Pet Adoption App: PetSearch

Final Project

Course: Database Technology
(COMP6799001)

Lecturer:
Dr. Raymond Bahana, ST., M.Sc

Kenneth Lay - 2602119272
Bryan Sujoso - 2602179151
Mohammad Sulthan Azka - 2602209601
Class: L3AC,L3CC

Computer Science Major
Binus International University
2023

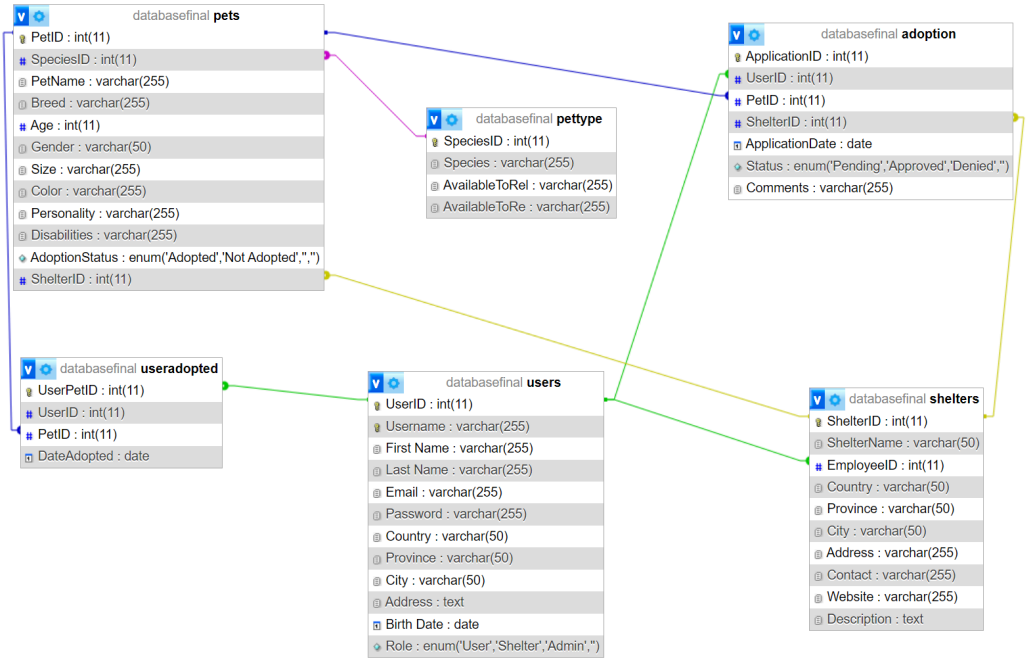
I. Problem Description

Many people who are looking to adopt a pet face challenges in finding the perfect companion and where to get them. Additionally, shelters and pet rescue organizations struggle to connect available pets with potential adopters efficiently. There is a need for a comprehensive and user-friendly platform that facilitates the process of pet adoption while providing a centralized solution for shelters to manage and promote their adoptable pets.

The Pet Search is a comprehensive database-driven platform designed to facilitate the pet adoption process, enhance user engagement, and support the operations of exotic animal stores, shelters and rescue organizations. This innovative platform aims to bridge the gap between prospective pet owners and pets in need of loving homes. It provides an intuitive and user-friendly interface for users to discover, connect with, and potentially adopt pets while offering shelters the tools to efficiently manage their operations.

II. Database Design

a. Entity Relationship Diagram



- pets (Strong): This entity contains information about the pets available for adoption. Attributes include:
  PetID (Primary Key) - int(11)
  SpeciesID (Foreign Key) - int(11)
  PetName - varchar(255)
  Breed - varchar(255)
  Age - int(11)
  Gender - varchar(50)
  Size - varchar(255)
  Color - varchar(255)
  Personality - varchar(255)
  Disabilities - varchar(255)
  Adoption Status - enum('Adopted','Not Adopted')
  ShelterID (Foreign Key) - int(11)

- pettype (Strong): This entity stores the types of pets available. Attributes include:
  SpeciesID (Primary Key) - int(11)
  Species - varchar(255)
  AvailableToRel - varchar(255)
  AvailableToRe - varchar(255)

- adoption (Weak): This entity manages the adoption applications. Attributes include:
  ApplicationID (Primary Key) - int(11)
  UserID (Foreign Key) - int(11)
  PetID (Foreign Key) - int(11)
  ShelterID (Foreign Key) - int(11)
  ApplicationDate - date
  Status - enum('Pending','Approved','Denied')
  Comments - varchar(255)

- useradopted (Weak): This is a junction table that links users with pets they have adopted. Attributes include:
  UserPetID (Primary Key) - int(11)
  UserID (Foreign Key) - int(11)
  PetID (Foreign Key) - int(11)
  DateAdopted - date

- users (Strong): This entity contains details about the users of the system, which could be adopters or employees. Attributes include:
  UserID (Primary Key) - int(11)
  Username (Unique Key) - varchar(255)
  FirstName - varchar(255)

LastName - varchar(255)
Email (Unique Key)- varchar(255)
Password - varchar(255)
Country - varchar(50)
Province - varchar(50)
City - varchar(50)
Address - text
BirthDate - date
Role - enum('User','Shelter','Admin')

- shelters (Strong): This entity holds information about the shelters that have pets. Attributes include:
  ShelterID (Primary Key) - int(11)
  ShelterName - varchar(50)
  EmployeeID (Foreign Key) - int(11)
  Country - varchar(50)
  Province - varchar(50)
  City - varchar(50)
  Address - varchar(255)
  Contact - varchar(255)
  Website - varchar(255)
  Description - text

b. Relations

1. Entity Relationship

pets to pettype: Each pet has a Species ID that relates to the pettype entity. This is a many-to-one relationship where many pets can be of one species type.

pets to adoption: A one-to-one relationship between pets and adoption indicates that each pet has a single adoption record.

pets to useradopted: This represents the pets that users have adopted. The one-to-one relationship means that once a pet is adopted, it is associated with a single adoption record.

pets to shelters: Each pet has a Shelter ID which indicates what shelter the pet is currently residing at. This is a many-to-one relationship where many pets can reside in one shelter.

users to useradopted: This indicates which users have adopted which pets. Since it's a one-to-many relationship, it means that users can adopt multiple pets, and different users can adopt pets at different times.

users to adoption: This one-to-many relationship shows that users can have many adoption applications, as a user can apply to adopt many pets, and each adoption record is linked to one user.

users to shelters: A shelter is capable of having one user managing the shelter account resulting in a one-to-one relationship.

shelters to adoption: Each adoption application is associated with a shelter, and a shelter can have many adoption applications. This is a one-to-many relationship.

The lines between the entities represent the relationships, with the "#" symbol indicating a foreign key, and the crow's foot notation at the end of the lines indicating "many" in the one-to-many relationships.

2. Keys Justification

- pets

PetID (Primary Key): we set PetID as primary key, because there is a need for a unique identifier for each pet, so we can manage the pets data easily. We are unable to use their age, gender, disabilities, names and others as primary keys, because other pets might be able to have the similar data.

SpeciesID (Foreign Key): The reason this is foreign key is because through observation we realized that if we repetitively put the type of pets with their availability toward region and religion it would make it look redundant, so we separate them into a separate species table and give the species table for each animal a SpeciesID and this SpeciesID is what connects that table the other table as foreign key.

ShelterID (Foreign Key): This has a similar reason to SpeciesID, but instead the shelter contained the Shelter name, location, and so on. We made it so there is no redundant content in the pets table and it acts as a connection to the shelters table.

- pettype

SpeciesID (Primary Key): At first we wanted to put multiple pettype but that would take a long time, so initially we put SpeciesID as a unique classifier for each type of species of different variety and breeds.

- adoption

ApplicationID (Primary Key): This is the unique identifier for each application and the one used to fetch the application, as other columns within this entity could have further duplicates of each other

UserID (Foreign Key): To connect the users table to the application or adoption table to check which user is the one adopting.

PetID (Foreign Key): To connect the pets table to the application or adoption table to check which pets are being adopted.

ShelterID (Foreign Key): To connect the shelters table to the application or adoption table to check which shelter the pet is being adopted from.

- useradopted

UserPetID (Primary Key): This primary key is just used as a connection between the users and pet for this table, to be honest it is actually unnecessary that's why this is counted as a weak table/entity.
UserID (Foreign Key): to connect to the users who own these pets.
PetID (Foreign Key): to connect to which pet is the one that the user adopted.

- users

UserID (Primary Key): This is a unique and primary identifier of each users, since we can't use their name, address, password and so on as the identifiers, because their could be duplicates, but we could use email and username but I think based of work ethics, it is not appropriate since it will be harder to find users by their usernames or email than by id.

Username (Unique Key): Each user should have a different Username, since there will be a problem with login if there are multiple users with the same username.

Email (Unique Key): Each user should have a different Email, since there will be a problem with login if there are multiple users with the same Email and plus to prevent duplicate accounts with the same email and so users also cannot make multiple accounts with the same email.

- shelters

ShelterID (Primary Key): To identify the shelters since some shelters could have different branches of them, or their might be some shelters with the same or similar name, so we need unique id identifier for it

EmployeeID (Foreign Key): EmployeedID actually is the connection at the users table, because the EmployeeID is technically the UserID of a user that falls under the role of “shelter”.

c. Normalization

Right now, our database is configured following the Third Normal Form (3NF) principles, which are a level of database normalization that guarantees data is arranged logically and effectively. All of the data in 3NF is saved so that no piece of information is duplicated, data dependencies are reasonable, and each item of information is only stored once. Anomalies during database operations, such as updates, deletions, and insertions, are successfully prevented by this form. Reaching 3NF is essential for preserving a database's effectiveness and integrity. Given that our database is already in 3NF, we do not require any further normalization processes at this stage.

III. Sample Queries

provide some sample queries (at least 5) to generate reports.

Checks if user exist or not

```
$select = "SELECT * FROM users WHERE Email = '$Email' AND Password = '$Password'";
```

```
$select = " SELECT * FROM users WHERE Email = '$Email' && password = '$Password' ";
```

Find the shelter from the selected pet

```
// Retrieve the ShelterID associated with the given PetID
$sql = "SELECT ShelterID FROM pets WHERE PetID = '$petID'";
```

Generate the shelter table based on the shelter id

```
"SELECT pets.PetID, pettype.Species, pets.PetName, pets.Breed, pets.Age,
 pets.Gender, pets.Size, pets.Color, pets.Personality, pets.Disabilities,
 pets.AdoptionStatus, shelters.ShelterName
FROM pets
LEFT JOIN pettype ON pets.SpeciesID = pettype.SpeciesID
LEFT JOIN shelters ON pets.ShelterID = shelters.ShelterID
WHERE pets.ShelterID = '$shelterID' $whereClause";
```

Check which shelter employee login and chec the shelter id

```
$shelterQuery = "SELECT ShelterID FROM shelters WHERE EmployeeID = '$employeeID'"
```

Finding out the distinct type of datas to put into the filter options

```php
$sqlSpecies = "SELECT DISTINCT SpeciesID FROM pets";
$resultSpecies = $connection->query($sqlSpecies);

// Fetch distinct breed values from the database
$sqlBreed = "SELECT DISTINCT Breed FROM pets";
$resultBreed = $connection->query($sqlBreed);

// Fetch distinct age values from the databasez
$sqlAge = "SELECT DISTINCT Age FROM pets";
$resultAge = $connection->query($sqlAge);

// Fetch distinct gender values from the database
$sqlGender = "SELECT DISTINCT Gender FROM pets";
$resultGender = $connection->query($sqlGender);

// Fetch distinct size values from the database
$sqlSize = "SELECT DISTINCT Size FROM pets";
$resultSize = $connection->query($sqlSize);

// Fetch distinct personality values from the database
$sqlPersonality = "SELECT DISTINCT Personality FROM pets";
$resultPersonality = $connection->query($sqlPersonality);

// Fetch distinct disabilities values from the database
$sqlDisabilities = "SELECT DISTINCT Disabilities FROM pets";
$resultDisabilities = $connection->query($sqlDisabilities);

// Fetch distinct adoption status values from the database
$sqlAdoptionStatus = "SELECT DISTINCT AdoptionStatus FROM pets";
```

Filtering queries based on selection

```php
if (isset($_GET['species']) && $_GET['species'] !== '') {
    $whereClause .= " AND pets.SpeciesID = '" . $connection->real_escape_string($_GET['species']) . "'";
}

if (isset($_GET['breed']) && $_GET['breed'] !== '') {
    $whereClause .= " AND pets.Breed = '" . $connection->real_escape_string($_GET['breed']) . "'";
}

if (isset($_GET['age']) && $_GET['age'] !== '') {
    $whereClause .= " AND pets.Age = '" . $connection->real_escape_string($_GET['age']) . "'";
}

if (isset($_GET['gender']) && $_GET['gender'] !== '') {
    $whereClause .= " AND pets.Gender = '" . $connection->real_escape_string($_GET['gender']) . "'";
}

if (isset($_GET['size']) && $_GET['size'] !== '') {
    $whereClause .= " AND pets.Size = '" . $connection->real_escape_string($_GET['size']) . "'";
}

if (isset($_GET['personality']) && $_GET['personality'] !== '') {
    $whereClause .= " AND pets.Personality = '" . $connection->real_escape_string($_GET['personality']) . "'";
}

if (isset($_GET['disabilities']) && $_GET['disabilities'] !== '') {
    $whereClause .= " AND pets.Disabilities = '" . $connection->real_escape_string($_GET['disabilities']) . "'";
}

if (isset($_GET['adoption_status']) && $_GET['adoption_status'] !== '') {
    $whereClause .= " AND pets.AdoptionStatus = '" . $connection->real_escape_string($_GET['adoption_status']) . "'";
}
```

Generating table with the specified filter in the Users table by what they filter it as, and also it changes the shelterid into shelternamе and speciesid into speciesname

```
// Fetch data from the database with filters
$sql = "SELECT pets.PetID, pettype.Species, pets.PetName, pets.Breed, pets.Age,
        pets.Gender, pets.Size, pets.Color, pets.Personality, pets.Disabilities,
        pets.AdoptionStatus, shelters.ShelterName
        FROM pets
        LEFT JOIN pettype ON pets.SpeciesID = pettype.SpeciesID
        LEFT JOIN shelters ON pets.ShelterID = shelters.ShelterID
        WHERE 1 $whereClause";
```

Adding pets

```
"INSERT INTO pets (SpeciesID, ShelterID, PetName, Breed, Age, Gender, Size, Color, Personality, Disabilities, AdoptionStatus, ShelterID)
 VALUES ('$speciesID', '$shelterID', '$petName', '$breed', '$age', '$gender', '$size', '$color', '$personality', '$disabilities', 'Not Adopted','$shelterQuery')";
```

Deleting using through admin

```
$delete = "DELETE FROM user WHERE id = $id";
```

For updating user information through admin

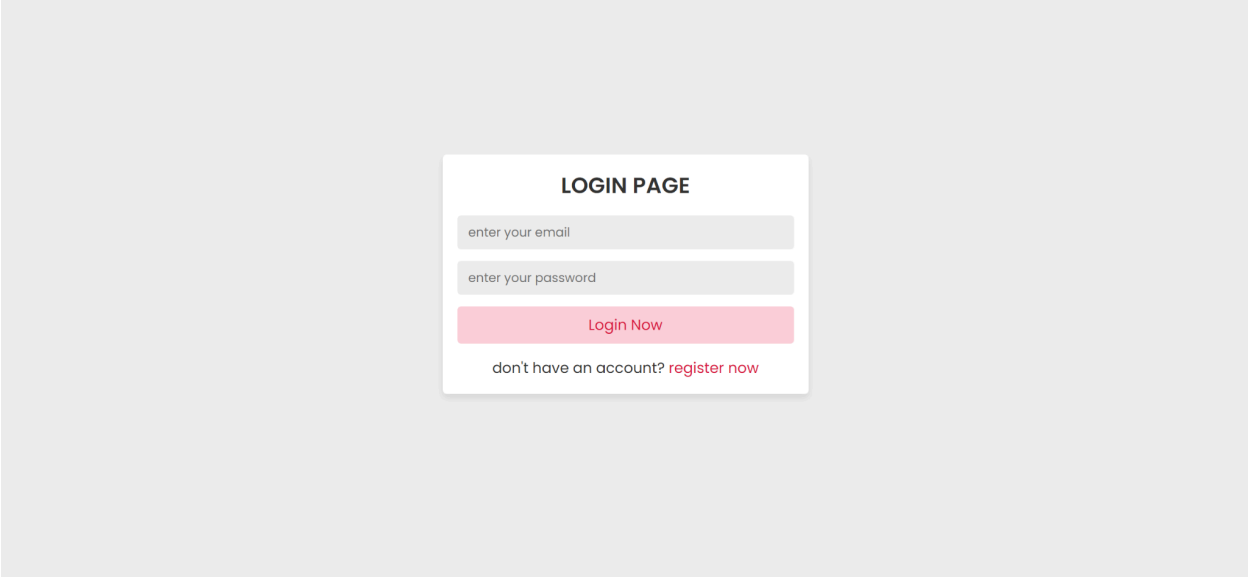
```
$insertquery =  "UPDATE `users` SET `Username`='$Name',`First Name`='$Fname',
                `Last Name`='$Lname',`Email`='$Email',`Password`='$Password',`Country`='$Country',
                `Province`='$Province',`City`='$City',`Address`='$Address',`Birth Date`='$Birth',`Role`='$Role' WHERE UserID = '$id'";
```

For inserting or adding new users through admin

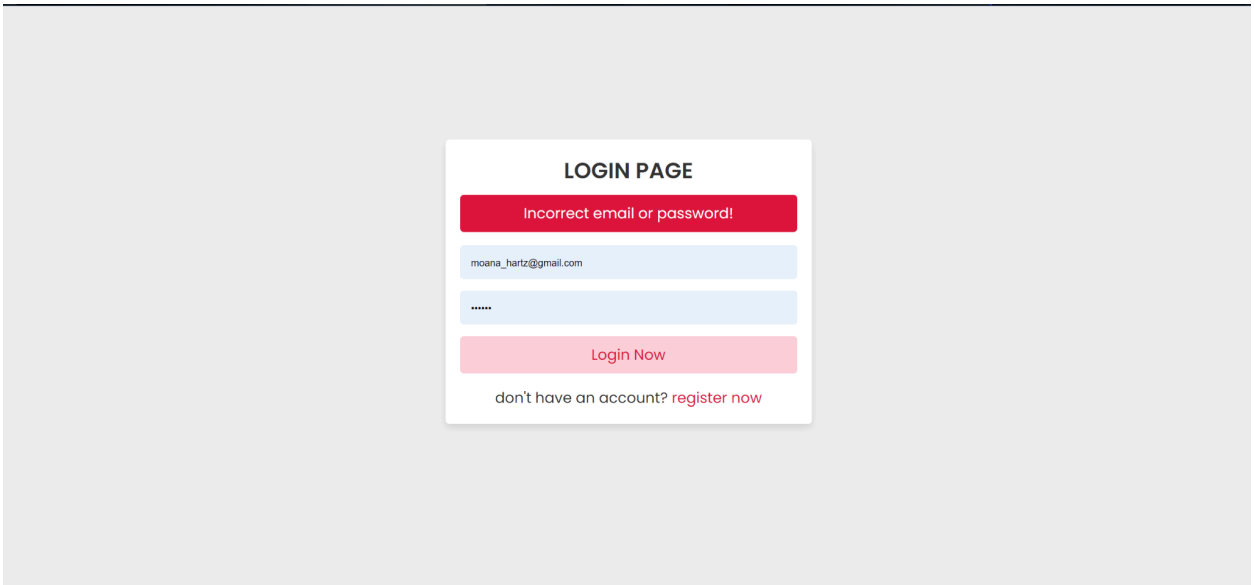
```
$insertquery =  "INSERT INTO `users` (`UserID`,`Username`, `First Name`, `Last Name`, `Email`, `Password`, `Country`, `Province`, `City`, `Address`, `Birth Date`, `Role`)
                VALUES (NULL,'$Name','$Fname','$Lname','$Email','$Password','$Country','$Province','$City','$Address','$Birth','$Role')";
```

IV. User Interface

Login Page

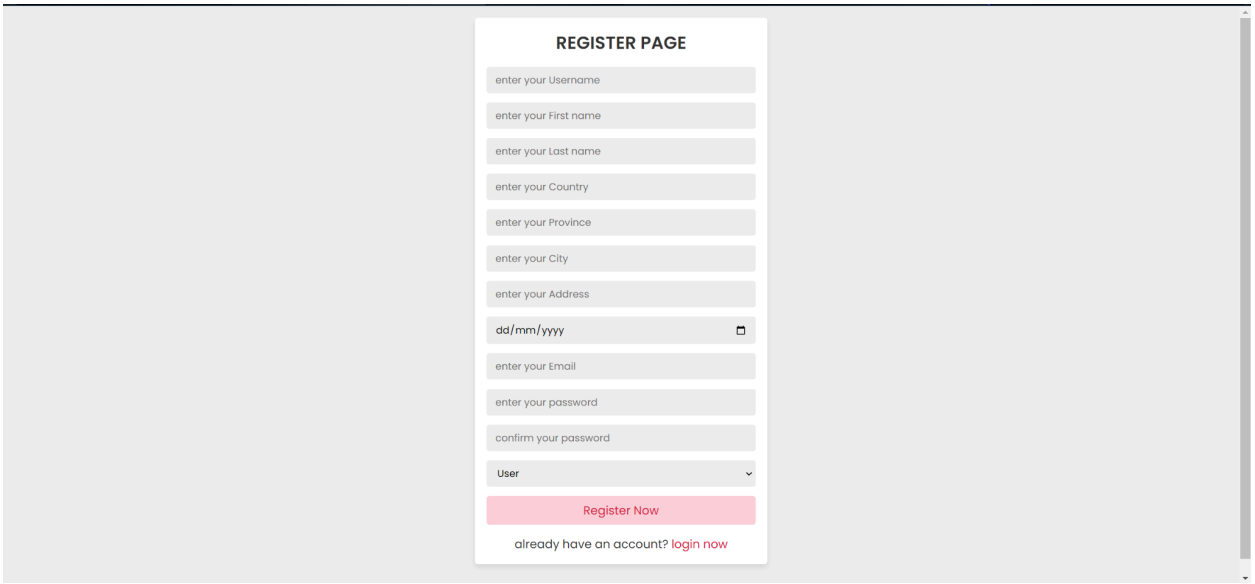


Login page is to allow login of the users. If the user doesn't have an account, they can also register for their account in this page and if the inputted data is wrong, it will show an error page like shown below.



Additionally, there are three types of login roles. Admin, shelter and user.

Registration Page



In this registration page, you can register yourself by inputting your username, first name, last name, country, province, city, address, birthdate, email, password, and your role (User or Shelter).

User

Pets Table

Species: All Breed: All Age: All Gender: All Size: All Personality: All Disabilities: All Adoption Status: All Filter

PetID	Species	PetName	Breed	Age	Gender	Size	Color	Personality	Disabilities	AdoptionStatus	ShelterName
1	Cat	Bailey	Persian	9	Male	Small	White	Bold	Deaf	Adopt	Shelter Pejaten
2	Dog	Bailey	Bulldog	10	Female	Medium	White	Hardy	Deaf	Adopt	Shelter Pejaten
3	Dog	Cooper	Bulldog	8	Female	Large	Spotted	Mild	Blind	Adopt	Shelter Pejaten
4	Cat	Charlie	Persian	6	Male	Large	Brown	Naughty	None	Adopt	Pusat Penyelamatan Satwa Tegal Alur
5	Dog	Charlie	Labrador	13	Female	Small	Black	Jolly	Blind	Adopt	Shelter Melati
6	Dog	Lucy	Labrador	10	Female	Medium	Brown	Hasty	Blind	Adopt	Shelter Melati
7	Dog	Lucy	Bulldog	11	Male	Large	Spotted	Hasty	None	Adopt	Shelter Pejaten
8	Dog	Molly	Beagle	5	Female	Small	White	Impish	None	Adopt	Pusat Penyelamatan Satwa Tegal Alur
9	Bird	Luna	Parrot	14	Male	Medium	White	Sassy	Blind	Adopt	Shelter Pejaten
10	Cat	Bella	Siamese	7	Male	Medium	White	Rash	Deaf	Adopt	Shelter Pejaten
11	Dog	Bailey	Labrador	2	Female	Medium	Black	Lonely	None	Adopt	Shelter Pejaten
12	Cat	Luna	Maine Coon	14	Female	Small	Brown	Hasty	None	Adopt	Shelter Pejaten

The user page shows the data of all available pets. The displayed pets can be filtered according to the needs, such as by the species, breed, age, gender, size, personality, disabilities and adoption status. Furthermore, by pressing the adopt button, the user can create an adoption request to the shelter the pet is currently residing in.

Shelter

Number of rows: 28

Pets Table

Filter

PetID	Species	PetName	Breed	Age	Gender	Size	Color	Personality	Disabilities	ShelterName	Action
1	Cat	Bailey	Persian	9	Male	Small	White	Bold	Deaf	Shelter Pejaten	Edit Delete
2	Dog	Bailey	Bulldog	10	Female	Medium	White	Hardy	Deaf	Shelter Pejaten	Edit Delete
3	Dog	Cooper	Bulldog	8	Female	Large	Spotted	Mild	Blind	Shelter Pejaten	Edit Delete
7	Dog	Lucy	Bulldog	11	Male	Large	Spotted	Hasty	None	Shelter Pejaten	Edit Delete
9	Bird	Luna	Parrot	14	Male	Medium	White	Sassy	Blind	Shelter Pejaten	Edit Delete
10	Cat	Bella	Siamese	7	Male	Medium	White	Rash	Deaf	Shelter Pejaten	Edit Delete
11	Dog	Bailey	Labrador	2	Female	Medium	Black	Lonely	None	Shelter Pejaten	Edit Delete
12	Cat	Luna	Maine Coon	14	Female	Small	Brown	Hasty	None	Shelter Pejaten	Edit Delete
13	Cat	Cooper	Persian	7	Male	Medium	Black	Bold	Deaf	Shelter Pejaten	Edit Delete
15	Bird	Charlie	Sparrow	47	Male	Large	Brown	Docile	None	Shelter Pejaten	Edit Delete
16	Bird	Charlie	LoveBird	43	Male	Small	Spotted	Relaxed	Blind	Shelter Pejaten	Edit Delete
17	Cat	Bailey	Maine Coon	11	Female	Large	Spotted	Naughty	Blind	Shelter Pejaten	Edit Delete

23	Cat	Charlie	Maine Coon	9	Male	Small	Black	Bashful	None	Shelter Pejaten	Edit Delete
24	Cat	Bella	Maine Coon	6	Male	Large	Black	Lax	Deaf	Shelter Pejaten	Edit Delete
25	Cat	Charlie	Maine Coon	13	Male	Medium	Black	Jolly	Blind	Shelter Pejaten	Edit Delete
26	Bird	Daisy	Sparrow	13	Female	Small	White	Relaxed	Deaf	Shelter Pejaten	Edit Delete
27	Dog	Cooper	Beagle	7	Male	Small	Brown	Quirky	Deaf	Shelter Pejaten	Edit Delete
32	Dog	Cooper	Beagle	6	Male	Medium	White	Careful	Deaf	Shelter Pejaten	Edit Delete
34	Dog	Max	Labrador	3	Female	Small	Spotted	Rash	Blind	Shelter Pejaten	Edit Delete
35	Cat	Stella	Persian	2	Male	Medium	White	Bold	Blind	Shelter Pejaten	Edit Delete
38	Dog	Bailey	Bulldog	9	Male	Medium	Black	Quiet	Blind	Shelter Pejaten	Edit Delete
39	Bird	Luna	Sparrow	2	Female	Medium	Spotted	Quiet	Deaf	Shelter Pejaten	Edit Delete
41	Bird	Lucy	LoveBird	21	Male	Small	Spotted	Brave	Deaf	Shelter Pejaten	Edit Delete
42	Bird	Molly	Sparrow	6	Female	Medium	Brown	Adamant	Deaf	Shelter Pejaten	Edit Delete
48	Cat	Bella	Siamese	8	Male	Large	White	Bold	Blind	Shelter Pejaten	Edit Delete
49	Cat	Cooper	Siamese	10	Female	Medium	Spotted	Naive	Blind	Shelter Pejaten	Edit Delete
50	Dog	Bella	Beagle	2	Male	Small	Black	Naughty	None	Shelter Pejaten	Edit Delete

Add Pet

Show a table of the shelter's pet data based on the ShelterID retrieved from the employees login. In this table, shelters are able to edit, delete or add pets.

Admin

Admin Page

Pet Finder

Delete Add New User

UserID	Username	First name	Last name	Email	Password	Country	Prov
2	ERONE64	Eronine	Onerino	Ononino64@gmail.com	999999999	Indonesia	DKI JAKA
3	Manison	Manisons	Lanisons	manlan@gmail.com	e3ceb5881a0a1fdaad01296d7554868d	Indonesia	DKI Jakc
4	AdminA	Monte	Carlo	adminmonte@gmail.com	cad9289d2c0b2428bbac9837194e8823	Indonesia	DKI JAKA
5	SmallHole	Sulthan	Gita	gitaloveazka22@gmail.com	dd4b21e9ef71e1291183a46b913ae6f2	Indonesia	DKI Jakc

	Country	Province	City	Address	Birthdate	Role	
	Indonesia	DKI JAKARTA	Jakarta Utara	Jl. Mangga Dua Raya, RT.11/RW.5, Ancol, Kec. Pademangan	1993-08-04	User	
96d7554868d	Indonesia	DKI Jakarta	Jakarta Barat	Jl. Lkr. Luar Barat No.8, RT.9/RW.6, Duri Kosambi, Kecamatan Cengkareng	2002-02-12	User	
:9837194e8823	Indonesia	DKI JAKARTA	Jakarta Utara	Jl. Pluit Karang Manis X no. 21	1989-06-07	Admin	
;b913ae6f2	Indonesia	DKI Jakarta	Jakarta Barat	Wang Plaza, Jl. Panjang No.kav 17, RT.14/RW.7, Kedoya Utara, Kec. Kb. Jeruk, jakarta, Daerah Khusus Ibukota Jakarta 11520	2001-09-11	User	

In the admin page, the admin is able to edit, delete and add users. Furthermore, with our extra feature to wipe out our entire user base.

V. Database Security

explain how you design your database security (if any), how do you assign your database user roles.

```php
session_start();

if(isset($_POST['submit'])) {
    $Email = mysqli_real_escape_string($connection, $_POST['Email']);
    $Password = md5($_POST['Password']);

    $select = "SELECT * FROM users WHERE Email = '$Email' AND Password = '$Password'";
    $result = mysqli_query($connection, $select);

    if(mysqli_num_rows($result) > 0) {
        $row = mysqli_fetch_array($result);

        // Store the retrieved user ID in the session
        $_SESSION['UserID'] = $row['UserID'];

        // Check the user role and set session accordingly
        if($row['Role'] == 'Admin') {
            $_SESSION['admin_name'] = $row['Username'];
            header('location:admins.php');
        } elseif($row['Role'] == 'User') {
            $_SESSION['User_name'] = $row['Username'];
            header('location:user.php');
        } elseif($row['Role'] == 'Shelter') {
            $_SESSION['shelter_name'] = $row['Username'];
            header('location:shelter2.php');
        }
    } else {
        $error[] = 'Incorrect email or password!';
    }
}
```

We use an if function where it will start when the user presses the submit form which is the login now button, in the if function, it will first check whether the user login actually exist in the database and if it matches the data. If the data matches correctly with the users in the database, it will then check the role whether it's an admin, shelter or user role. After checking the role, it will directly redirect them to their specific page while saving their username and user ID.

Admin: The admin role is capable of editing, deleting and adding the users data.

Shelter: The shelter role is used by the shelter managers themselves and would be used by them to add or delete any data within the database regarding their specified shelter. They will not be able to modify any data regarding the users themselves.

User: The user role will have the most restricted access to the website as they are only capable of viewing and filtering the data of the pets alongside their other details, however, they are not capable of removing or adding data. The user role will only be able to choose their specified pet and adopt them which would update the pet's availability to be adopted.

VI. Github Repo Link

https://github.com/KELASU/Database-Final