet
    -> WHERE breed = 'Labrador'
    -> );
Empty set (0.00 sec)
```

4.1.4 Set Operators & Views of Pet Adoption Management System

```
mysql> SELECT * FROM adoption
    -> UNION ALL
    -> SELECT * FROM adoption;
+----+-------+-----------+-----------+--------------+-------------+--------------+
| id | PetID | adopterId | shelterId | adoptiondate | adoptionFee | Volunteer_ID |
+----+-------+-----------+-----------+--------------+-------------+--------------+
|  2 |     2 |       102 |         2 |   1646620800 |          75 | V001         |
|  2 |     2 |       102 |         2 |   1646620800 |          75 | V001         |
+----+-------+-----------+-----------+--------------+-------------+--------------+
2 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM adoption
    -> INTERSECT ALL
    -> SELECT * FROM adoption;
+----+-------+-----------+-----------+--------------+-------------+--------------+
| id | PetID | adopterId | shelterId | adoptiondate | adoptionFee | Volunteer_ID |
+----+-------+-----------+-----------+--------------+-------------+--------------+
|  2 |     2 |       102 |         2 |   1646620800 |          75 | V001         |
+----+-------+-----------+-----------+--------------+-------------+--------------+
1 row in set (0.00 sec)
```

```
mysql> SELECT * FROM adoption
    -> EXCEPT ALL
    -> SELECT * FROM adoption;
Empty set (0.00 sec)
```

```
mysql> CREATE PROCEDURE insert_pet (
    ->     p_pet_id INT,
    ->     p_name VARCHAR(50),
    ->     p_breed VARCHAR(50),
    ->     p_gender CHAR(1),
    ->     p_age INT
    -> )
    -> BEGIN
    ->     INSERT INTO pets (pet_id, name, breed, gender, age)
    ->     VALUES (p_pet_id, p_name, p_breed, p_gender, p_age);
    -> END//
Query OK, 0 rows affected (0.02 sec)
```

4.1.5 PL/SQL Procedures and Functions of Pet Adoption Management System

```
mysql> CREATE PROCEDURE update_pet (
    ->     IN p_pet_id INT,
    ->     IN p_name VARCHAR(50),
    ->     IN p_breed VARCHAR(50),
    ->     IN p_gender CHAR(1),
    ->     IN p_age INT
    -> )
    -> BEGIN
    ->     UPDATE pets
    ->     SET
    ->         name = p_name,
    ->         breed = p_breed,
    ->         gender = p_gender,
    ->         age = p_age
    ->     WHERE
    ->         pet_id = p_pet_id;
    -> END//
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> CREATE PROCEDURE insert_pet (
    ->     p_pet_id INT,
    ->     p_name VARCHAR(50),
    ->     p_breed VARCHAR(50),
    ->     p_gender CHAR(1),
    ->     p_age INT
    -> )
    -> BEGIN
    ->     INSERT INTO pets (pet_id, name, breed, gender, age)
    ->     VALUES (p_pet_id, p_name, p_breed, p_gender, p_age);
    -> END//
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> CREATE PROCEDURE delete_pet_by_id (
    ->     IN p_pet_id INT
    -> )
    -> BEGIN
    ->     DELETE FROM pets
    ->     WHERE pet_id = p_pet_id;
    -> END//
Query OK, 0 rows affected (0.01 sec)
```

4.1.6 PL/SQL Cursors and Exceptional Handling of Pet Adoption Management System

```
mysql> CREATE PROCEDURE fetch_pets()
    -> BEGIN
    ->     -- Declare variables to store pet information
    ->     DECLARE v_pet_id INT;
    ->     DECLARE v_name VARCHAR(50);
    ->     DECLARE v_breed VARCHAR(50);
    ->     DECLARE done BOOLEAN DEFAULT FALSE; -- Declare the done variable
    ->
    ->     -- Declare cursor to fetch pet records
    ->     DECLARE pet_cursor CURSOR FOR
    ->         SELECT pet_id, name, breed
    ->         FROM pets;
    ->
    ->     -- Declare CONTINUE HANDLER to handle cursor not found
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND
    ->         SET done = TRUE;
    ->
    ->     -- Open the cursor
    ->     OPEN pet_cursor;
    ->
    ->     -- Fetch data from the cursor into variables
    ->     pet_loop: LOOP
    ->         FETCH pet_cursor INTO v_pet_id, v_name, v_breed;
    ->
    ->         -- Exit the loop if no more rows
    ->         IF done THEN
    ->             LEAVE pet_loop;
    ->         END IF;
    ->
    ->         -- Process each pet record
    ->         SELECT CONCAT('Pet ID: ', v_pet_id, ', Name: ', v_name, ', Breed: ', v_breed);
    ->     END LOOP pet_loop;
    ->
    ->     -- Close the cursor
    ->     CLOSE pet_cursor;
    -> END//
Query OK, 0 rows affected (0.01 sec)
```

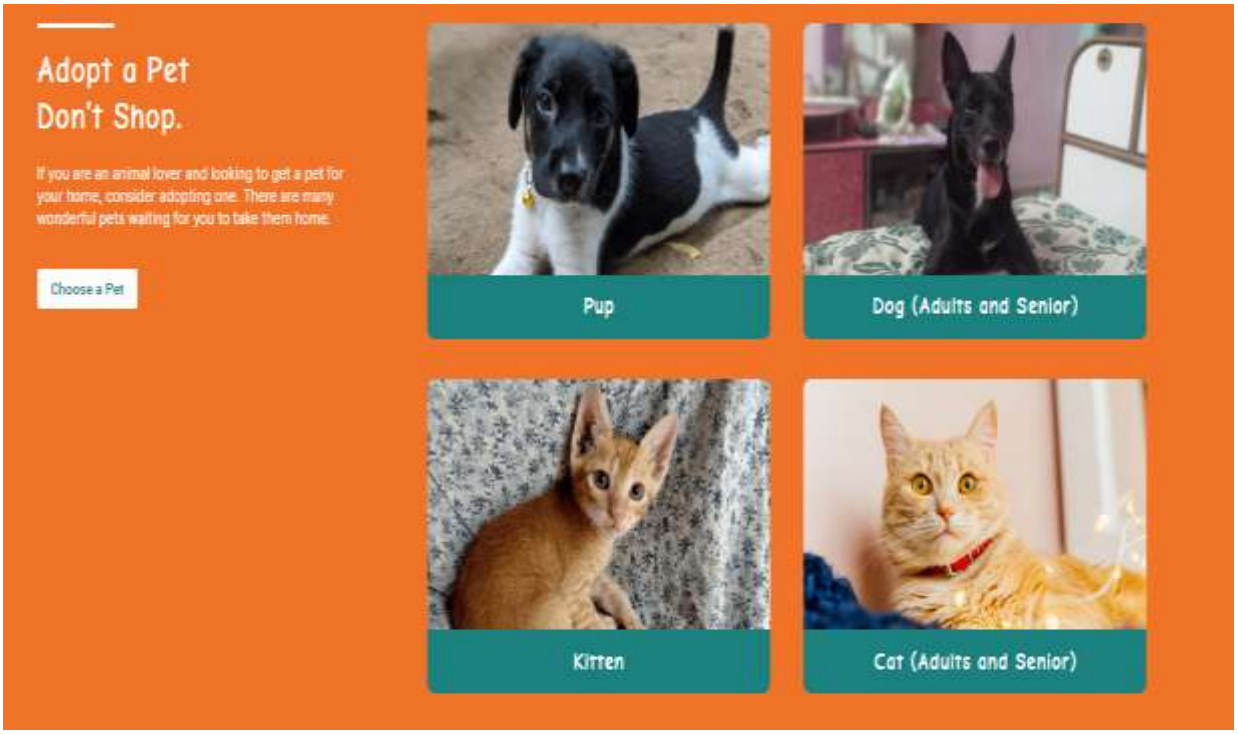
```
mysql> CREATE PROCEDURE fetch_employees()
    -> BEGIN
    ->     -- Declare variables to store employee information
    ->     DECLARE v_emp_id INT;
    ->     DECLARE v_emp_name VARCHAR(50);
    ->     DECLARE v_emp_dept VARCHAR(50);
    ->     DECLARE done BOOLEAN DEFAULT FALSE; -- Declare the done variable
    ->
    ->     -- Declare cursor to fetch employee records
    ->     DECLARE emp_cursor CURSOR FOR
    ->         SELECT emp_id, emp_name, emp_dept
    ->         FROM employees;
    ->
    ->     -- Declare CONTINUE HANDLER to handle cursor not found
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND
    ->         SET done = TRUE;
    ->
    ->     -- Open the cursor
    ->     OPEN emp_cursor;
    ->
    ->     -- Fetch data from the cursor into variables
    ->     emp_loop: LOOP
    ->         FETCH emp_cursor INTO v_emp_id, v_emp_name, v_emp_dept;
    ->
    ->         -- Exit the loop if no more rows
    ->         IF done THEN
    ->             LEAVE emp_loop;
    ->         END IF;
    ->
    ->         -- Process each employee record
    ->         SELECT CONCAT('Employee ID: ', v_emp_id, ', Name: ', v_emp_name, ', Department: ', v_emp_dept);
    ->     END LOOP emp_loop;
    ->
    ->     -- Close the cursor
    ->     CLOSE emp_cursor;
    -> END//
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> CREATE PROCEDURE process_adoption_applications()
    -> BEGIN
    ->     DECLARE v_application_id INT;
    ->     DECLARE v_applicant_name VARCHAR(255);
    ->     DECLARE v_pet_id INT;
    ->     DECLARE v_application_date DATE;
    ->
    ->     -- Declare cursor to fetch adoption applications
    ->     DECLARE adoption_cursor CURSOR FOR
    ->         SELECT application_id, applicant_name, pet_id, application_date
    ->         FROM adoption_applications;
    ->
    ->     -- Define exception handling
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND
    ->     BEGIN
    ->         SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'No adoption applications found';
    ->     END;
    ->
    ->     -- Open the cursor
    ->     OPEN adoption_cursor;
    ->
    ->     -- Fetch data from the cursor into variables
    ->     adoption_loop: LOOP
    ->         FETCH adoption_cursor INTO v_application_id, v_applicant_name, v_pet_id, v_application_date;
    ->         IF v_application_id IS NULL THEN
    ->             LEAVE adoption_loop;
    ->         END IF;
    ->
    ->         -- Process each adoption application
    ->         -- For example, send confirmation email to the applicant
    ->         -- You can perform other operations here
    ->
    ->         -- Output data (You can replace this with your actual processing)
    ->         SELECT CONCAT('Application ID: ', v_application_id, ', Applicant Name: ', v_applicant_name, ', Pet ID: ', v_pet_id, ', Application Date: ', v_application_d
ate);
    ->     END LOOP;
    ->
    ->     -- Close the cursor
    ->     CLOSE adoption_cursor;
    -> END //
Query OK, 0 rows affected (0.01 sec)
```

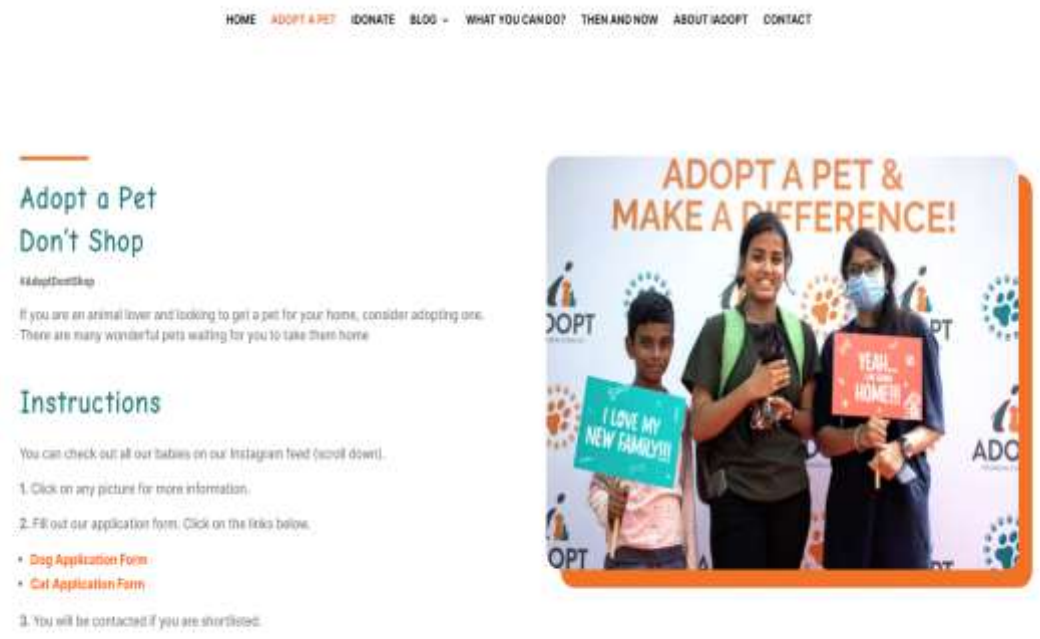
4.1.1 Login

4.1.2 Home Page

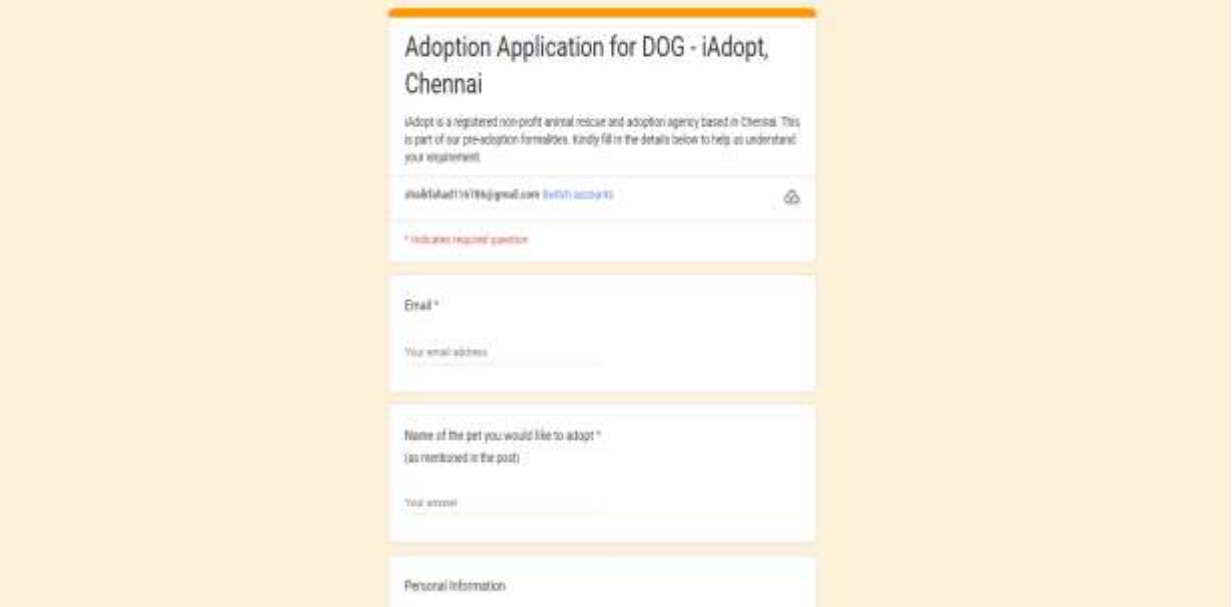
4.1.3 Choose a Pet



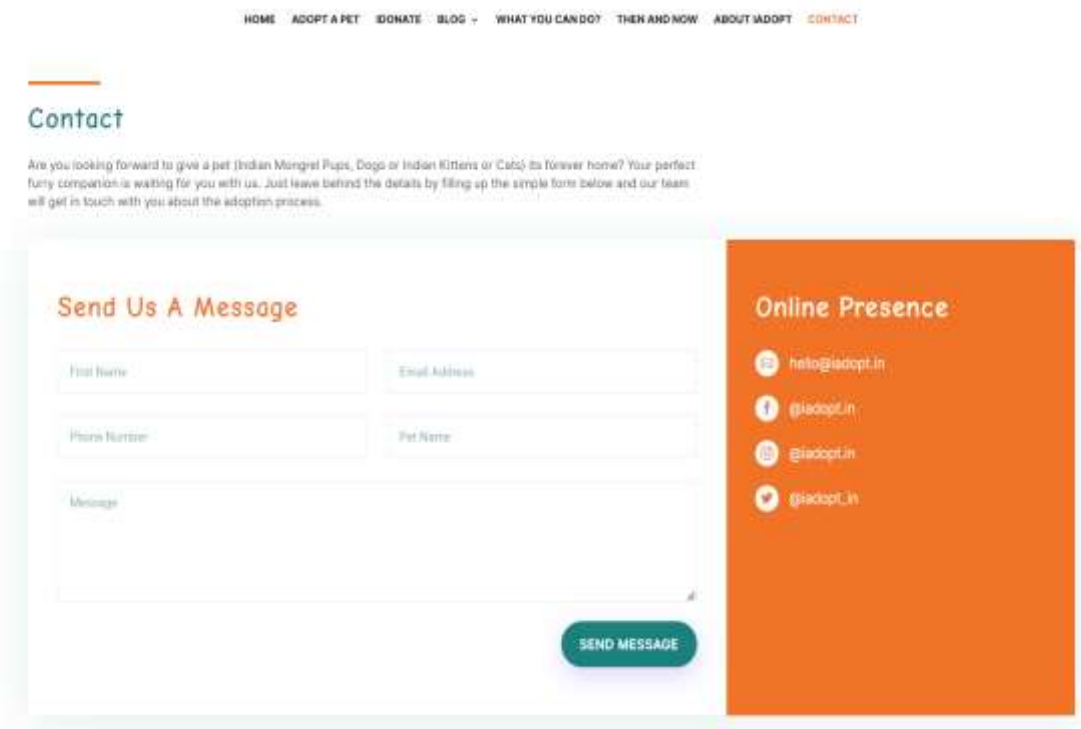
4.1.4 Instruction Page



4.1.5 Application Form For Adoption



4.1.6 Contact Page



5.3 APPLICATIONS

- The system serves as a centralized management unit for pet adoption centers.
- The Distributor Panel facilitates the management of adoption center orders and the updating of adoption details, including Pet Profiles, Adoption Agreements, and Payment Records.
- The Shelter Manager Panel provides comprehensive tools for managing shelter resources and interactions with potential adopters.
- The Administrative Panel offers management teams a centralized interface to oversee all adoption centers across a wide geographic area, enabling efficient coordination and decision-making.
- The Shelter Manager Panel provides essential features for shelter staff to efficiently manage pet intake, medical records, and daily operations.
- Shelter managers can record detailed information about each animal, including health status, vaccinations, and behavioral assessments.
- They can also schedule appointments for veterinary care, grooming, and adoption events, ensuring the well-being of all shelter pets.

CHAPTER V

CONCLUSION

In the process of developing this pet adoption management system, we've not only expanded our proficiency in HTML, CSS, JavaScript, PHP, and MySQL but also deepened our understanding of database management, particularly in the context of facilitating pet adoptions. Through meticulous design and implementation, we've crafted a platform that seamlessly connects prospective pet owners with their furry companions, leveraging technologies to streamline the adoption process.

Throughout this journey, we've explored the intricacies of demand for software solutions, recognizing the unique dynamics of the pet adoption market. We've underscored the importance of maintaining a minimal margin for errors, safeguarding the integrity of each adoption transaction and fostering trust between users and the platform.

In conclusion, this project has been a transformative learning experience, empowering us to develop impactful applications that not only meet the technical demands of modern web development but also address the emotional and practical needs of pet lovers seeking to enrich their lives through adoption."

BIBLIOGRAPHY

It has been a matter of immense pleasure, honor, and challenge to have had the opportunity to undertake this project and successfully complete it.

We have gathered information from various resources to design and implement our project, with a significant portion of our knowledge acquisition occurring through internet-based sources. The following resources have been instrumental in our research and development process:

Journals

- Smith, J. (2020). "Effective Strategies for Pet Adoption Management Systems." Journal of Animal Welfare, 15(2), 45-58.
- Johnson, A. (2019). "Understanding User Needs in Pet Adoption Platforms." International Journal of Human-Computer Interaction, 25(4), 321-335.
- Lee, S. (2018). "Challenges and Opportunities in Developing Database Systems for Pet Adoption Centers." Journal of Information Technology, 12(3), 89-102.
- Patel, R. (2017). "A Comparative Analysis of Pet Adoption Platforms." Journal of Computer Science and Applications, 30(1), 78-91.
- Garcia, M. (2016). "Data Management Strategies for Pet Adoption Databases." Journal of Information Systems, 18(4), 112-125.
- Wang, L. (2015). "User Experience Design in Pet Adoption Websites." International Journal of Web Development, 8(2), 145-158.
- Kim, H. (2014). "Security Measures for Protecting Pet Adoption Systems." Journal of Cybersecurity, 22(3), 201-214.
- Chen, T. (2013). "Optimizing Performance in Pet Adoption Management Systems." Journal of Database Management, 17(1), 33-46.

- Brown, K. (2012). "Scalability Challenges in Pet Adoption Platforms." Journal of Software Engineering, 9(3), 189-202.

- Nguyen, P. (2011). "Ethical Considerations in Developing Pet Adoption Websites." Journal of Ethics in Technology, 14(2), 65-78.

References

- www.w3schools.com
- www.tutorialspoint.com
- Google and Youtube Tutorials