Chapter 3

3.1 Introduction to Design

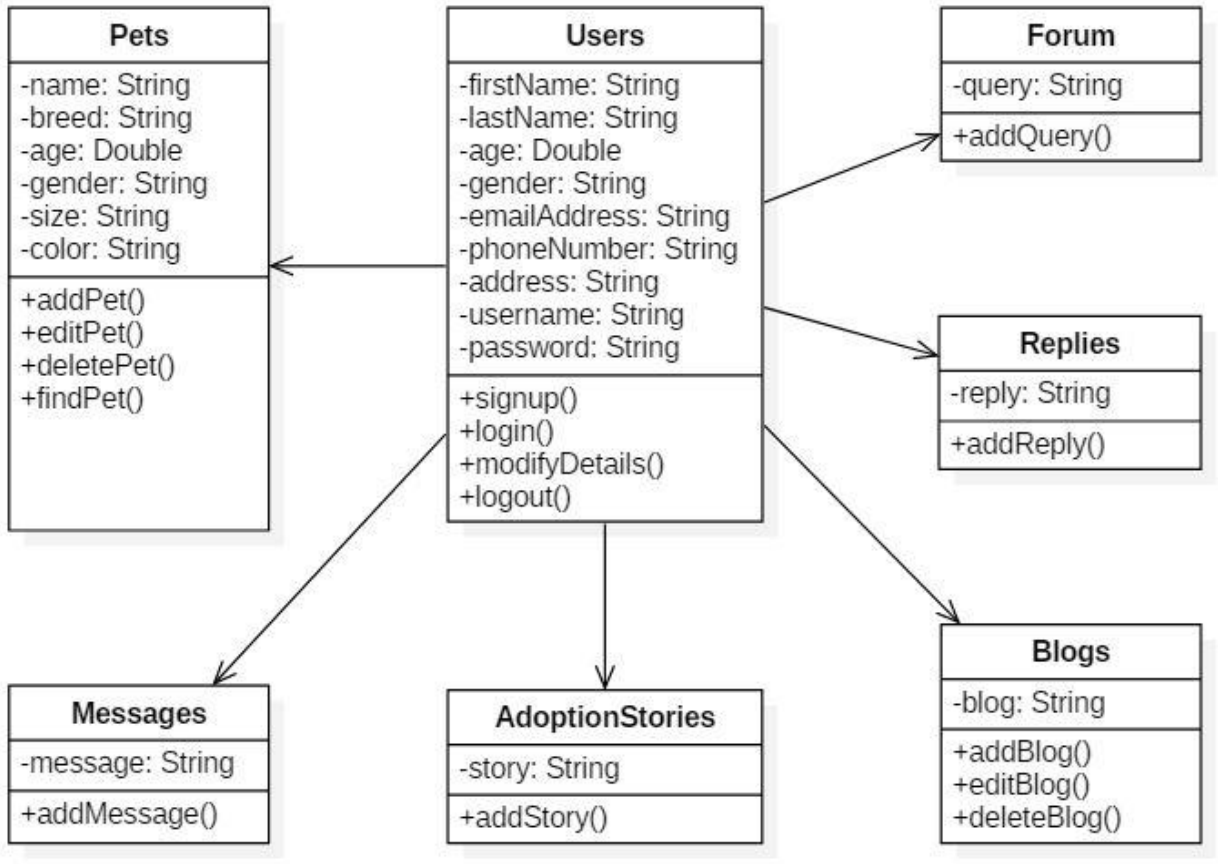
Design can be defined as the procedure of dividing the functions of the program into classes and defining their interactions and the way the data will be stored. It guides us to build the system in an effective way.

3.2 Structural Design/Model

The type of modelling that displays the organization of components of a system and their relationships is called Structural modelling. The model represents static diagrams like class diagrams that does not change over time.

3.2.1 Final Class Diagram

Class diagram is a static model that illustrates the structure and architecture of the system to be built. It also describes the relationship between the classes including their properties and functions.



In the class diagram shown above, the final classes are represented including the relationships between them.

3.3 Behavioral Model

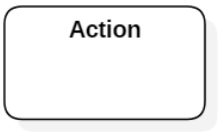
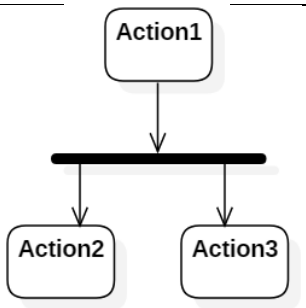
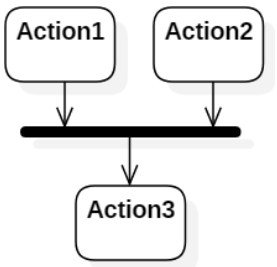
Behavioral model represents the actual working of the system with the help of dynamic behavior of objects. It is dynamic in nature and changes over time.

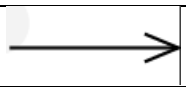
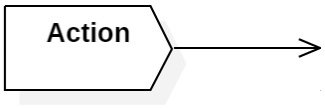
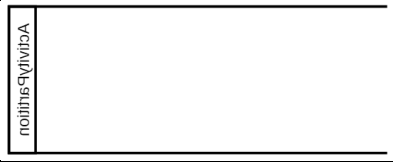
3.3.1 Activity Diagram

Activity diagram is a part of behavioral model that changes in relation the way the user interacts with the system. Primarily, it is a flowchart showing various possible ways the user can interact within the system.

I have used activity diagram because it helps to perceive the flow of system with the help of logics.

The notations used in the activity diagram are as follows:

Notation Used	Notation	Description
Initial		Represents the start of activity
Final		Represents the end of activity
Action		Represents the activity
Decision		Represents conditions
Fork		Splits one activity into two concurrent activities
Join		Combines two concurrent activities into one activity

Control Flow		Represents the flow of processes
Send Signal		Represents message sent within the system
Receive Signal		Represents message received from Send Signal
Swimlane		Divides activities in relations to sub-processes

a. Activity Diagram for Admin

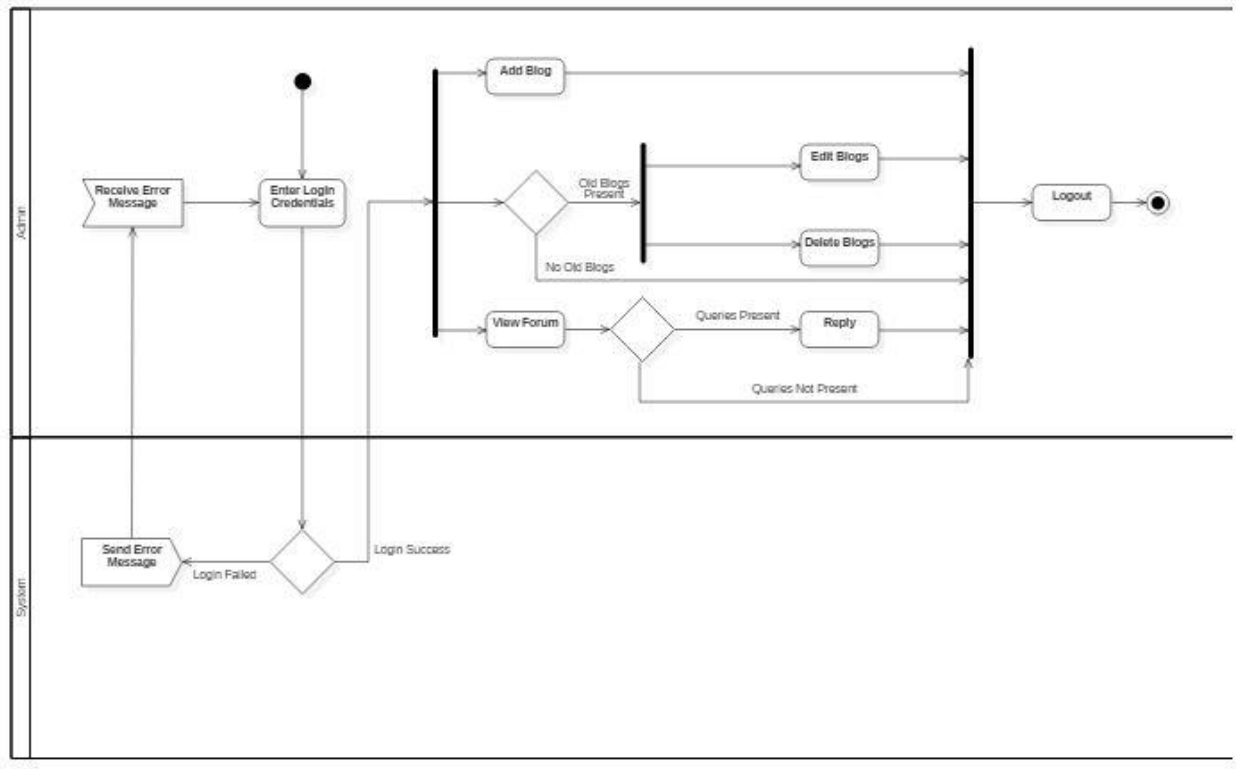


Figure 1:Activity Diagram for Admin

First of all, the admin enters the login credentials. The system validates the login request and if the login attempt is successful, the admins is redirected to the system interface and if not, the system sends an error message which is then received by the admin and is prompted to enter the login details again. In the interface, the admin can add blogs, edit and delete old blogs if present and reply to forum queries if present. Lastly, the admin can logout and end the process.

b. Activity Diagram for Guardian

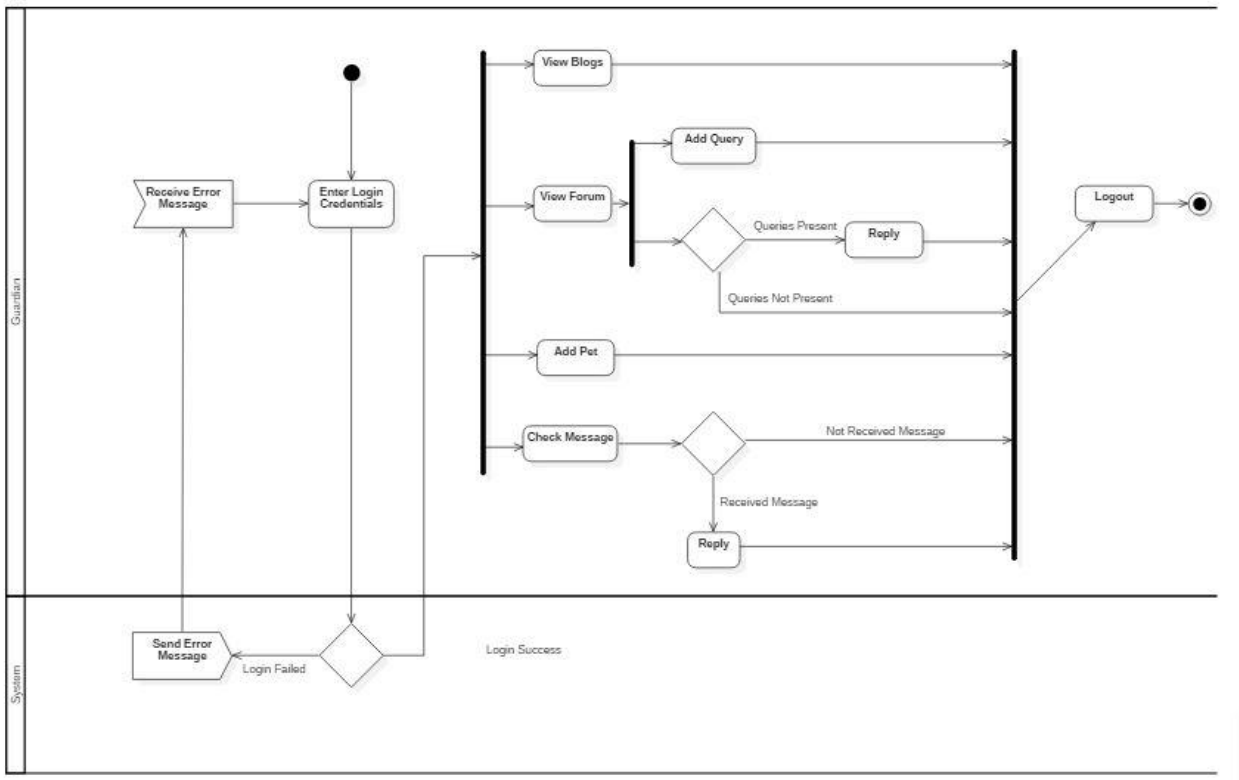


Figure 2: Activity Diagram for Guardian

First of all, the user enters the login credentials. The system validates the login request and if the login attempt is successful, the admins is redirected to the system interface and if not, the system sends an error message which is then received by the admin and is prompted to enter the login details again. In the interface, the guardian can view blogs, forum and reply to queries if present. He/She can also register pet for adoption and check messages from adopters and reply to them if present. In the end, the guardian can logout and end the process.

c. Activity Diagram for Adopters

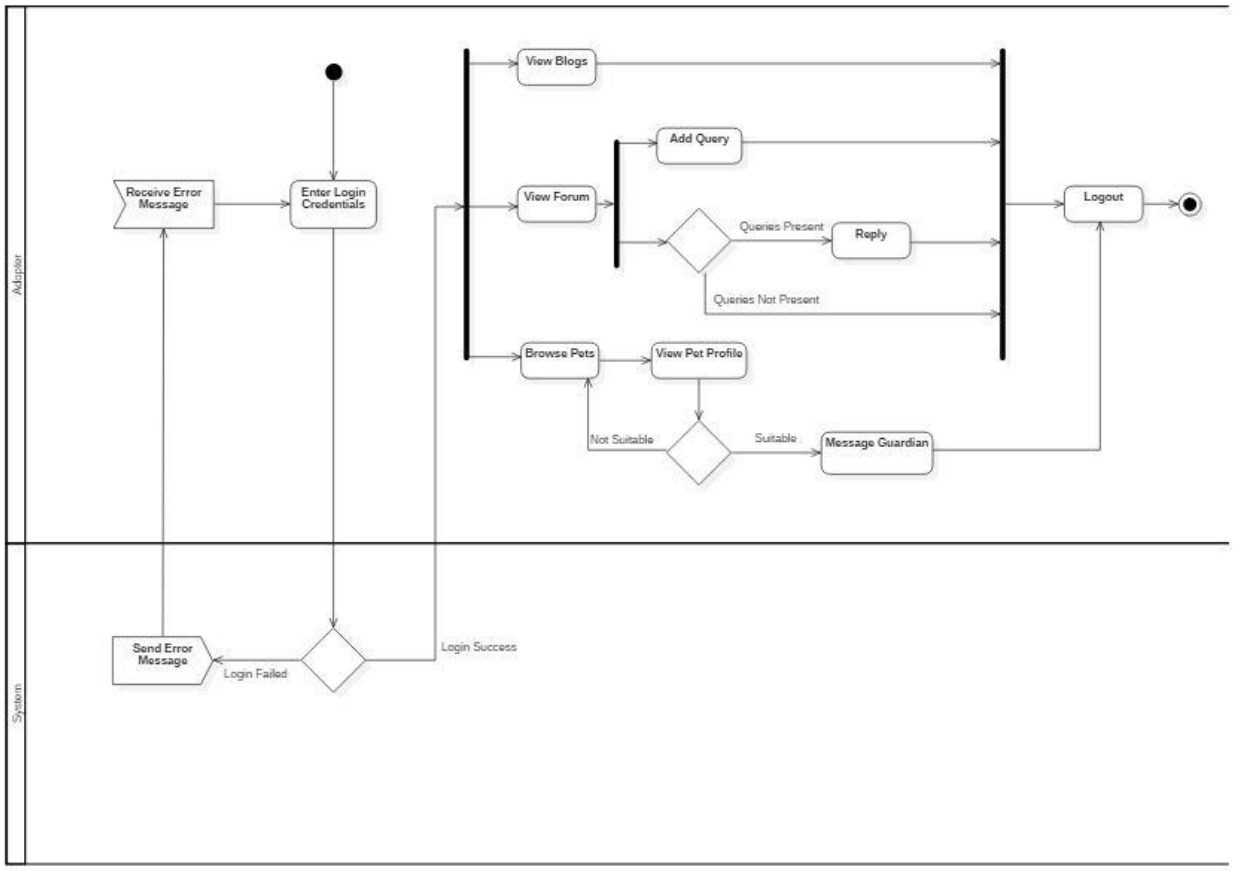


Figure 3: Activity Diagram for Adopters

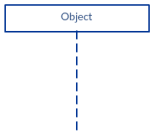
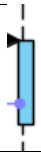
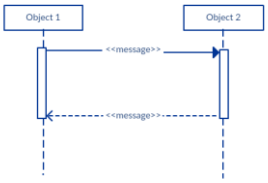
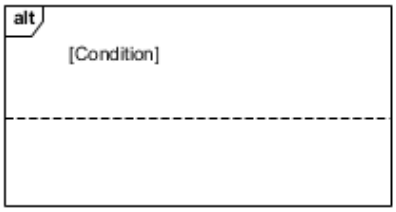
First of all, the adopter enters the login credentials. The system validates the login request and if the login attempt is successful, the adopter is redirected to the system interface and if not, the system sends an error message which is then received by the admin and is prompted to enter the login details again. In the interface, the admin can view blogs, forum and reply to queries if present. He/She can browse pets and view their profile and inquire about the pet with its guardian by messaging them. In the end, the guardian can logout and end the process.

3.3.2 Sequence Diagram

Sequence diagram is a type of dynamic behavioral diagram representing the interactions between objects of various classes and the lifeline of those objects. It is generally used to organize complex systems with the help of sequence of objects.

I have used sequence diagram because it depicts the interaction among objects in a sequential order to finish a task. It helps us to visualize the system.

The notations used in the sequence diagram are as follows:

Notation Used	Notation	Description
Actor		Invoker of the interaction
Lifeline		Represents different objects or parts interacting with each other during the sequence
Activation Bar		Indicates that an object is active
Message Arrow		Represents communication between objects
Alternate Frame		Choice to be made between two or more than two message sequence

a. Sequence Diagram for Login and Registration

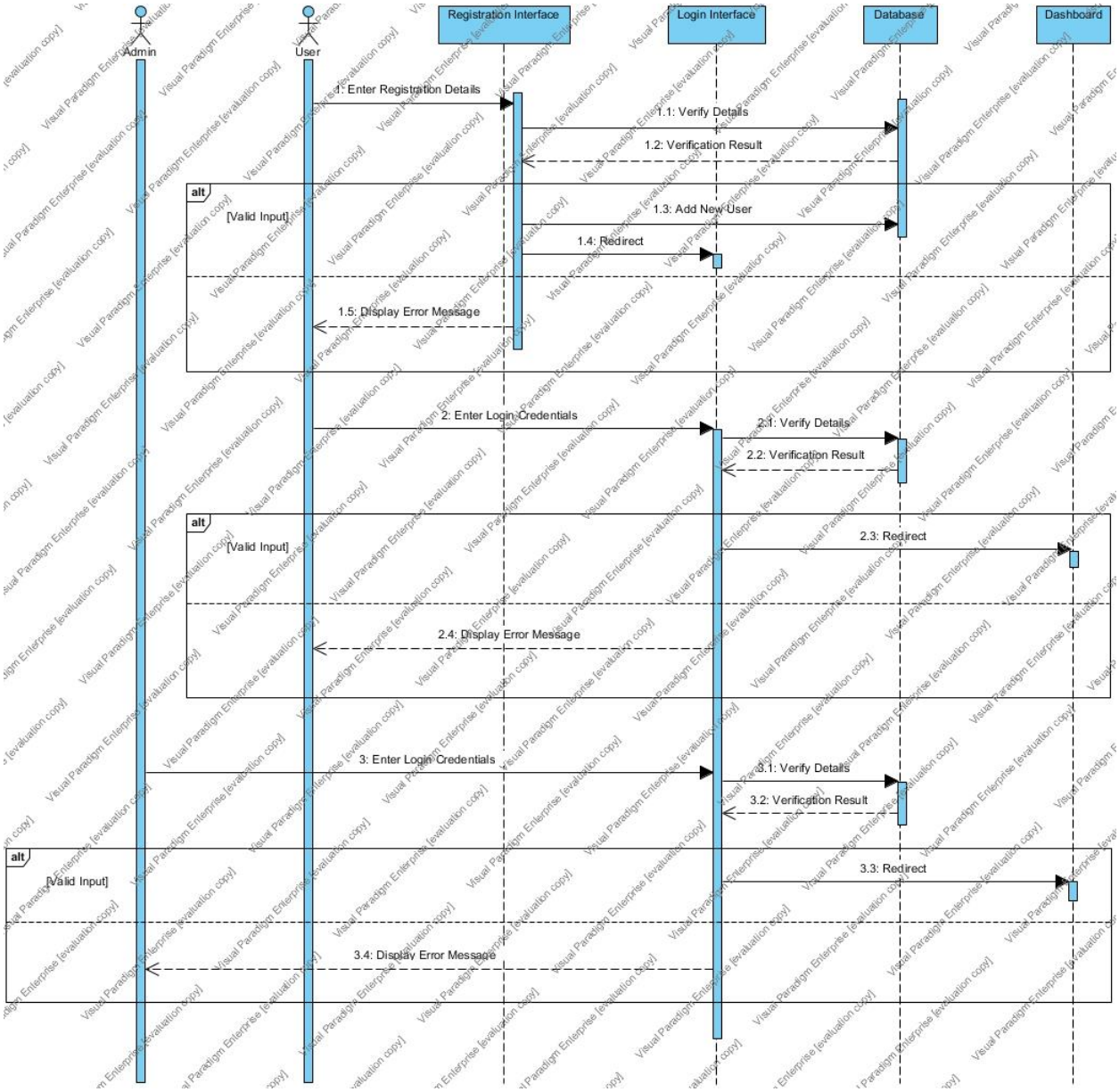


Figure 4:Sequence Diagram for Registration and Login

The diagram above represents the sequence diagram from registration and login. A guest user can enter registration details via the registration interface. The details are verified by the database and if the details are valid, a new user is created in the database and the user is redirected to the login interface. However, if the details are not valid, an error message is displayed to the user. The user and admins can provide login credentials via the login interface which is also verified by the database. If the details are verified, the users are redirected to their respective dashboard and if not, an error message is displayed to the users.

b. Sequence Diagram for Admin

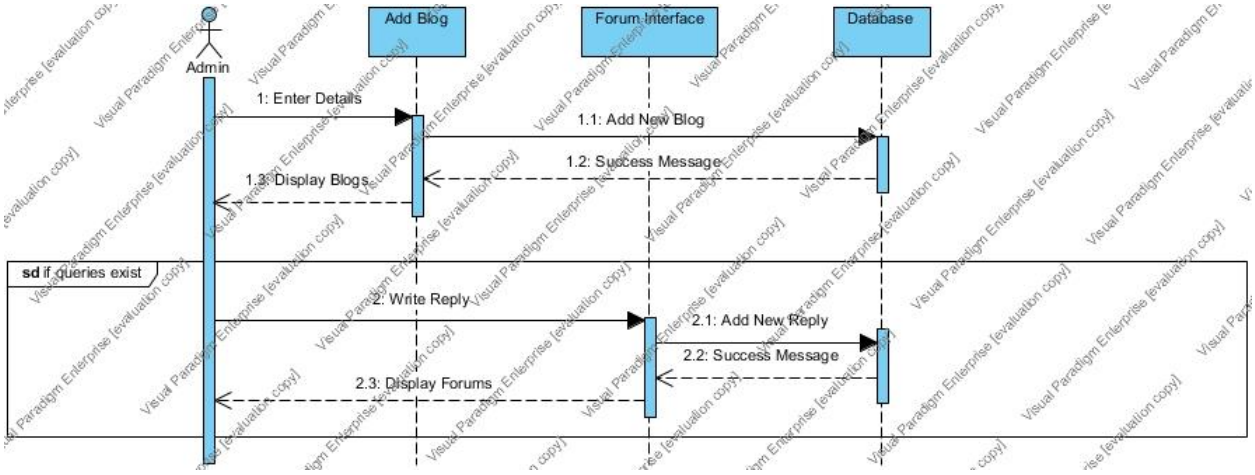


Figure 5: Sequence Diagram for Admin

The admin can add a blog via the add blog interface. The interface prompts the database to add a new blog and a success message is passed back and it displays blogs to the admin. The admin can also write a reply to the queries in the forum if any queries exist. The forum interface prompts the database to add a new reply and a success message is passed back and it displays forum contents to the admin.

c. Sequence Diagram for Guardian

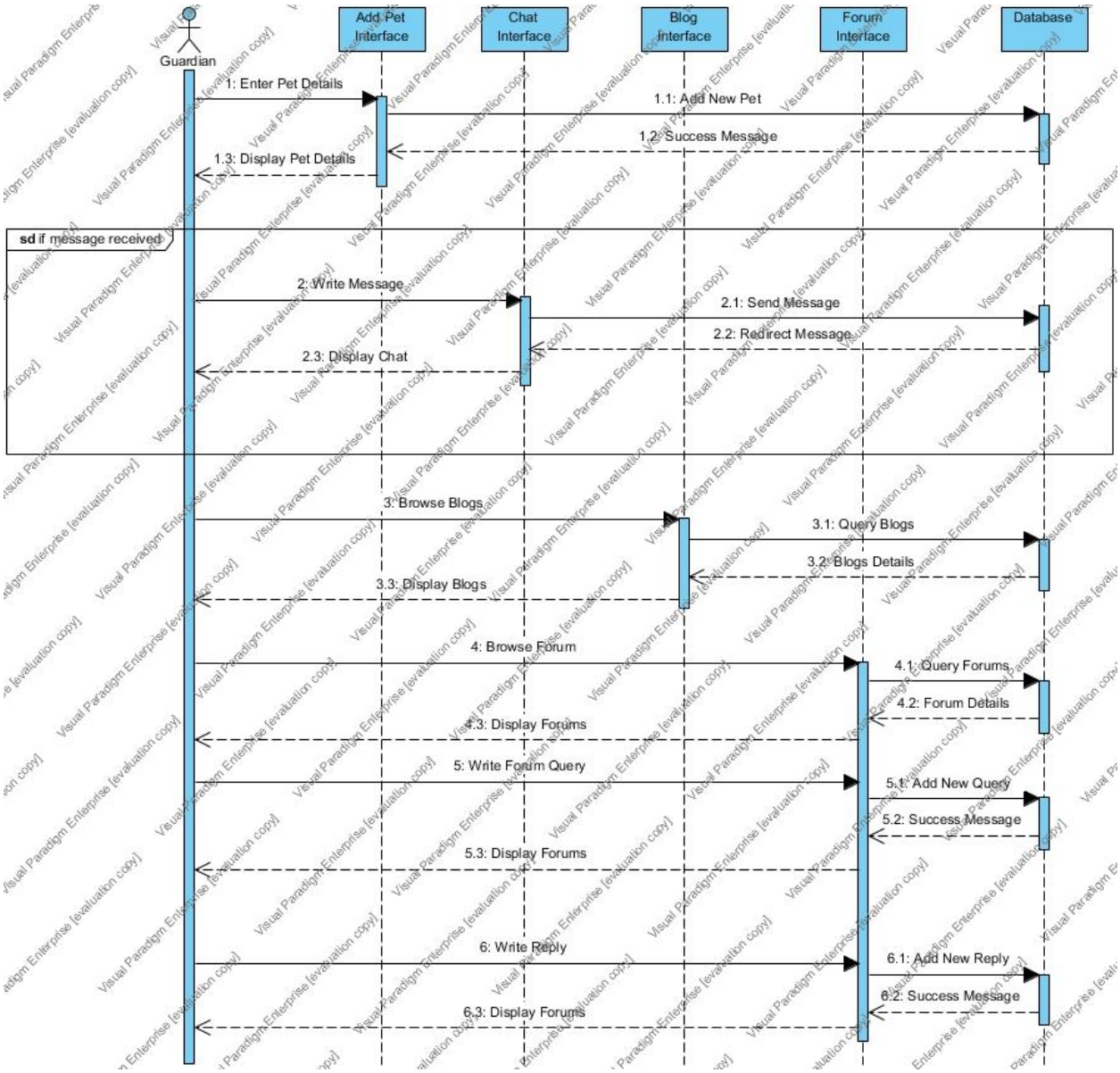


Figure 6: Sequence Diagram for Guardian

The guardian can add a pet via add pet interface by entering the details. The interface induces the database to add a new pet and the database replies a success message while the pet details are displayed to the user. The guardian can also reply to a message if received. The chat interface sends the message to the database. He/She can also browse blogs and forum while he/she can also add a new query or reply to one in the forum.

d. Sequence Diagram for Adopter

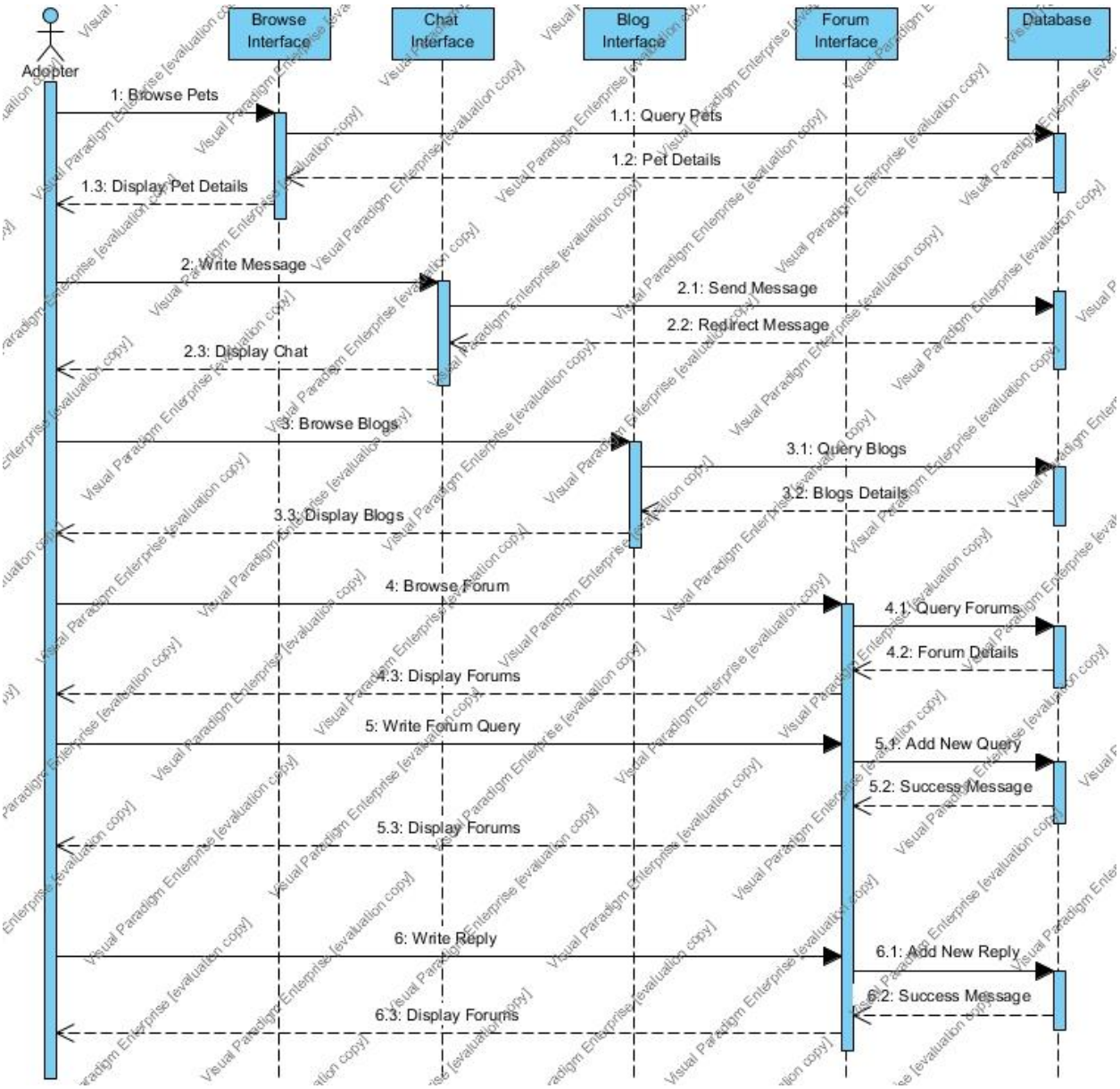


Figure 7:Sequence Diagram for Adopter

The adopter can browse pets via the system interface which fetches and displays the pet details from the database. The adopter can also write a message to the guardian. The chat interface sends the message to the database. He/She can also browse blogs and forum while he/she can also add a new query or reply to one in the forum.

3.4 Database Model

Database model is a variation of data model which represents the logical structure of the database to be used in the system. It primarily determines the way data can be stored and manipulated.

3.4.1 Data Dictionary

The data about data stored in the database is known as data dictionary. They contain information like attributes, datatype, length, constraints, etc.

The data dictionary of my system are as follows:

Database Tables

Name
Users:Entity
Pets:Entity
Messages:Entity
AdoptionStories:Entity
Blogs:Entity
Forums:Entity
Replies:Entity

Users Table

Column Name	Data Type	PK/FK	Nullable
User_id	Int	PK	No
Usertype	Varchar(10)		Yes
First_name	Varchar(15)		Yes
Last_name	Varchar(15)		Yes
email	Varchar(30)		Yes
phoneNumber	Varchar(10)		Yes
address	Varchar(50)		Yes
Username	Varchar(15)		Yes
Password	Varchar(20)		Yes

Pets Table

Column Name	Data Type	PK/FK	Nullable
pet_id	Int	PK	No
type	Varchar(3)		Yes
name	Varchar(10)		Yes
breed	Varchar(10)		Yes
age	Varchar(6)		Yes
gender	Varchar(6)		Yes
size	Varchar(6)		Yes
color	Varchar(10)		Yes
Owner_id	int	FK	No

Blogs Table

Column Name	Data Type	PK/FK	Nullable
Blog_id	Int	PK	No
Blogtitle	Varchar(50)		Yes
blog	Varchar(500)		Yes
date	date		Yes

AdoptionStories Table

Column Name	Data Type	PK/FK	Nullable
story_id	Int	PK	No
Story	Varchar(500)		Yes
User_id	Int	FK	No

Messages Table

Column Name	Data Type	PK/FK	Nullable
Message_id	Int	PK	No
message	Varchar(255)		Yes

UserMessages Table

Column Name	Data Type	PK/FK	Nullable
Message_id	Int	FK	No
User_id	Int	FK	No
Time	Time		Yes
Date	Date		Yes

Forums Table

Column Name	Data Type	PK/FK	Nullable
forum_id	Int	PK	No
Forum_title	Varchar(50)		Yes
Forum_content	Varchar(500)		Yes

UsersForums Table

Column Name	Data Type	PK/FK	Nullable
forum_id	Int	FK	No
User_id	Int	FK	No
date	date		Yes

Replies Table

Column Name	Data Type	PK/FK	Nullable
reply_id	Int	PK	No
reply	Varchar(500)		Yes
Forum_id	Int	FK	No

UserReplies Table

Column Name	Data Type	PK/FK	Nullable
reply_id	Int	FK	No
User_id	Int	FK	No
Date	Date		Yes

3.4.2 ER Diagram

ER Diagram is a type of database that shows the relationships between entities of the system along with their attributes.

The ER Diagram for my system is given below:

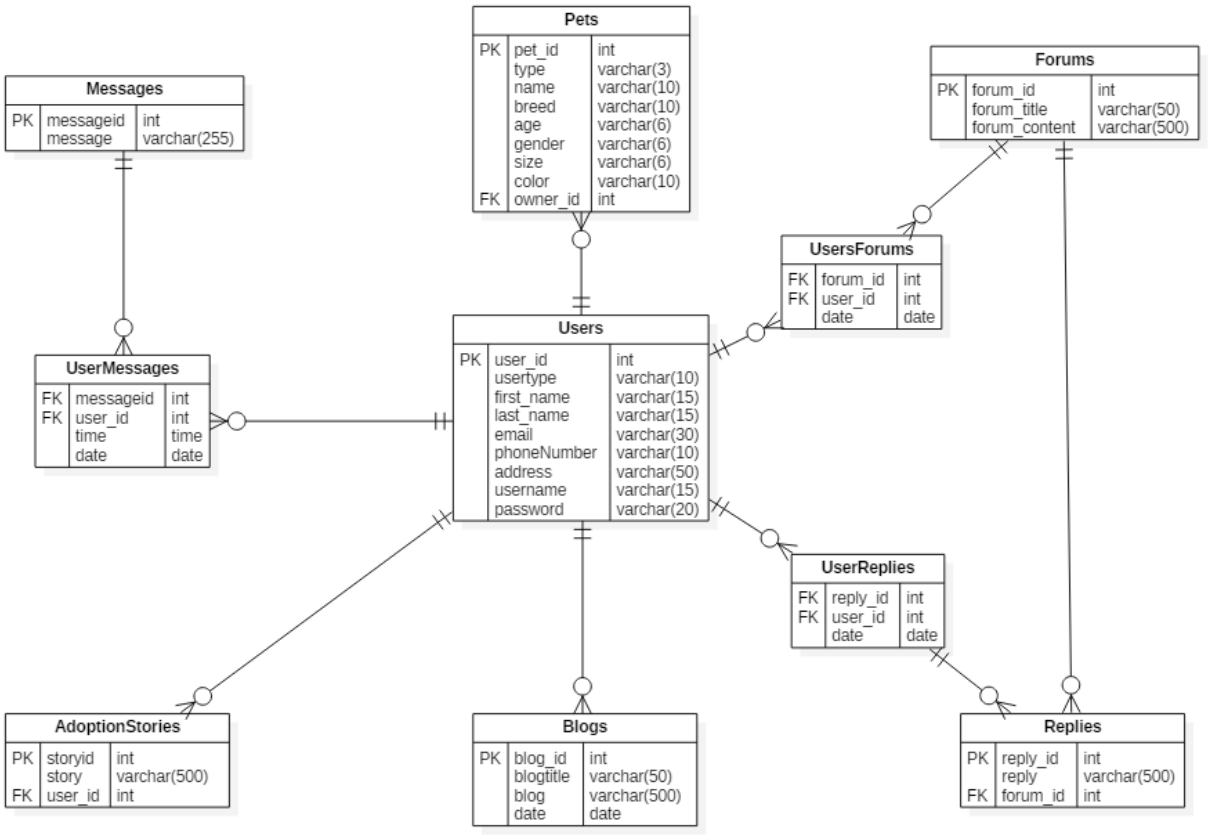


Figure 8: ER Diagram

The above given ER Diagram represents all the entities and attributes of my database along with the constraints and datatypes.

3.5 Architectural Model

An architectural model is a scaled-down version of the system that captures various and essential properties of the system.

My system follows MVC pattern and 3 tier architecture where:

Model: Responsible for maintaining the data of the application. It retrieves and stores data in response to the View.

View: It is the user interface that displays data to the user with the help of Model. It also enables the users to modify the data.

Controller: Responsible for handling user input and interact with model. It receives input from view, validates it and performs business operations modifying the data model.

I have used this pattern because:

i. It supports fast and parallel development.
ii. It can provide multiple views according to the type of users.
iii. It is very easy to make modifications.

3.6 UI Modelling

UI Modelling is a type of development technique that involves users to use mock-up User Interface of a system. It saves time and cost of directly writing code by providing prototypes.

3.6.1 Prototyping

Prototyping is a technique that provides the stakeholders an idea of what the final product will look like in prior to spending time and money to finalize the product. It helps the stakeholders to evaluate the system, ensure that it does what is intended and suggest improvements.

I have developed prototypes for my system using a prototyping tool named Adobe XD (eXperience Design). I have chosen this tool because:

i. Encourages consistent design because of multiple pages in a single screen.
ii. Supports live preview and across multiple operating systems and devices.
iii. Very easy to use.

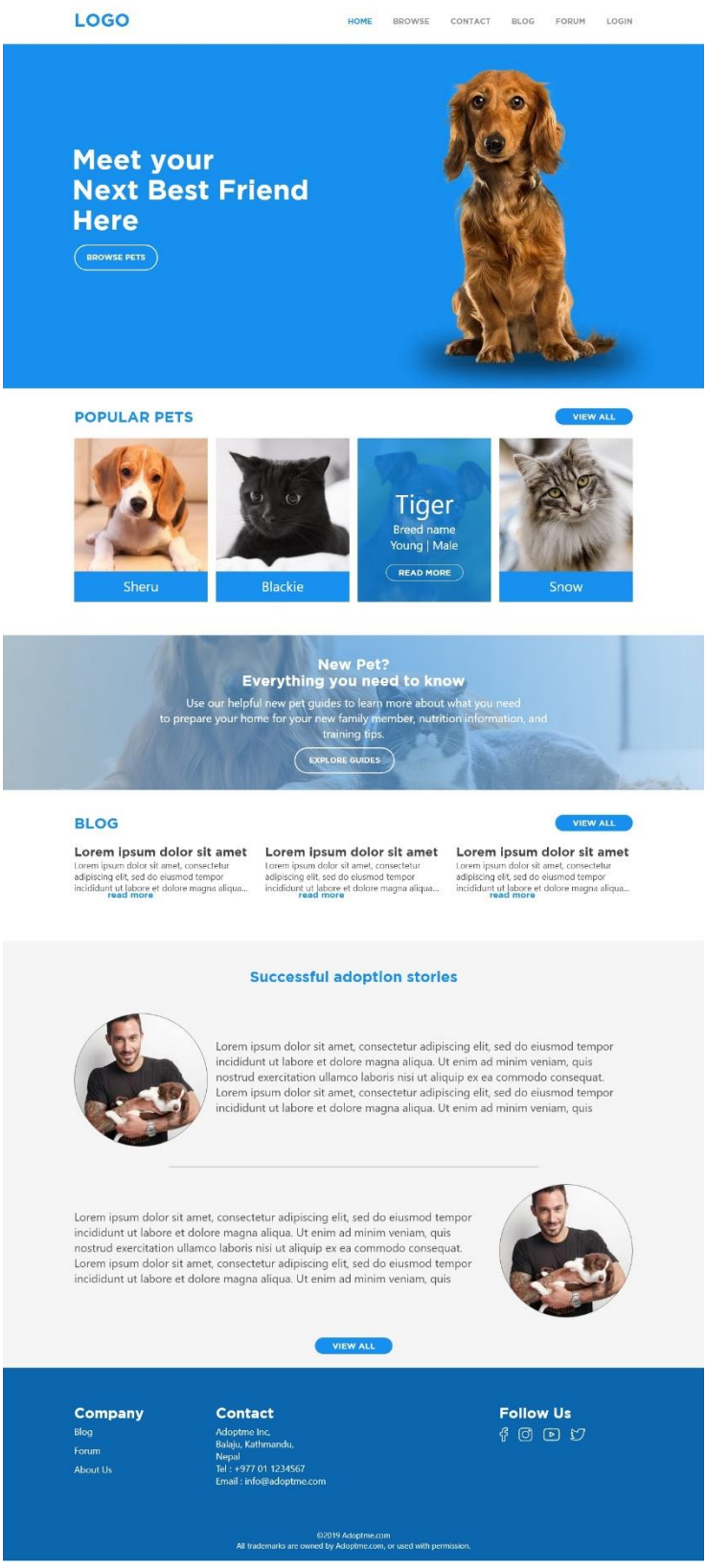


Figure 9: Home Page

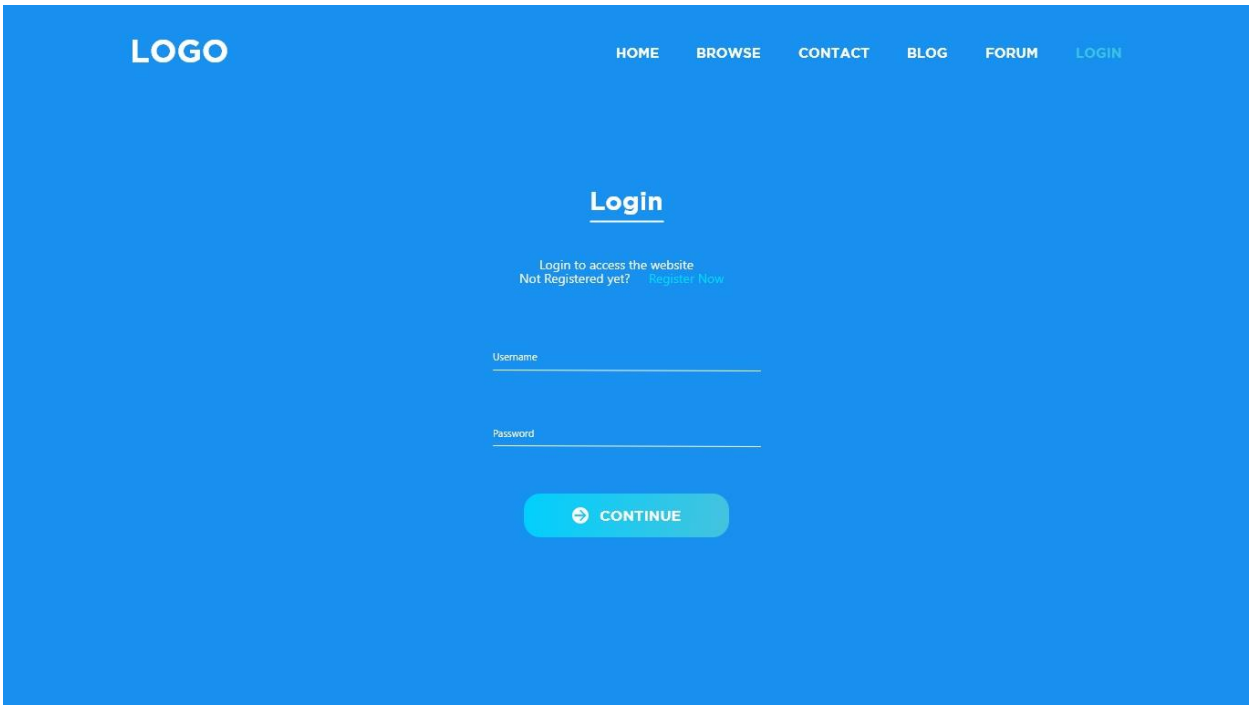


Figure 10: Login Interface

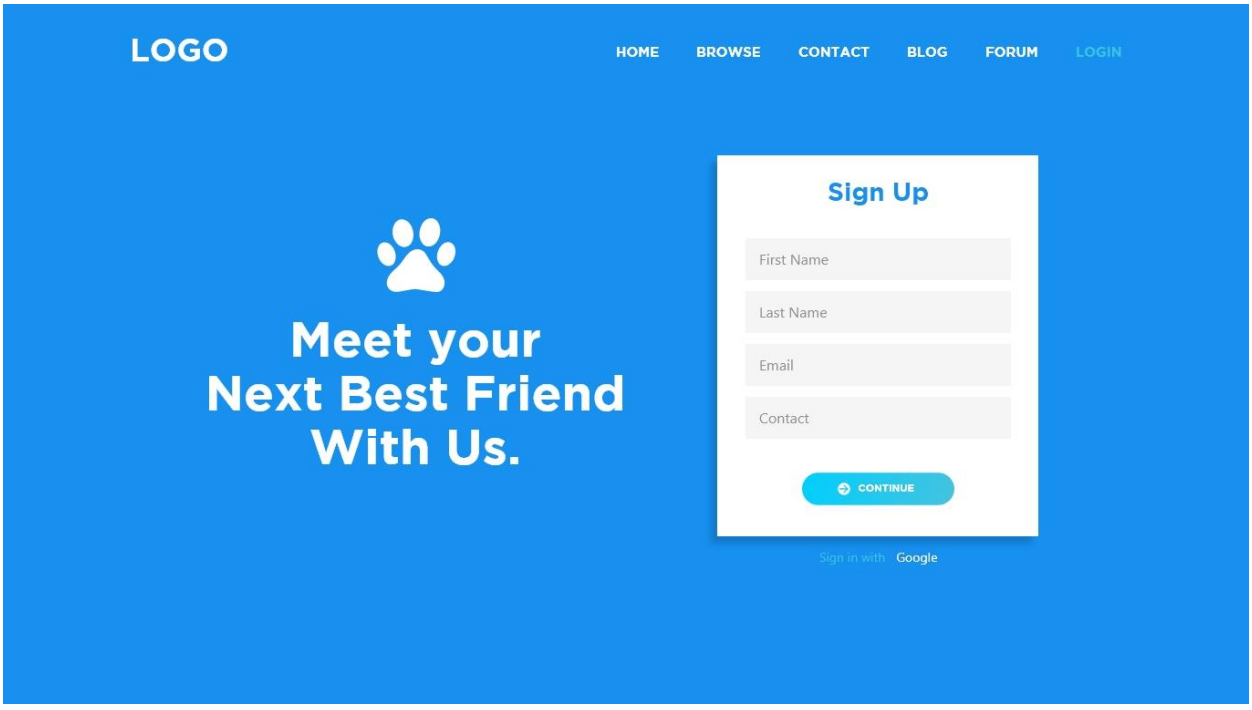


Figure 11: Register User Interface

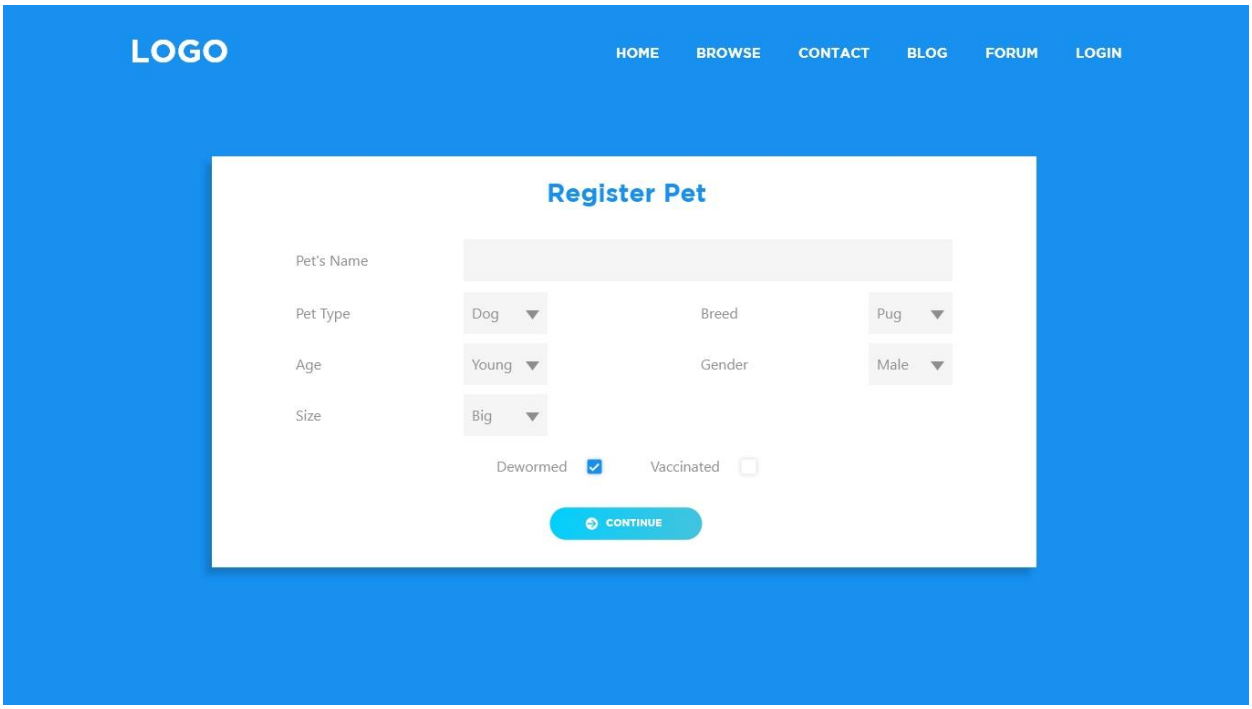


Figure 12: Register Pet Interface

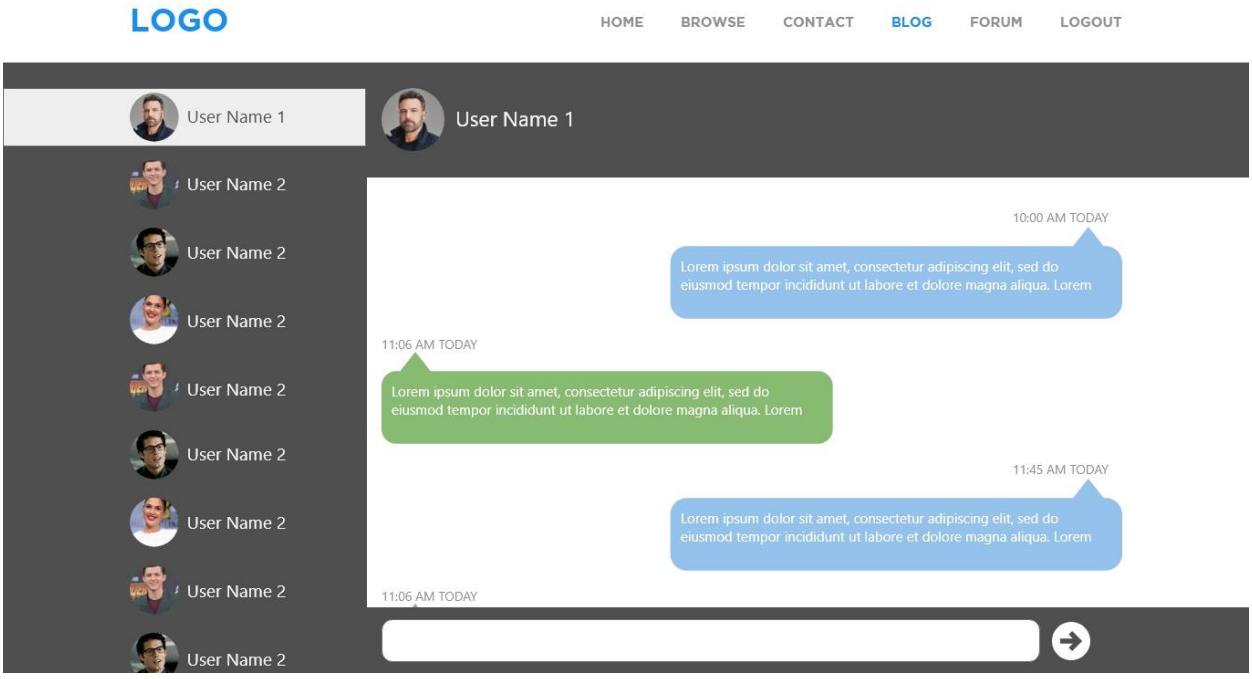


Figure 13: Chat Interface

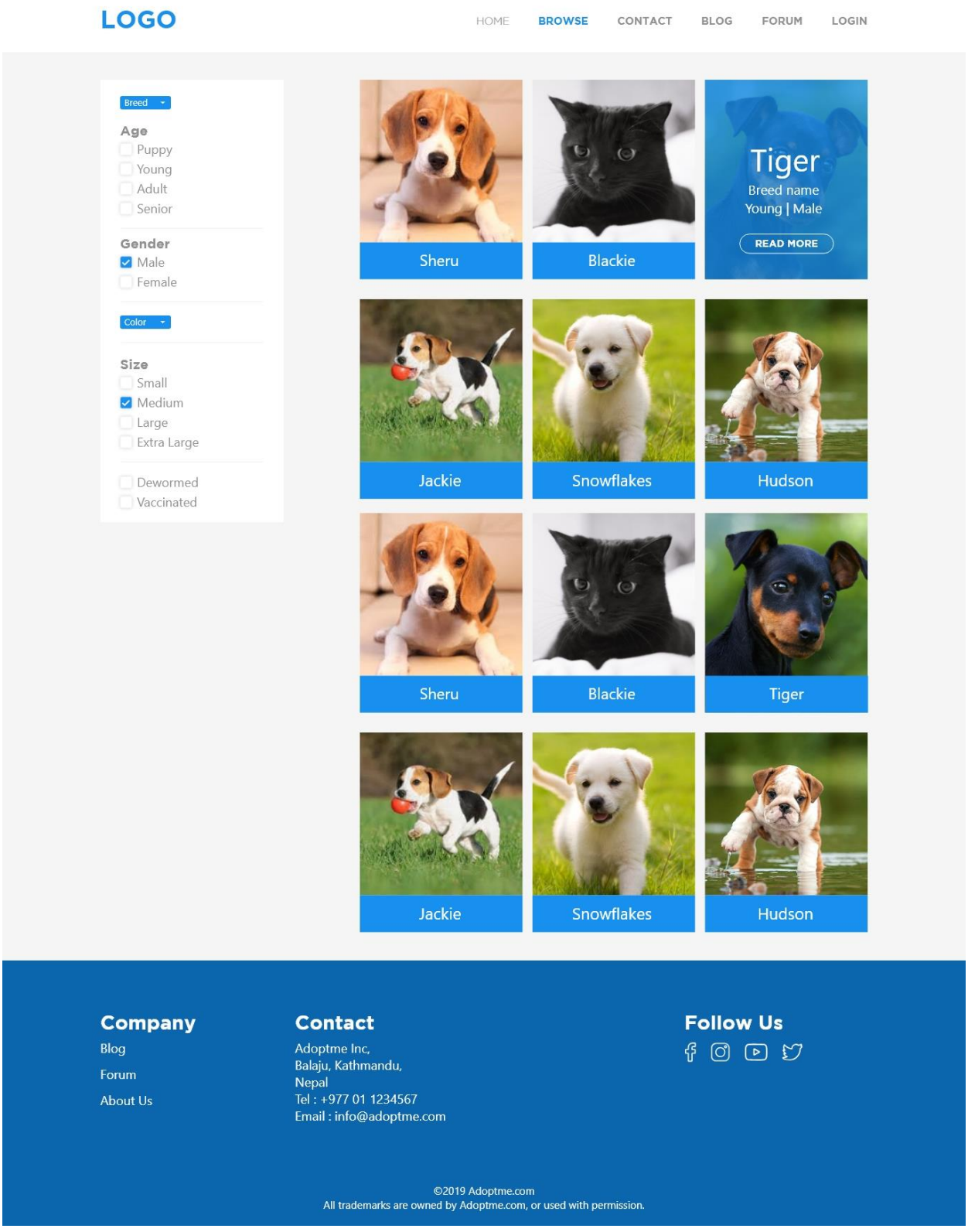


Figure 14: Browse Interface

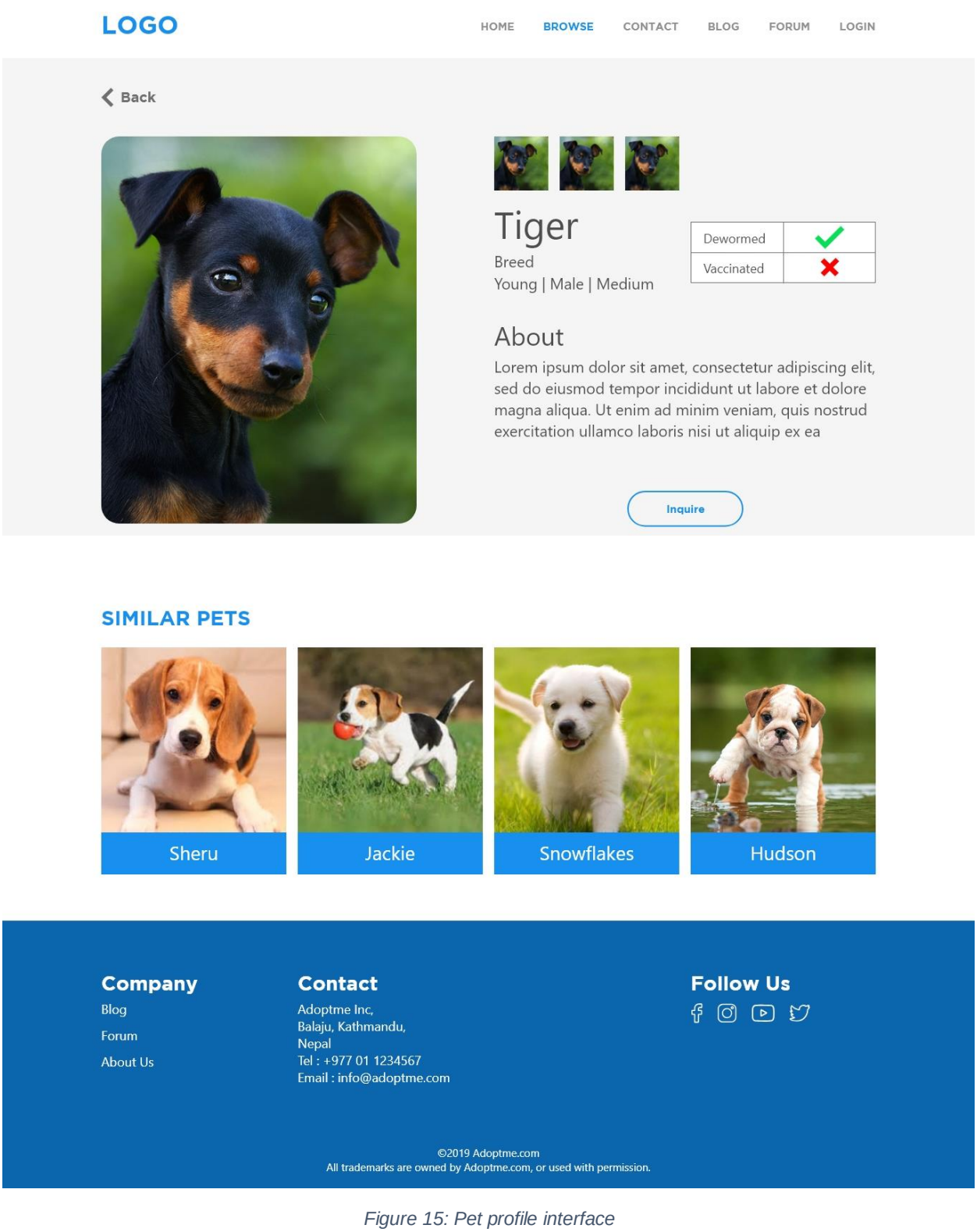


Figure 15: Pet profile interface

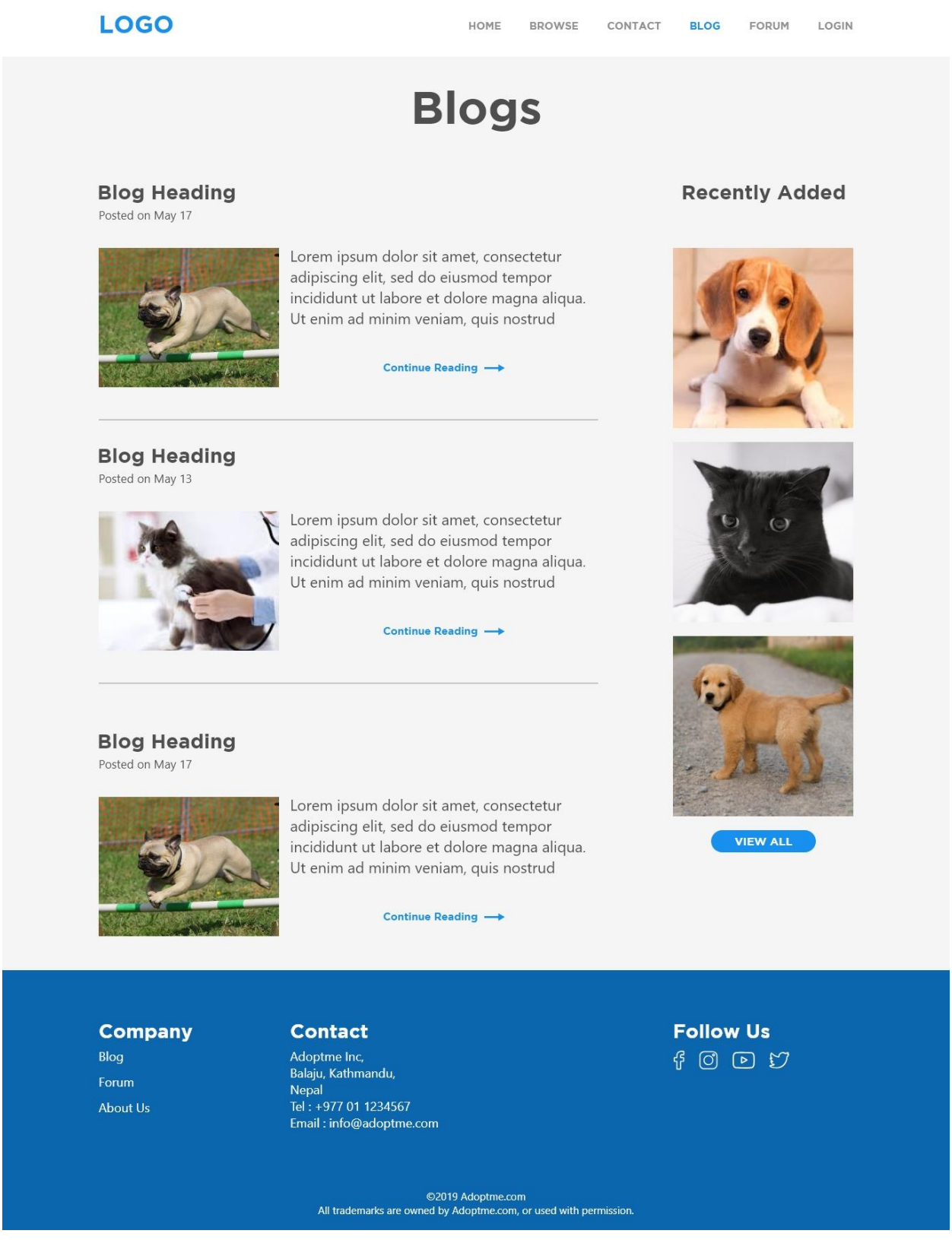


Figure 16: Blog Interface

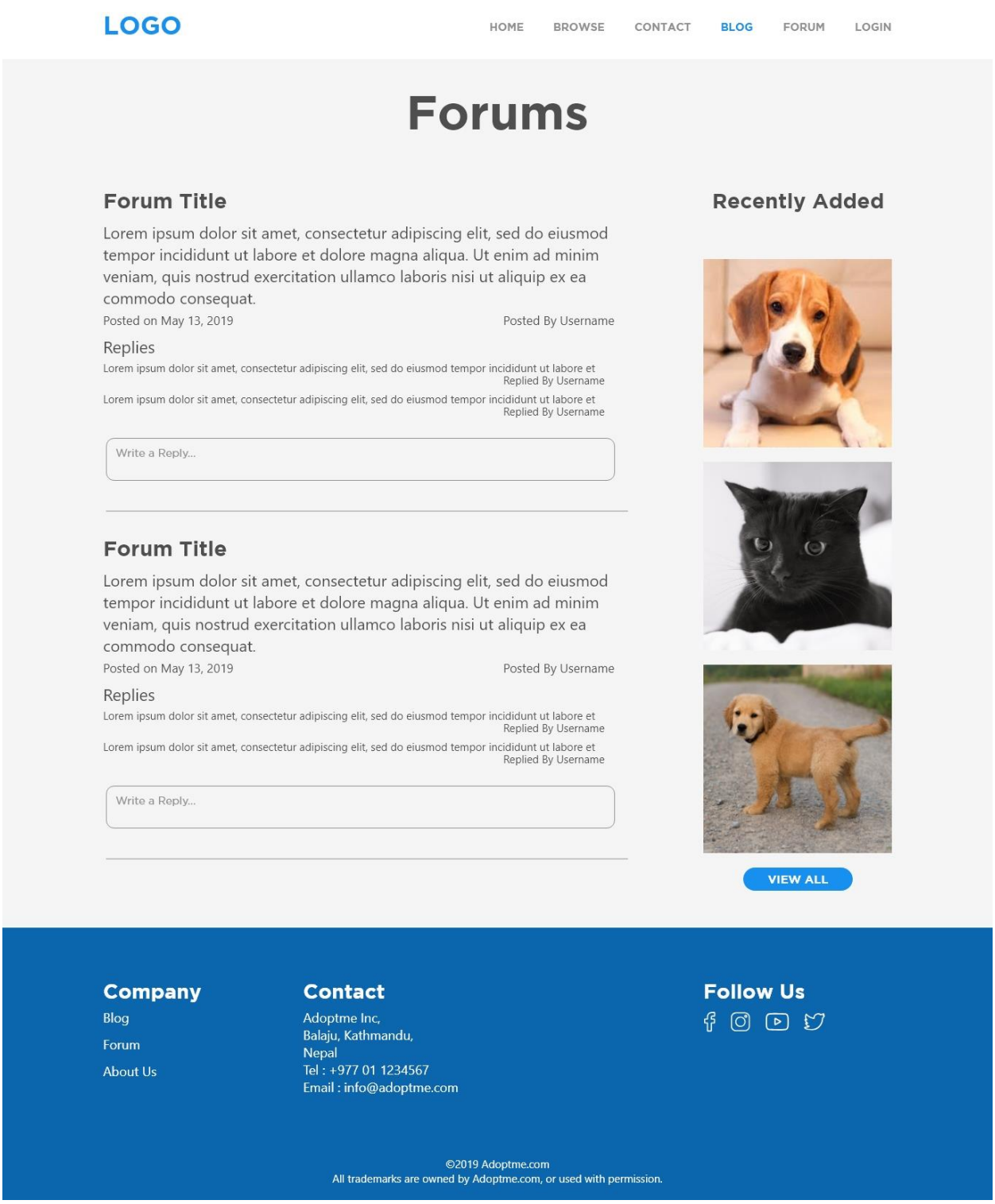


Figure 17: Forum Interface