

**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG**

-----oo-----



BÁO CÁO MÔN HỌC LẬP TRÌNH MẠNG CĂN BẢN

-&-

**ĐỀ TÀI: ỨNG DỤNG NHẬN NUÔI THÚ CUNG VÀ TÌM THÚ CUNG
BỊ THẤT LẠC**

Giảng viên hướng dẫn : **ThS. TRẦN HỒNG NGHI**

Sinh viên thực hiện:

- | | |
|-----------------------|----------------|
| 1. NGUYỄN ĐỨC TÀI | MSSV: 21521395 |
| 2. TRẦN MINH DUY | MSSV: 21522010 |
| 3. VÕ THỊ QUỲNH NHƯ | MSSV: 21522434 |
| 4. NGUYỄN HOÀI PHƯƠNG | MSSV: 21520408 |

Lớp : **NT109.N21.ANTT**

Khoá : **16**

TP. Hồ Chí Minh, tháng 06 năm 2023

MỤC LỤC

CHƯƠNG 1. GIỚI THIỆU	1
1.1. Giới thiệu chung.....	1
1.2. Mục tiêu và chức năng chính	1
1.3. Công nghệ sử dụng	1
1.4. Đóng góp đê tài	2
CHƯƠNG 2. CƠ SỞ LÝ THUYẾT	2
2.1. Kết nối TCP đê chat giữa server- client.....	2
2.2. Giao tiếp với Email Server	3
2.3. Bảo vệ dữ liệu	4
2.4. Lập trình bắt đồng bộ	5
2.5. Real-time Database Firebase	5
CHƯƠNG 3. PHÂN TÍCH THIẾT KẾ HỆ THỐNG.....	6
3.1. Sơ đồ phân rã chức năng.....	6
3.2. Thiết kế Database.....	6
3.3. Giao diện người dùng	8
3.3.1. Trang Login.....	8
3.3.2. Trang Loading.....	9
3.3.3. Trang đăng kí	10
3.3.4. Trang Home	11
3.3.5. Trang post thông tin	12
3.3.6. Trang nhận nuôi	13
3.3.7. Trang tìm kiếm.....	14
3.3.8. Trang cài đặt.....	15
3.3.9. Giao diện chat tổng	16

3.3.10. Giao diện chat giữa 2 users	16
3.3.11. Trang forgot password	17
3.3.12. Trang xác nhận mã otp khi yêu cầu tạo lại mật khẩu.....	17
3.4. Timeline	18
3.4.1. Giai đoạn 1: Phân tích đề tài	18
3.4.2. Giai đoạn 2: Thiết kế giao diện + Code	18
3.4.3. Giai đoạn 3: Xử lý Code logic BackEnd	18
3.4.4. Giai đoạn 4: Viết báo cáo.....	19
CHƯƠNG 4. HIỆN THỰC ĐỀ TÀI:	19
CHƯƠNG 5. KIỂM THỬ	32
5.1. Kiểm thử chức năng đăng kí	32
5.2. Kiểm thử chức năng đăng nhập	33
5.3. Kiểm thử chức năng trang đăng thông tin (Form Post)	34
5.4. Kiểm thử chức năng trang nhận nuôi thú cưng (Form Adopt)	36
5.5. Kiểm thử chức năng trang tìm kiếm thú cưng (Form Find)	37
5.6. Kiểm thử chức năng cài đặt người dùng (Form setting).....	38
5.7. Kiểm thử chức năng nhắn tin giữa 2 users	39
5.8. Kiểm thử chức năng nhắn tin với tất cả users.....	40
5.9. Kiểm thử chức năng quên mật khẩu	40
5.10. Kiểm thử về hiệu suất	42
CHƯƠNG 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN.....	43
6.1. Kết luận.....	43
6.2. Hướng phát triển	43

TÀI LIỆU THAM KHẢO

PHỤ LỤC

DANH MỤC HÌNH ẢNH

Hình 2.1 Mô hình kết nối TCP.....	3
Hình 2.2 Mô hình bất đồng bộ.....	5
Hình 3.1 Sơ đồ phân rã chức năng.....	6
Hình 3.2 Hình thể hiện trang Login	8
Hình 3.3 Giao diện trang chờ đăng nhập	9
Hình 3.4 Giao diện trang đăng ký	10
Hình 3.5 Giao diện trang Home	11
Hình 3.6 Giao diện trang đăng thông tin thú cưng	12
Hình 3.7 Giao diện trang thể hiện thông tin thú cưng được đăng lên.....	13
Hình 3.8 Giao diện trang thú cưng đi lạc.....	14
Hình 3.9 Giao diện trang cài đặt	15
Hình 3.10 Giao diện chat tổng	16
Hình 3.11 Giao diện chat giữa 2 user.....	16
Hình 3.12 Giao diện trang forgot pasword	17
Hình 3.13 Giao diện trang xác nhận mã OTP	17
Hình 5.1 Kiểm thử chức năng đăng ký	32
Hình 5.2 Kiểm thử chức năng đăng nhập	33
Hình 5.3 Kiểm thử chức năng trang đăng thông tin cho nhận nuôi.....	34
Hình 5.4 Kiểm thử chức năng trang đăng thông tin tìm pet bị mất	35
Hình 5.5 Kiểm thử chức năng trang nhận nuôi thú cưng.....	36
Hình 5.6 Kiểm thử chức năng trang tìm kiếm thú cưng	37
Hình 5.7 Kiểm thử chức năng cài đặt người dùng.....	38
Hình 5.8 Kiểm thử chức năng nhắn tin hai người	39
Hình 5.9 Kiểm thử chức năng nhắn tin với server (nhiều người).....	40
Hình 5.10 Kiểm thử chức năng gửi mã OTP	40
Hình 5.11 Mã OTP được gửi đến email.....	41
Hình 5.12 Nhập mã OTP	41
Hình 5.13 Đổi mật khẩu thành công	41
Hình 5.14 Đăng nhập với mật khẩu vừa đổi	42

LỜI CẢM ƠN

Trong quá trình làm đồ án này, chúng em gặp không ít những khó khăn và trở ngại do vốn kiến thức còn hạn chế. Dù bận rộn nhiều công việc nhưng cô vẫn dành nhiều thời gian và tâm huyết trong việc hướng dẫn chúng em. Cô luôn quan tâm, chỉ bảo và sửa chữa những vấn đề quan trọng giúp chúng em định hướng và làm việc theo quan điểm đúng đắn, chính sự tận tâm và nhiệt huyết của cô đã giúp cho chúng em có được tinh thần, một niềm tin và khối lượng kiến thức phong phú để ngày hôm nay, và cuối cùng đồ án của chúng em được hoàn thành.

Với tấm lòng biết ơn sâu sắc, chúng em xin chân thành gửi lời cảm ơn đến cô đã chân tình hướng dẫn – giúp đỡ chúng em trong suốt quá trình học tập tại trường. Cảm ơn cô đã giúp đỡ chúng em trong quá trình hoàn thành đồ án này.

Một lần nữa, chúng em xin chân thành cảm ơn cô !

NHÓM THỰC HIỆN

NỘI DUNG BÁO CÁO

CHƯƠNG 1. GIỚI THIỆU

1.1. Giới thiệu chung

Trên thế giới, việc nuôi và chăm sóc thú cưng ngày càng trở thành một nhu cầu quan trọng và phổ biến. Tuy nhiên, có nhiều trường hợp thú cưng bị đi lạc hoặc cần tìm một mái ấm mới. Để giải quyết vấn đề này, chúng em xây dựng một ứng dụng Pet Care Winform C#, nhằm giúp người dùng tìm kiếm và nhận nuôi thú cưng một cách thuận tiện và nhanh chóng.

1.2. Mục tiêu và chức năng chính

Mục tiêu chính của ứng dụng Pet Care là cung cấp một nền tảng kết nối giữa người muốn nhận nuôi thú cưng và những thú cưng cần được nuôi dưỡng mới.

Tính năng tìm kiếm thú cưng thất lạc: Ứng dụng cho phép người dùng tìm kiếm thông tin về các thú cưng bị đi lạc trong khu vực gần họ và liên lạc với chủ sở hữu để giúp đưa thú cưng trở về nhà.

Tính năng nhận nuôi thú cưng: Người dùng có thể xem danh sách các thú cưng đang tìm kiếm mái ấm mới và đăng ký nhận nuôi thông qua ứng dụng.

Đăng tin cho nhận nuôi thú cưng: Người dùng có thể tạo hồ sơ cá nhân cho thú cưng của mình, bao gồm tên, màu sắc, giống, giới tính và thông tin sức khỏe và đăng tin cho phép nhận nuôi

Cộng đồng Pet Care: Ứng dụng cung cấp một môi trường giao tiếp cho người dùng chia sẻ kinh nghiệm, câu chuyện về việc chăm sóc thú cưng. Người dùng có thể tương tác với nhau, tạo mối quan hệ và hỗ trợ lẫn nhau trong việc chăm sóc thú cưng thông qua việc chat với nhau

1.3. Công nghệ sử dụng

Ngôn ngữ lập trình: C#.

Giao diện người dùng: Winform.

Cơ sở dữ liệu: Firebase để lưu trữ thông tin về người dùng, thú cưng và các bài viết tìm kiếm, cho nhận nuôi thú cưng

Kết nối mạng: Ứng dụng sử dụng kết nối mạng để tải về thông tin về thú cưng thất lạc và cập nhật thông tin. Về phần chat thì tạm thời triển khai trên mạng LAN.

1.4. Đóng góp đê tài

Đê tài này đóng góp vào việc giảm thiểu số lượng thú cưng bị đi lạc và cung cấp một cách để những thú cưng cần mái ấm mới được tìm thấy nhanh chóng.

Người dùng có thể tìm thấy và nhận nuôi thú cưng theo ý muốn một cách thuận tiện và an toàn.

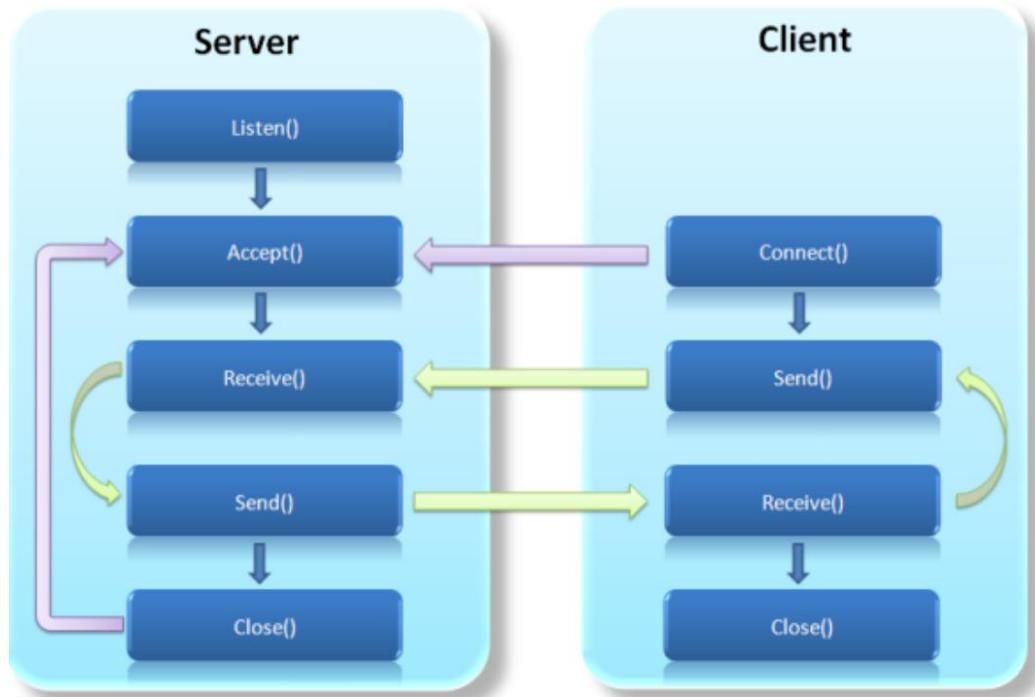
Môi trường giao tiếp trong cộng đồng Pet Care giúp người dùng chia sẻ kinh nghiệm và học hỏi từ nhau, tạo nên một cộng đồng yêu thú cưng đoàn kết và hỗ trợ nhau.

Qua việc phát triển ứng dụng Pet Care Winform C#, chúng em hy vọng mang đến một công cụ hữu ích và thiết thực cho cộng đồng yêu thú cưng.

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

2.1. Kết nối TCP để chat giữa server- client

Để đảm bảo độ tin cậy trong ứng dụng, nhóm đã triển khai tính năng chat tổng để cho phép nhiều client kết nối đến 1 server theo mô hình sau:



Hình 2.1 Mô hình kết nối TCP

2.2. Giao tiếp với Email Server

Sử dụng giao thức SMTP để gửi mail: SMTP (Simple Mail Transfer Protocol) là một giao thức được sử dụng để truyền tải và gửi thư điện tử qua mạng. Nó được thiết kế để chuyển giao thư từ một máy chủ thư đến máy chủ thư khác thông qua mạng Internet.

SMTP hoạt động dựa trên mô hình client-server, trong đó người gửi thư là client và máy chủ thư là server. Quá trình truyền tải thư điện tử thông qua SMTP diễn ra theo các bước sau:

Thiết lập kết nối: Máy khách (client) kết nối với máy chủ thư (server) trên cổng 25 (hoặc cổng khác nếu được cấu hình). Thông thường, kết nối sử dụng TCP/IP.

Xác thực: Sau khi kết nối được thiết lập, máy chủ thư yêu cầu máy khách xác thực danh tính. Điều này có thể được thực hiện bằng cách yêu cầu người dùng nhập tên người dùng và mật khẩu hoặc bằng cách sử dụng các phương pháp xác thực khác như TLS (Transport Layer Security).

Truyền tải thông điệp: Sau khi xác thực thành công, máy chủ thư cho phép máy khách gửi thông điệp. Thông điệp này bao gồm các thông tin như người gửi, người nhận, tiêu đề và nội dung.

Xác nhận: Sau khi máy khách gửi thông điệp thành công, máy chủ thư gửi một phản hồi trạng thái (status response) để xác nhận rằng thông điệp đã được nhận.

Đóng kết nối: Sau khi thông điệp đã được gửi và xác nhận, máy khách đóng kết nối với máy chủ thư.

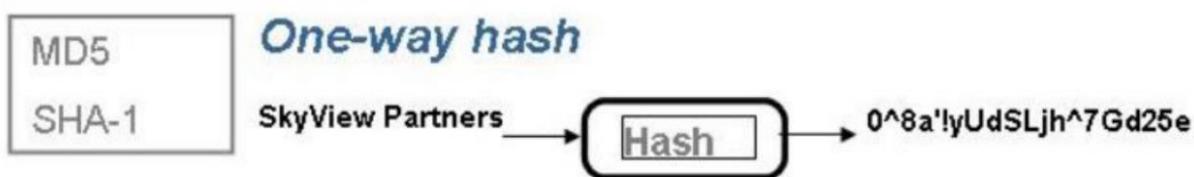
SMTP cũng hỗ trợ một số lệnh khác nhau để kiểm soát quá trình truyền tải thư, như lệnh HELO (Hello) để bắt đầu phiên làm việc, lệnh MAIL FROM để xác định người gửi, lệnh RCPT TO để xác định người nhận, và lệnh DATA để truyền tải nội dung thư.

SMTP cũng có thể được sử dụng kết hợp với các giao thức khác như POP (Post Office Protocol) hoặc IMAP (Internet Message Access Protocol) để cho phép người dùng truy cập và quản lý thư điện tử trên máy chủ thư từ xa.

2.3. Bảo vệ dữ liệu

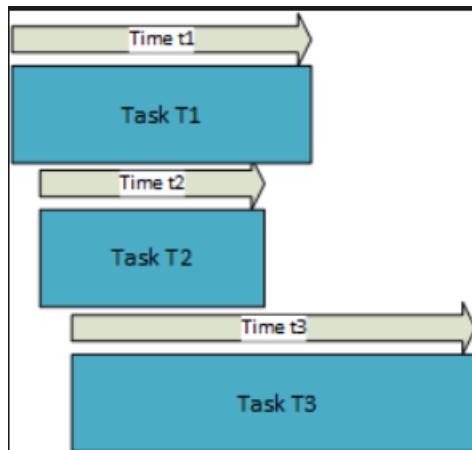
Hash functions:

- Băm từng chuỗi bit (password, etc...) tạo thành các chuỗi có độ dài nhất định.
- Không thể khôi phục dữ liệu ban đầu từ chuỗi đã băm.
- Chỉ cần sự thay đổi nhỏ trong giá trị ban đầu có thể thay đổi hoàn toàn kết quả băm.
- Sử dụng để xác minh mật khẩu.



2.4. Lập trình bất đồng bộ

Mô hình trong đó các nhiệm vụ không phải tuân thủ theo trình tự thời gian được gọi là mô hình bất đồng bộ (asynchronous). Như vậy mô hình bất đồng bộ cũng cho phép thực hiện song song nhiều nhiệm vụ cùng lúc.



Hình 2.2 Mô hình bất đồng bộ

Trong lập trình socket, mô hình bất đồng bộ được sử dụng rất phổ biến ở cả client và server. Đối với server, mô hình này cho phép xử lý đồng thời nhiều client và phát huy tốt khả năng xử lý song song của server. Đối với client, nó giúp chương trình không bị treo giao diện khi thực hiện các nhiệm vụ kéo dài.

2.5. Real-time Database Firebase

Firebase cung cấp một dịch vụ cơ sở dữ liệu thời gian thực, cho phép bạn lưu trữ và đồng bộ dữ liệu giữa các thiết bị và ứng dụng một cách nhanh chóng và dễ dàng. Dịch vụ cơ sở dữ liệu thời gian thực của Firebase được gọi là Firebase Realtime Database.

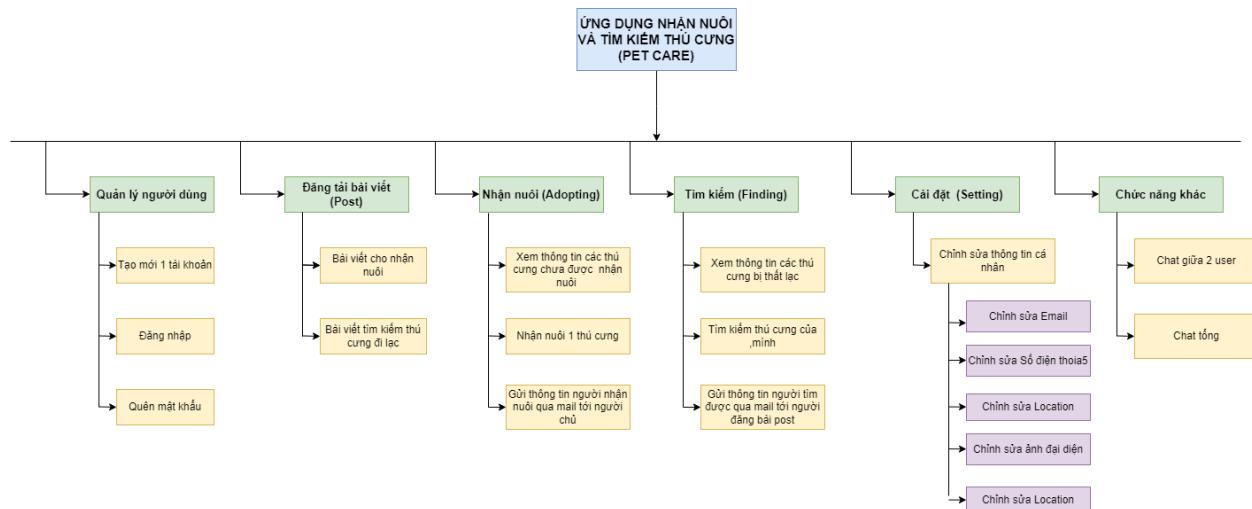
Firebase Realtime Database là một cơ sở dữ liệu NoSQL dạng cây, nơi dữ liệu được lưu trữ dưới dạng JSON. Đặc điểm nổi bật của nó là khả năng tự động đồng bộ dữ liệu giữa các thiết bị và ứng dụng một cách trực tiếp và thời gian thực. Khi dữ liệu được thay đổi trên một thiết bị, nó sẽ tự động được cập nhật trên tất cả các thiết bị khác mà có kết nối đến cùng cơ sở dữ liệu Firebase Realtime Database.

Dịch vụ cơ sở dữ liệu thời gian thực của Firebase cung cấp các tính năng mạnh mẽ như:

- Đồng bộ dữ liệu thời gian thực: Dữ liệu được tự động đồng bộ giữa các thiết bị và ứng dụng trong thời gian thực, đảm bảo rằng mọi người dùng đều nhìn thấy các thay đổi mới nhất.
- Tích hợp mạnh mẽ: Firebase Realtime Database tích hợp tốt với các sản phẩm và dịch vụ khác của Firebase, cho phép bạn xây dựng các ứng dụng hoàn chỉnh với các tính năng như xác thực người dùng, thông báo đẩy và phân tích.
- Quyền truy cập và bảo mật: Người dùng có thể kiểm soát quyền truy cập vào dữ liệu của mình thông qua quy tắc bảo mật linh hoạt, đảm bảo rằng chỉ những người được cho phép mới có thể truy cập và sửa đổi dữ liệu.
- Mở rộng linh hoạt: Firebase Realtime Database có khả năng mở rộng dễ dàng theo nhu cầu của người dùng. Người dùng có thể xử lý hàng triệu người dùng và dữ liệu lớn mà không gặp vấn đề về hiệu suất.

CHƯƠNG 3. PHÂN TÍCH THIẾT KẾ HỆ THỐNG

3.1. Sơ đồ phân rã chức năng



Hình 3.1 Sơ đồ phân rã chức năng

3.2. Thiết kế Database

USER (UserName, Password, Email, Phone, Location, Avatar).

PET(PetID, PetName, PetColor, SubType, Sex, Gender, isAdopted, isFinded, PetDateofBirth, HealthInfo).

POST (PostID, TypePost, DatePost, UserPost, Phone, Email).

PETHEALTH (HealthID, FavoriteFood, NameClient, Height, Weight, Vaccinations, PreviousOwner).

Tân từ:

USER:

- UserName: tên đăng nhập của người dùng.
- Password: mật khẩu của người dùng.
- Email: địa chỉ email của người dùng.
- Phone: số điện thoại của người dùng.
- Location: địa chỉ của người dùng.
- Avatar: hình đại diện của người dùng.

PET:

- PetID: định danh duy nhất cho mỗi con vật.
- PetName: tên của con vật.
- PetColor: màu sắc của con vật.
- SubType: loại con vật.
- Sex: giới tính của con vật (đực hoặc cái).
- Gender: giới tính của con vật (nam hoặc nữ).
- IsAdopted: xác định liệu con vật đã được nhận nuôi hay chưa.
- IsFound: xác định liệu con vật đã được tìm thấy hay chưa.
- DateOfBirth: ngày sinh của con vật.
- HealthInfo: thông tin sức khỏe của con vật.

POST:

- PostID: định danh duy nhất cho mỗi bài đăng.
- PostType: loại bài đăng (ví dụ: tìm kiếm, cho nhận nuôi).
- PostDate: ngày đăng bài.
- NameClient: Tên người dùng đăng bài.

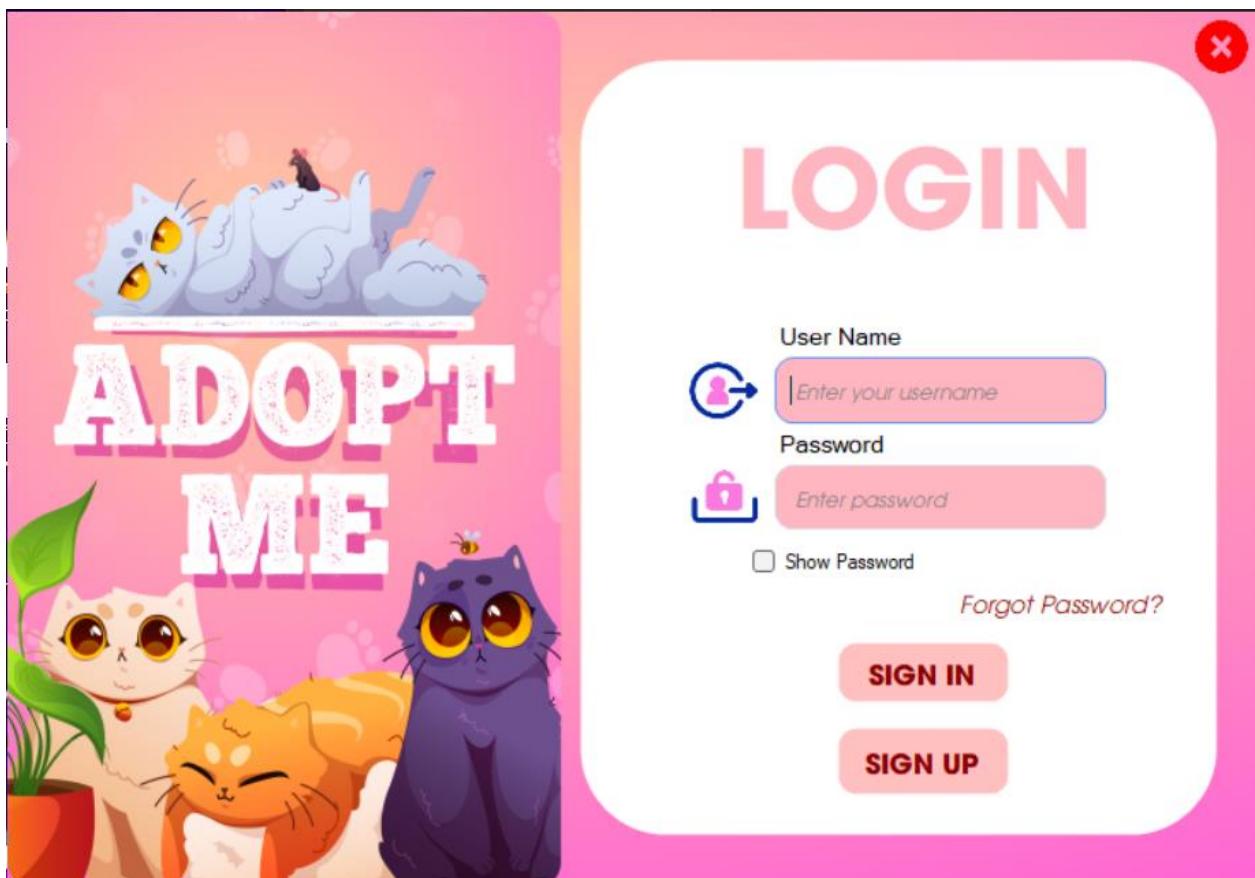
- Phone: số điện thoại liên lạc trong bài đăng.
- Email: địa chỉ email liên lạc trong bài đăng.

PETHEALTH:

- HealthID: định danh duy nhất cho mỗi thông tin sức khỏe.
- FavoriteFood: thức ăn yêu thích của con vật.
- PetID: định danh của con vật được liên kết với thông tin sức khỏe.
- Height: chiều cao của con vật.
- Weight: cân nặng của con vật.
- Vaccinations: thông tin về các loại vắc-xin của con vật.
- PreviousOwner: thông tin về chủ sở hữu trước đó của con vật.

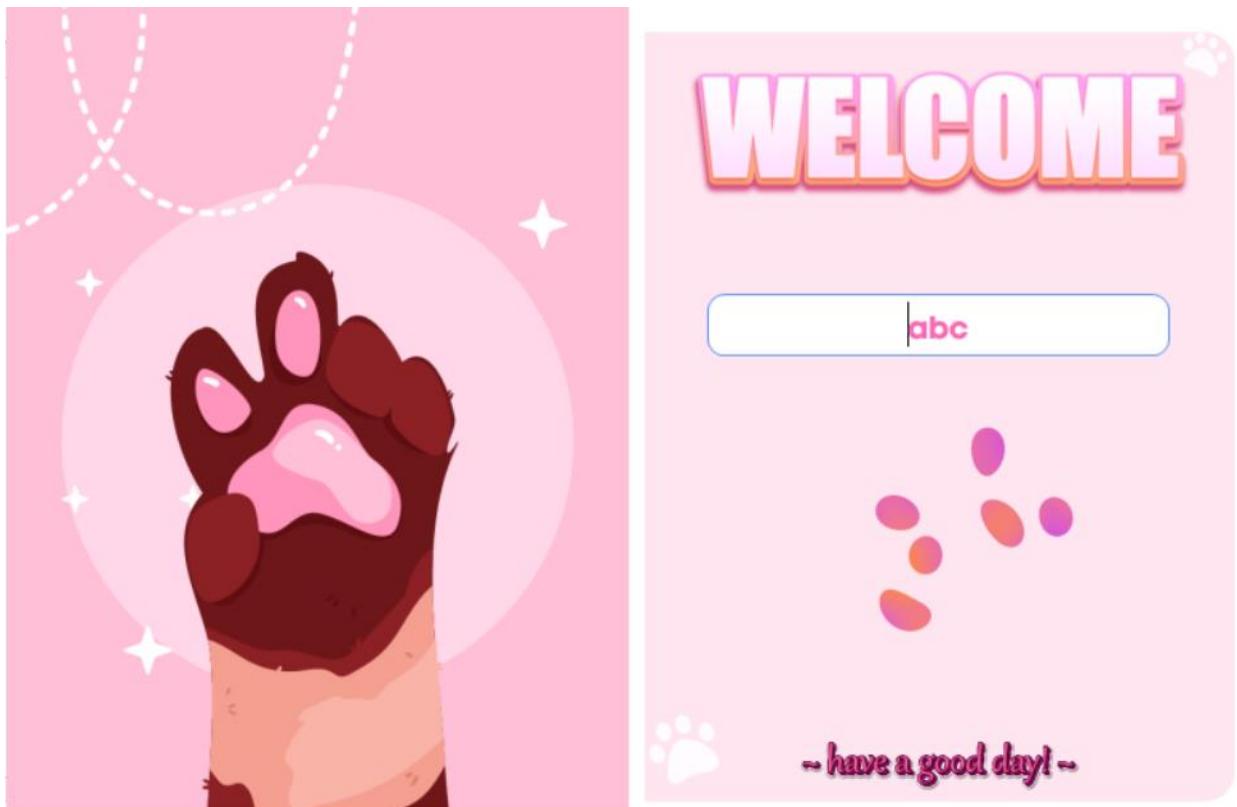
3.3. Giao diện người dùng

3.3.1. Trang Login



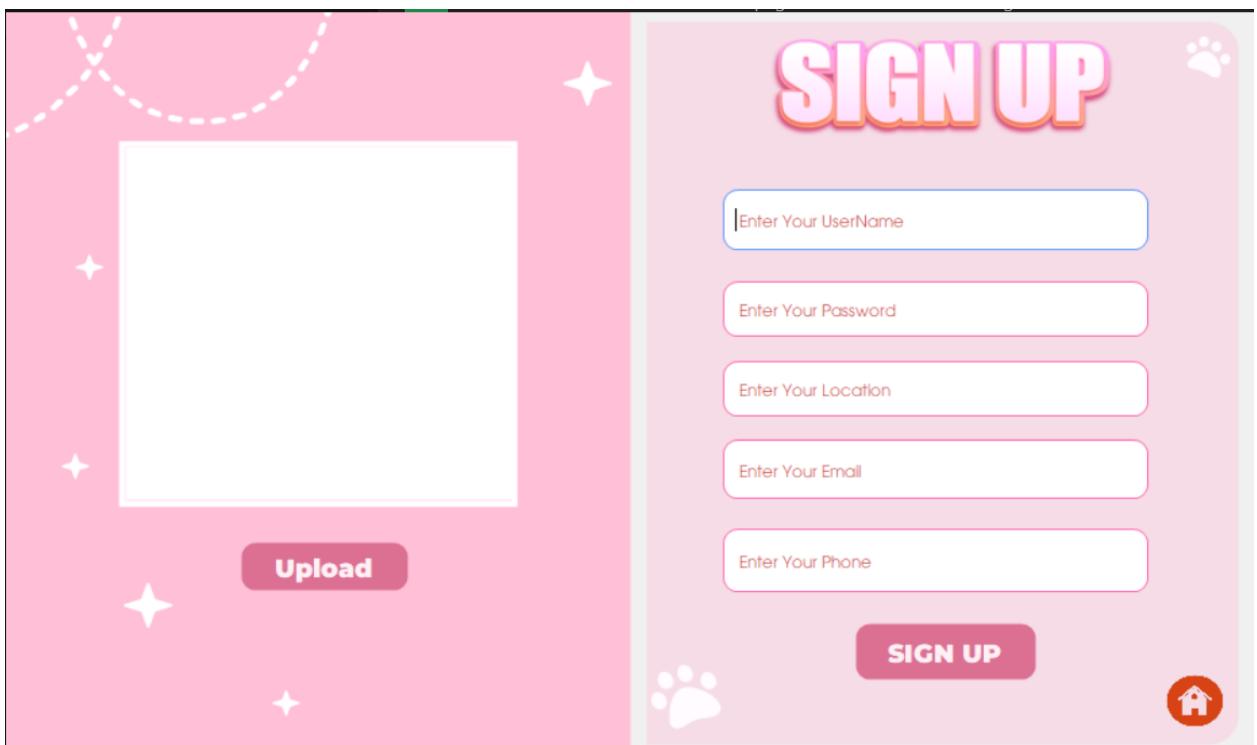
Hình 3.2 Hình ảnh trang Login

3.3.2. Trang Loading



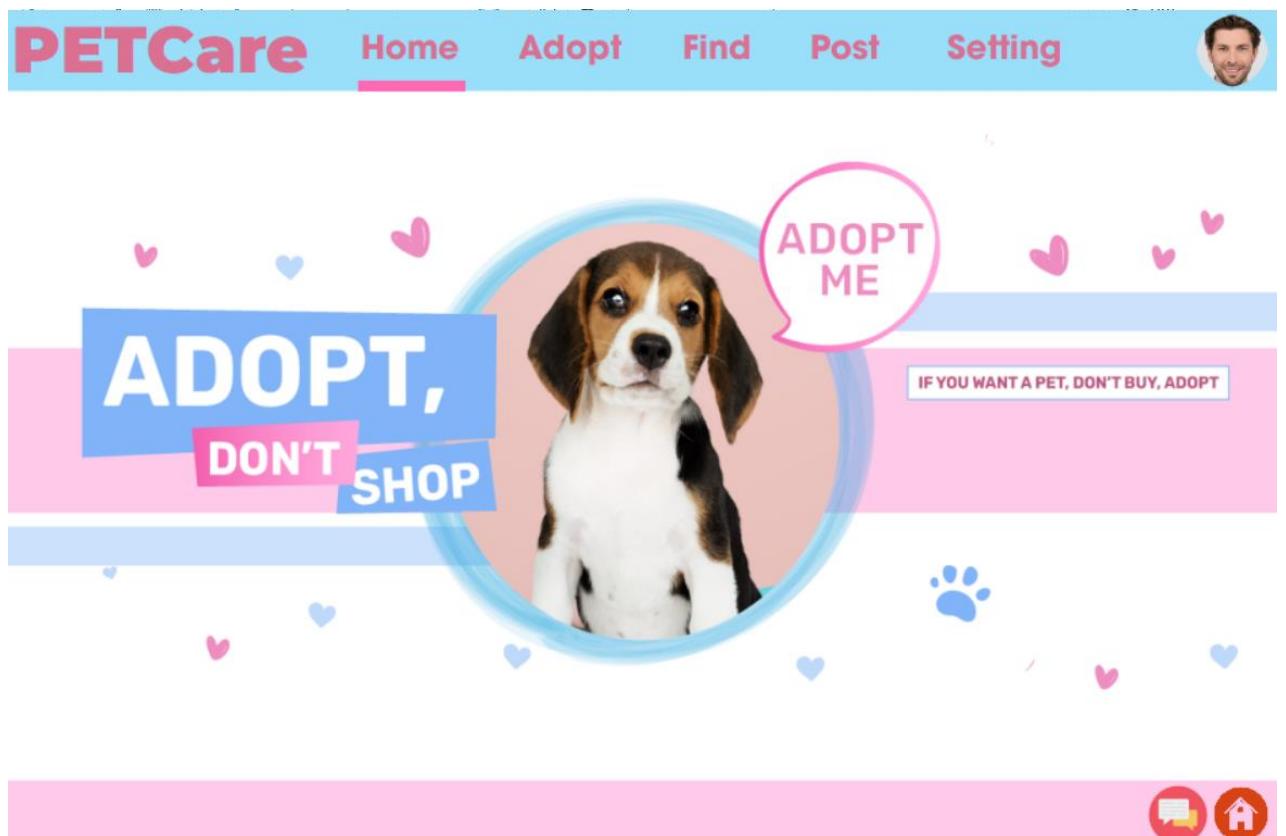
Hình 3.3 Giao diện trang chờ đăng nhập

3.3.3. Trang đăng ký



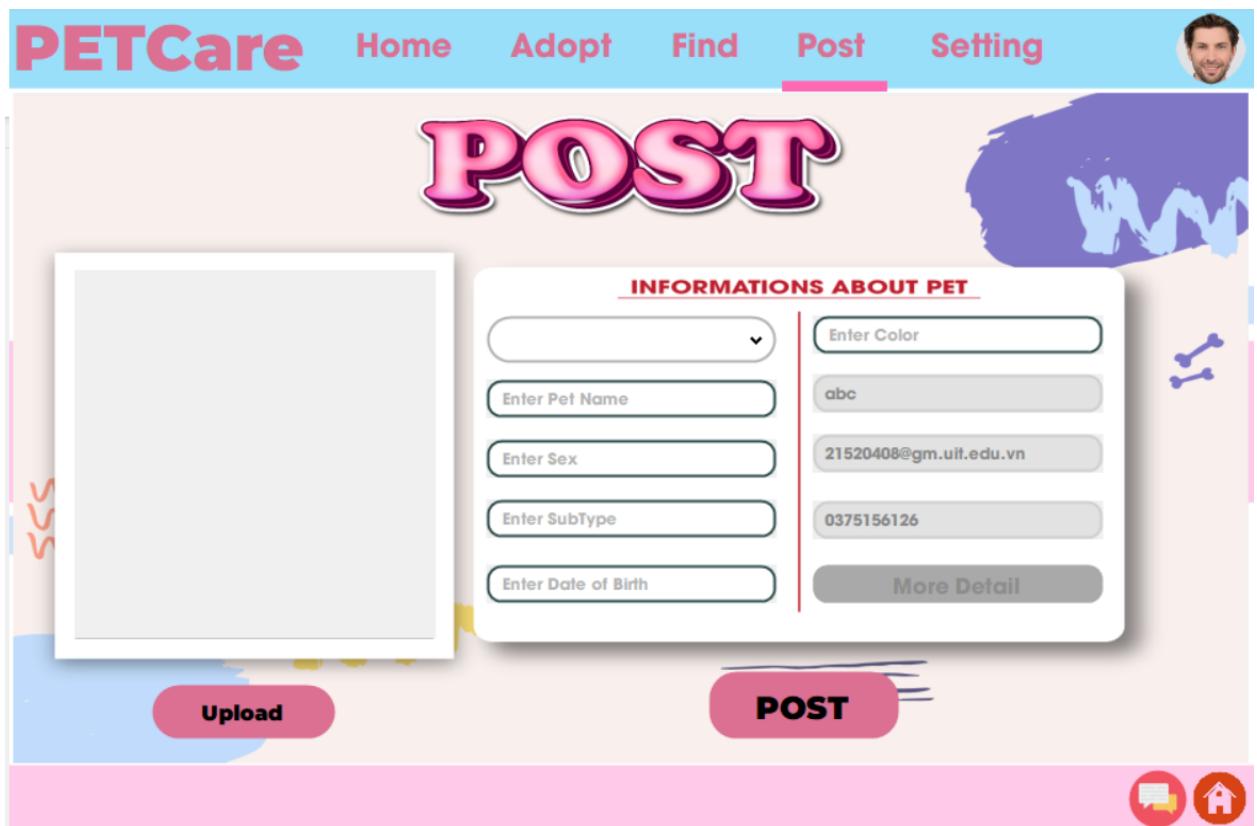
Hình 3.4 Giao diện trang đăng ký

3.3.4. Trang Home



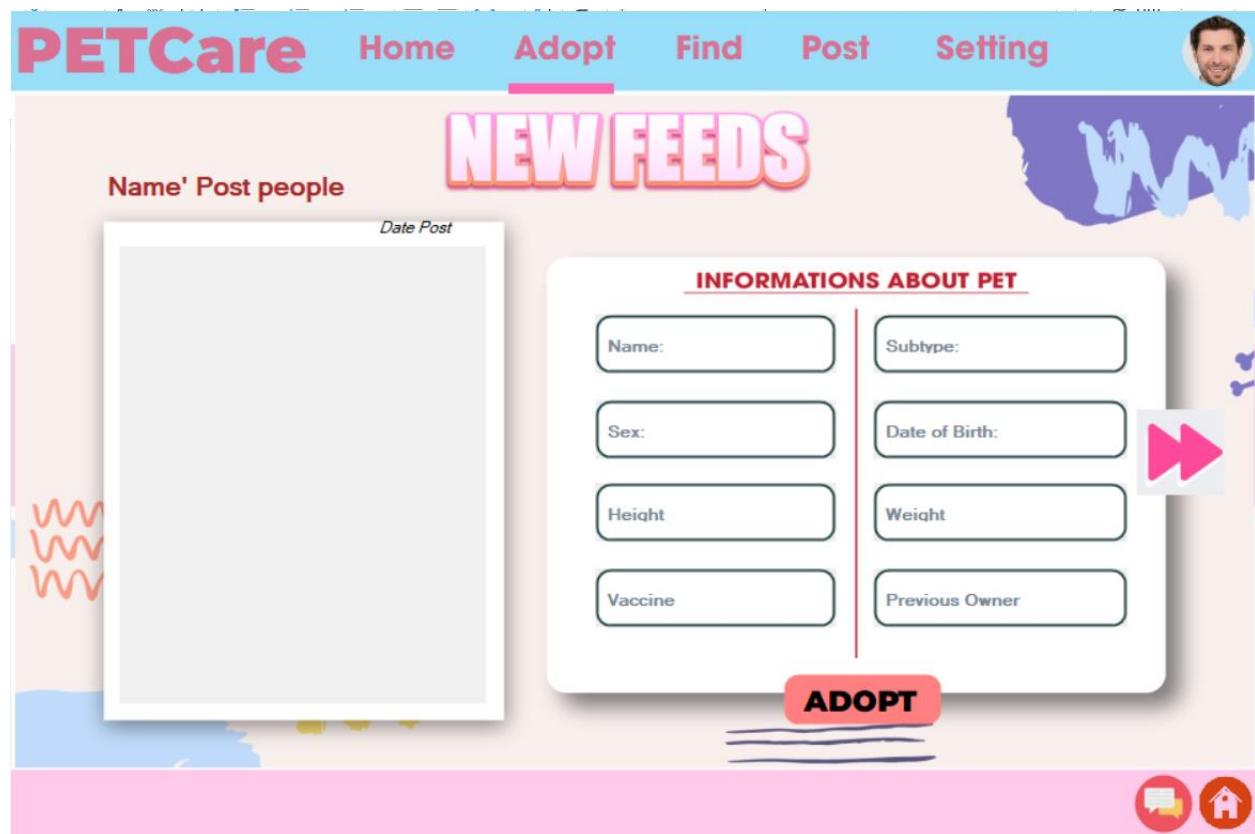
Hình 3.5 Giao diện trang Home

3.3.5. Trang post thông tin



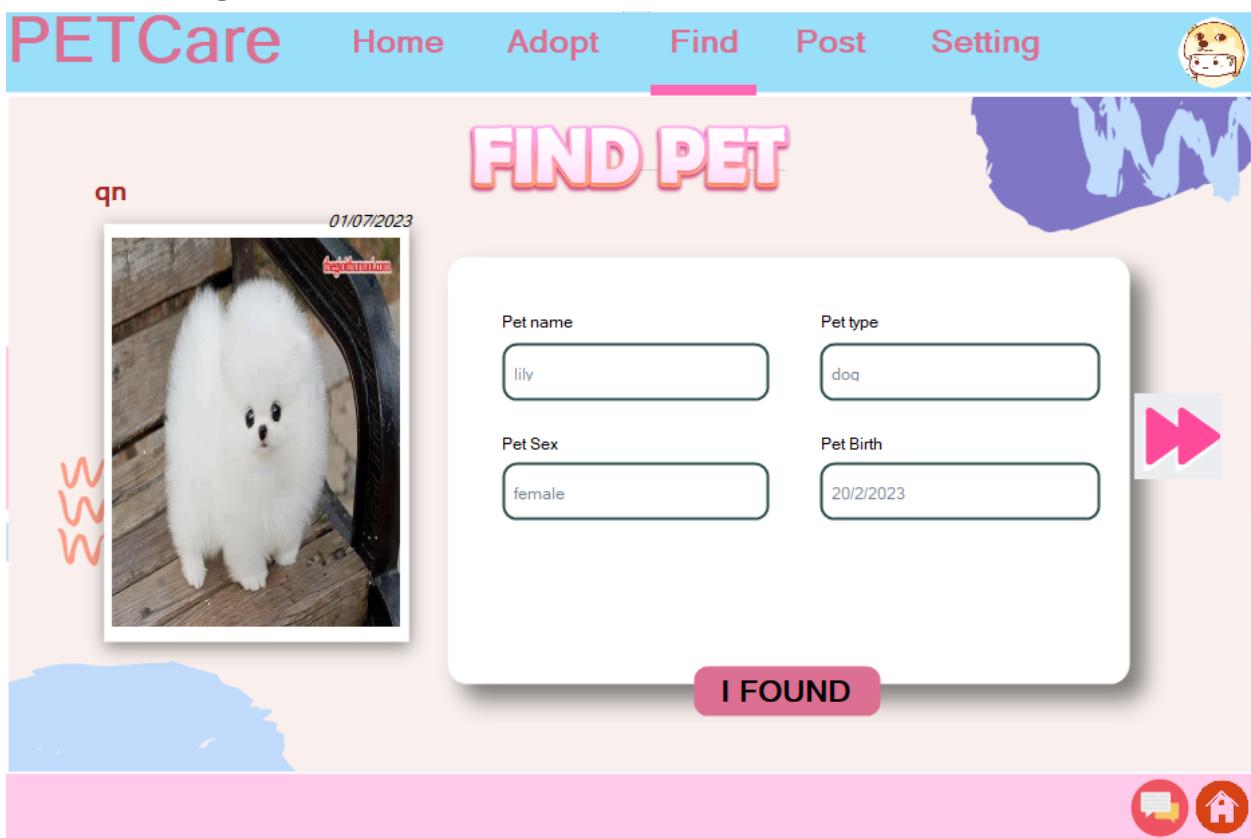
Hình 3.6 Giao diện trang đăng thông tin thú cưng

3.3.6. Trang nhận nuôi



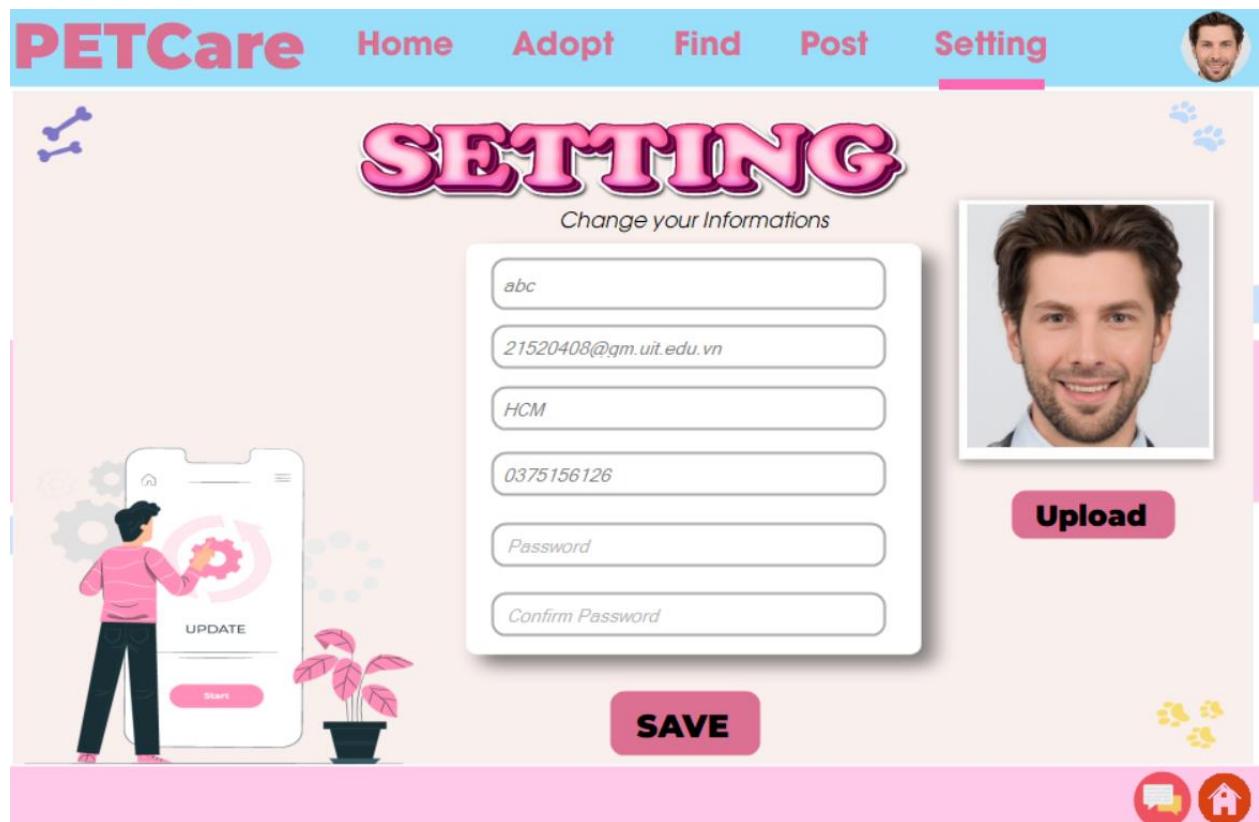
Hình 3.7 Giao diện trang thể hiện thông tin thú cưng được đăng lên

3.3.7. Trang tìm kiếm



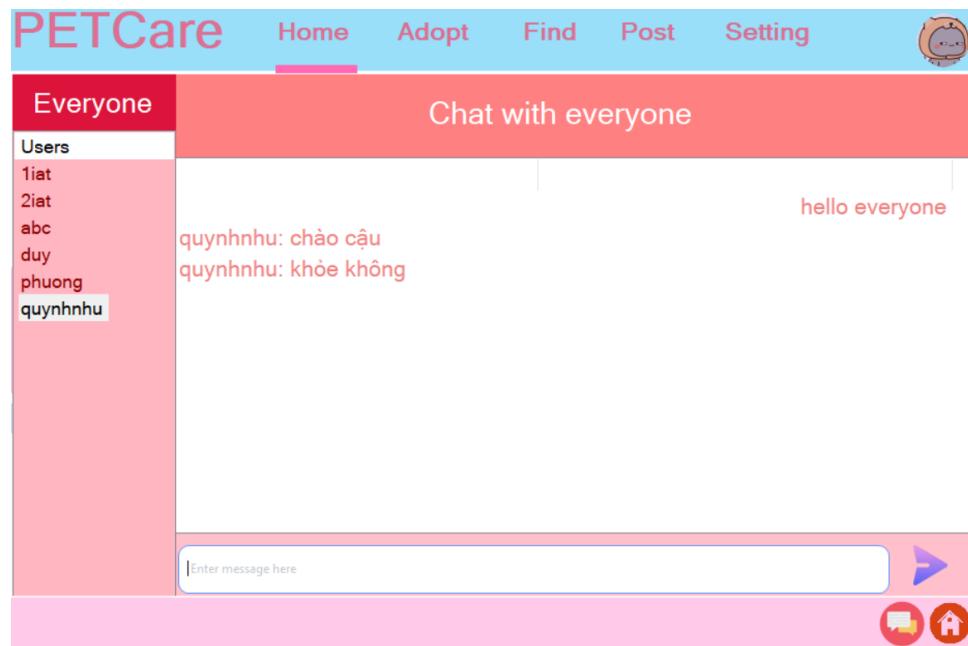
Hình 3.8 Giao diện trang thú cưng đi lạc

3.3.8. Trang cài đặt



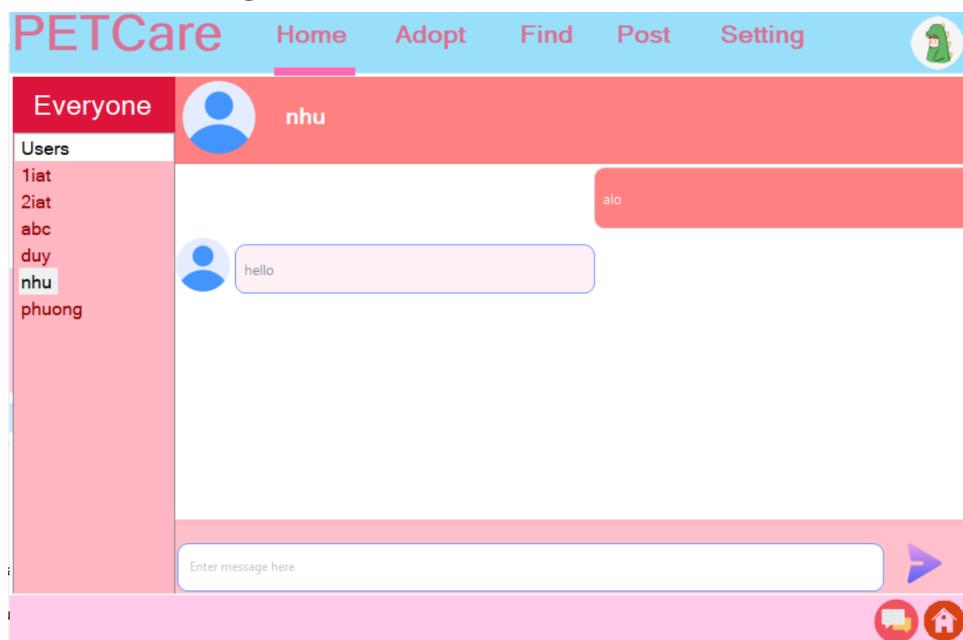
Hình 3.9 Giao diện trang cài đặt

3.3.9. Giao diện chat tổng



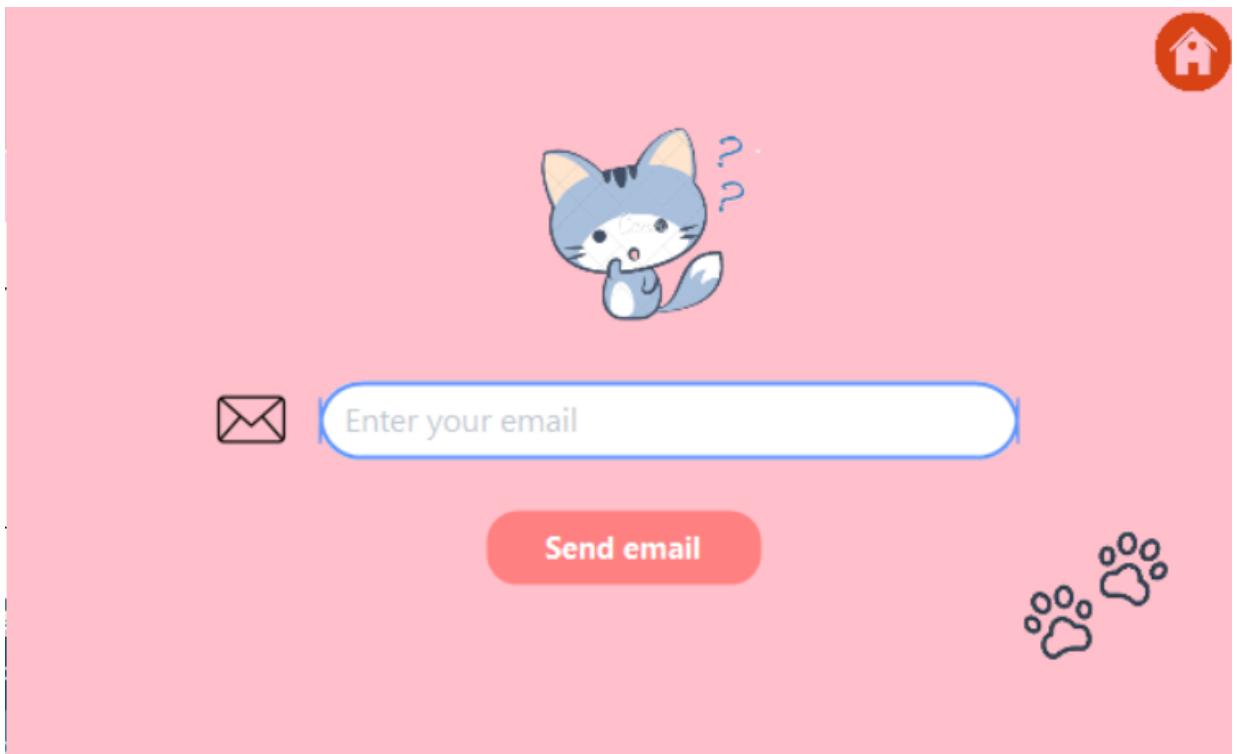
Hình 3.10 Giao diện chat tổng

3.3.10. Giao diện chat giữa 2 users



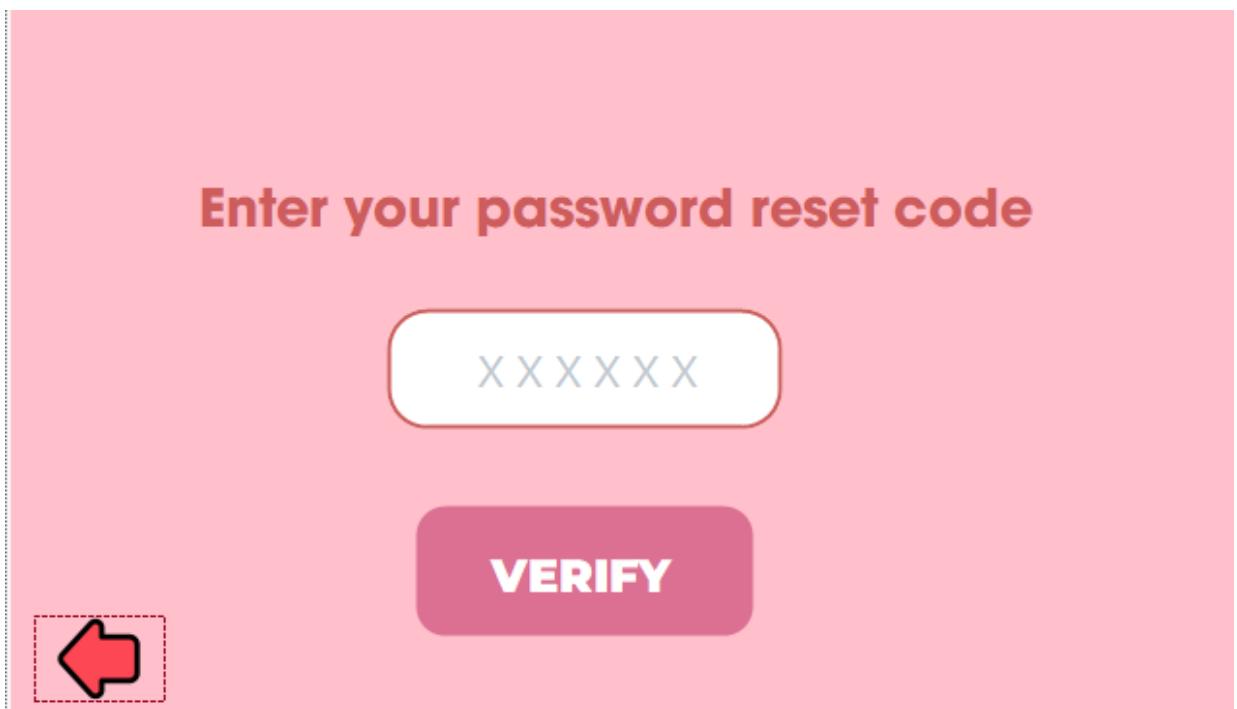
Hình 3.11 Giao diện chat giữa 2 user

3.3.11. Trang forgot password



Hình 3.12 Giao diện trang forgot pasword

3.3.12. Trang xác nhận mã otp khi yêu cầu tạo lại mật khẩu



Hình 3.13 Giao diện trang xác nhận mã OTP

3.4. Timeline

3.4.1. Giai đoạn 1: Phân tích đề tài

Giai đoạn	Thời gian Bắt đầu	Thời gian Kết thúc	Công việc	Mô tả công việc	Thực hiện bởi	Trạng thái	Khó khăn gặp phải	Ghi chú
Giai đoạn 1: PHÂN TÍCH ĐỀ TÀI	7/3/2023	13/3/2023	Chọn đề tài	Nhóm cần thực hiện Chọn đề tài, mô tả các chức năng cần có của Đề tài	Cả nhóm	Hoàn thành		
	7/3/2023	13/3/2023	Mô tả chức năng	Liệt kê các chức năng của đồ án. Nhận xét các chức năng xem phù hợp với thực tế hay không	Cả nhóm	Hoàn thành		

3.4.2. Giai đoạn 2: Thiết kế giao diện + Code

Giai đoạn 2: THIẾT KẾ GIAO DIỄN + CODE + VIẾT REPORT	13/3/2023	27/3/2023	Thiết kế UI	Thiết kế giao diện đăng nhập, giao diện trang chủ, giao diện đăng ký, giao diện chờ đăng nhập	Hoài Phương + Đức Tài	Hoàn thành	Lỗi không thể chép giá trị từ form này sang form khác. Lỗi Control Panel khi chạy bị lêch so với lúc đầu	
	9/5/2023	16/5/2023	Demo Database	Thiết kế bản Demo Database ứng với các chức năng mô tả	Quỳnh Như	Hoàn thành		
	9/5/2023	16/5/2023	Tìm tài liệu hướng dẫn	Tìm tài liệu hướng dẫn kết nối Database Firebase	Quỳnh Như	Hoàn thành		Link tham khảo
	9/5/2023	16/5/2023	Kết nối Database	Code thực hiện kết nối database firebase cho form Adopt(nhận nuôi)	Hoài Phương	Hoàn thành		
	18/4/2023	9/5/2023	Tìm hiểu Đăng kí, đăng nhập trên Firebase	Tìm hiểu cơ chế đăng kí, đăng nhập trong Firebase	Hoài Phương	Hoàn thành	Fix lỗi không kết nối được firebase.	Đã fix được lỗi
				Hoàn thiện thiết kế Database	Hoài Phương	Hoàn		

3.4.3. Giai đoạn 3: Xử lý Code logic BackEnd

Giai đoạn 3: XỬ LÝ CODE LOGIC BE	23/5/2023	31/5/2023	Thử nghiệm lấy dữ liệu về form	Test lấy dữ liệu từ firebase về form	Minh Duy	Hoàn thành		
	23/5/2023	31/5/2023	Thực hiện chức năng bấm mật khẩu	Code bấm mật khẩu đăng nhập để lưu lên firebase	Minh Duy	Hoàn thành		
	31/5/2023	6/6/2023	Test thử chức năng đăng kí, đăng nhập	Thử tạo 1 tài khoản và đăng nhập sử dụng ứng dụng	Hoài Phương	Hoàn thành		
	6/6/2023	13/6/2023	Code chức năng quên mật khẩu	Code chức năng quên mật khẩu	Quỳnh Như	Hoàn thành		
	6/6/2023	13/6/2023	Code form Find	Code chức năng tìm kiếm thử cung cấp thử lục. Đăng tìm thông tin thử cung bị thất lạc	Minh Duy	Hoàn thành		
	6/6/2023	13/6/2023	Code Form Adopt	Code chức năng hiển thị tất cả các thủ cung được Post sau đó nhận nuôi nếu yêu thích	Quỳnh Như	Hoàn thành		
	6/6/2023	13/6/2023		Kiểm tra lại một lần nữa UI hiệu chỉnh (nếu có), tính chính xác	Đức Tài	Hoàn thành		
	6/6/2023	13/6/2023	Hoàn thiện UI	Kiểm tra lại một lần nữa UI hiệu chỉnh (nếu có), tính chính xác button text box, back ground phù hợp. Chính lại cái form Home, form Waiting Access, Form Info of Pet	Đức Tài	Hoàn thành		
	6/6/2023	17/6/2023	Code thêm tính năng chat	Cho phép chat trao đổi thông tin của 2 người dùng và chat với tất cả mọi người	Quỳnh Như	Hoàn thành	Không thể chat online	Tính năng phát sinh
	6/6/2023	13/6/2023	Code Form Post	Đăng thông tin thú cưng bao gồm các thông tin về thú cưng, hình ảnh và đăng lên	Hoài Phương	Hoàn thành		
	6/6/2023	13/6/2023	Code Form Settings	Chỉnh sửa thông tin người dùng bao gồm email, phone, location, password, avatar	Quỳnh Như	Hoàn thành		
	17/6/2023	20/6/2023	Publish ứng dụng	Tiến hành publish ứng dụng cho	Đức Tài	Hoàn		

3.4.4. Giai đoạn 4: Viết báo cáo

Giai đoạn 4: KIỂM THÚ & VIẾT BÁO CÁO	20/6/2023	24/6/2023	Kiểm tra chức năng nhận nuôi thú cưng		Đức Tài	Hoàn thành	Do dữ liệu được lưu trên firebase, nonSQL nên không thể thực hiện truy vấn, vì vậy phải duyệt các đối tượng qua id nên hiệu suất không cao	
	20/6/2023	24/6/2023	Kiểm tra chức năng Tìm kiếm thú cưng		Minh Duy	Hoàn thành		
	27/6/2023	30/6/2023	Đổi UI các form	Thiết kế lại UI form loading, form thể hiện sức khỏe thú cưng, form find, form sign up	Đức Tài	Hoàn thành		
	20/6/2023	Đến ngày Báo cáo	Kiểm tra các chức năng để xuất ban đầu	Kiểm tra lại với các chức năng liệt kê ban đầu thì đã hoạt động ổn định hay chưa, phản hồi chưa thực hiện được	Quỳnh Như	Hoàn thành		
THỰC HIỆN & VIẾT BÁO CÁO	20/6/2023	Đến ngày Báo cáo	Kiểm tra các chức năng để xuất ban đầu	Kiểm tra lại với các chức năng liệt kê ban đầu thì đã hoạt động ổn định hay chưa, phản hồi chưa thực hiện được	Quỳnh Như	Hoàn thành		
	20/6/2023	Đến ngày Báo cáo	Đánh giá hiệu suất hoạt động của ứng dụng trên mạng Internet	Tính toán hiệu suất hoạt động của ứng dụng, bao nhiêu người dùng có thể sử dụng, có xảy ra tình trạng giật lag, quá tải server, những vấn đề cần khắc phục (nếu có)	Hoài Phương + Minh Duy	Hoàn thành		
	30/6/2023	30/06/2023	Soạn side, chỉnh format báo cáo		Đức Tài	Hoàn thành		
	20/6/2023	Đến ngày Báo cáo	Viết báo cáo	Hoàn thành tất cả các mục của báo cáo, hoàn chỉnh báo cáo cho phù hợp. Báo cáo đó án trước giảng viên	Cả nhóm	Hoàn thành		

CHƯƠNG 4. HIỆN THỰC ĐỀ TÀI:

Hình ảnh code minh họa các form. Code chi tiết được đính kèm ở file báo cáo

```
private async void guna2Button1_Click(object sender, EventArgs e)
{
    // Login

    if (string.IsNullOrEmpty(guna2TextBox1.Text) || string.IsNullOrEmpty(guna2TextBox2.Text))
    {
        // Check if textbox is Empty
        MessageBox.Show("Please Put Username or Password.", "Notification", MessageBoxButtons.OK, MessageBoxIcon.Information)
    }
    else
    {
        try
        {
            await Task.Run(() =>
            {
                FirebaseResponse response = client.Get("Users/");
                Dictionary<string, User> result = response.ResultAs<Dictionary<string, User>>();

                bool login = false;
                bool check = true; // check islogin
                foreach (var get in result)
                {
                    string user_result = get.Value.username;
                    string pass_result = get.Value.password;
                    avatarimg = get.Value.avatar;
                    email = get.Value.email;
                    phone = get.Value.phone;
                    string hashPass = ComputeSha256Hash(guna2TextBox2.Text);
                    if (guna2TextBox1.Text == user_result && hashPass == pass_result )
                    {
                        if (get.Value.islogin != true)
                        {
                            login = true;
                            check = true;
                            get.Value.islogin = true;
                        }
                    }
                }
            });
        }
    }
}
```

```

        get.Value.islogin = true;
        var updateResponse = client.Update("Users/" + get.Key, get.Value);
        break;
    }
    else
    {
        MessageBox.Show("You have already login!", "Notification", MessageBoxButtons.OK, MessageBoxIcon.Information);
        check = false;
        break;
    }
}

Invoke(new Action(() =>
{
    if (login)
    {
        MessageBox.Show("Welcome " + guna2TextBox1.Text, "Login sucessfully", MessageBoxButtons.OK, MessageBoxIcon.Information);
        ShareVariable.Username = guna2TextBox1.Text;
        ShareVariable.Phone = phone;
        ShareVariable.Email = email;
        this.Hide();
        Form2 f2 = new Form2();
        f2.guna2TextBox1.Text = this.guna2TextBox1.Text;
        f2.ShowDialog();
    }
    else if (check)
    {
        MessageBox.Show("Incorrect username or password.", "Notification", MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
}));
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message);
}

```

Hình: Code minh họa Form login

Ở chức năng login, sẽ thực hiện lấy nội dung từ Guna2textbox1 (User) và Guna2textbox2 (password). Password sẽ được băm với hàm băm Sha256 sau đó so sánh với giá trị lưu trên firebase. Nếu thành công kết quả sẽ trả về messagebox (Welcome + “Username”), ngược lại trả về thông báo lỗi như “Incorrect username or password”, “You have already login!”, ...

```

private void guna2Button1_Click(object sender, EventArgs e)
{
    // Register

    if (string.IsNullOrEmpty(guna2TextBox1.Text) || string.IsNullOrEmpty(guna2TextBox2.Text)
        || string.IsNullOrEmpty(guna2TextBox3.Text) || string.IsNullOrEmpty(guna2TextBox4.Text)
        || Avatar.Image == null)
    {
        //Check all textbox if some are Empty.
        MessageBox.Show("Please specify all data needed.", "Notification", MessageBoxButtons.OK, MessageBoxIcon.Information)
    }
    else
    {
        string hashPass = ComputeSha256Hash(guna2TextBox2.Text);

        //The User Class
        var user = new User
        {

            username = guna2TextBox1.Text,
            password = hashPass,
            email = guna2TextBox3.Text,
            phone = guna2TextBox4.Text,
            location = guna2TextBox5.Text,
            avatar = ImageToBase64String(Avatar),
            islogin = false,
        };
        if(!id_Leave())
        {

            //but if you want to have like a unique key use Push not Set.
            FirebaseResponse response = client.Set("Users/" + guna2TextBox1.Text, user);

            MessageBox.Show("Register Account Successfully");
            this.Close();
            fLogin f1 = new fLogin();
            f1.Show();
        }
    }
}

1 reference
private bool id_Leave()
{
    //Identifying if the name is existed or not.
    // From looping it using foreach.

    FirebaseResponse response = client.Get("Users/");
    Dictionary<string, User> getSameId = response.ResultAs<Dictionary<string, User>>();
    foreach (var sameID in getSameId)
    {
        string getsame = sameID.Value.username;
        if (guna2TextBox1.Text == getsame)
        {
            MessageBox.Show("Name is Already Taken.");
            guna2TextBox1.Text = string.Empty;
            return true;
        }
    }
    return false;
}

```

Hình: Code minh họa Form Register

Ở chức năng đăng kí, cho phép điền các thông tin như Username, Password, email, số điện thoại, nơi sinh sống. Sau khi thực hiện điền thông tin, dữ liệu sẽ được lưu trữ lên firebase. Sau đó người dùng có thể đăng nhập dựa vào thông tin đã đăng kí.

```
private void guna2Button1_Click(object sender, EventArgs e)
{
    if (string.IsNullOrEmpty(guna2TextBox1.Text)) MessageBox.Show("Please input your Email!");
    else
    {
        bool verifyemail = false;
        FirebaseResponse response = client.Get("Users/");
        Dictionary<string, User> result = response.ResultAs<Dictionary<string, User>>();
        foreach (var get in result)
        {
            string usr_email = get.Value.email;
            if (usr_email == guna2TextBox1.Text)
            {
                verifyemail = true;
                ShareVariable.Email = usr_email;
                break;
            }
        }
        if (verifyemail == false) MessageBox.Show("Can not find email!");
        else
        {
            string from, pass, messbody;
            Random random = new Random();
            randomCode = (random.Next(999999)).ToString();
            if (randomCode.Length != 6)
            {
                randomCode = new string('0', 6 - randomCode.Length) + randomCode;
            }
            ShareVariable.OTP = randomCode;
            MailMessage message = new MailMessage();
            to = guna2TextBox1.Text.ToString();
            from = "adoptionpet818@gmail.com";
            pass = "nqrhurlflgaphvqr";
            messbody = $"Your Password Reset Code is {randomCode}";
            message.To.Add(to);
            message.From = new MailAddress(from);
            message.Body = messbody;
            message.Subject = "Password Reset Code";
            SmtpClient smtp = new SmtpClient("smtp.gmail.com");
            smtp.EnableSsl = true;
        }
    }
}
```

```
        smtp.Port = 587;
        smtp.DeliveryMethod = SmtpDeliveryMethod.Network;
        smtp.Credentials = new NetworkCredential(from, pass);
        try
        {
            smtp.Send(message);
            MessageBox.Show("Code Successfully Sent");
            this.Hide();
            Form_Verify form_Verify = new Form_Verify();
            form_Verify.ShowDialog();
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }
}
```

Hình: Code minh họa Form Forgot Password

Chức năng quên mật khẩu, người dùng sẽ nhập email đã đăng ký tài khoản và hệ thống sẽ gửi một mã OTP gồm 6 số tới email. Người dùng sử dụng mã OTP để xác minh và đặt lại mật khẩu. Mật khẩu vừa nhập sẽ được băm và đưa lên lưu trữ trên Firebase.

```

private async void PostButton_Click(object sender, EventArgs e)
{
    FirebaseResponse resp = await client.GetTaskAsync("Couter/Node");
    CounterClass get = resp.ResultAs<CounterClass>();

    DateTime currentDate = DateTime.Now;
    string dateString = currentDate.ToString("dd/MM/yyyy");

    if (string.IsNullOrEmpty(PetNameTextBox.Text) || string.IsNullOrEmpty(PetSubtypeTextBox.Text) ||
        string.IsNullOrEmpty(PetColorTextBox.Text) || string.IsNullOrEmpty(PetDofTextBox.Text) ||
        string.IsNullOrEmpty(PetSexTextBox.Text) || string.IsNullOrEmpty(NameClientTextBox.Text) ||
        string.IsNullOrEmpty(EmailTextBox.Text) || string.IsNullOrEmpty(PhoneTextBox.Text) ||
        picturepet.Image == null)
    {
        // Check if textbox is Empty
        MessageBox.Show("Please Put Full Information", "Notification", MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
    else
    {
        try
        {
            PostData pd = new PostData()
            {
                PetName = PetNameTextBox.Text,
                PetSubtype = PetSubtypeTextBox.Text,
                PetColor = PetColorTextBox.Text,
                PetDob = PetDofTextBox.Text,
                PetSex = PetSexTextBox.Text,
                NameClient = NameClientTextBox.Text,
                Email = EmailTextBox.Text,
                Phone = PhoneTextBox.Text,
                DatePost = dateString,
                isAdopted = isadopted,
                isFinded = isfinded,
                Health_Pet = healthPet,
                id = Convert.ToInt32(get.id) + 1,
                imgstr = ImageIntoBase64String(picturepet)
            };
            int Id = Convert.ToInt32(get.id) + 1;

            var set = client.Set("Post/" + Id, pd);
            MessageBox.Show("Post Successfully!", "Notification", MessageBoxButtons.OK, MessageBoxIcon.Information);

            //Clear all info
            PetNameTextBox.Clear();
            PetSubtypeTextBox.Clear();
            PetColorTextBox.Clear();
            PetDofTextBox.Clear();
            NameClientTextBox.Clear();
            EmailTextBox.Clear();
            PhoneTextBox.Clear();
            picturepet.Image = null;
            PetSexTextBox.Clear();

            var obj = new CounterClass
            {
                id = Id.ToString()
            };

            SetResponse response = await client.SetTaskAsync("Couter/Node", obj);
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }
}

```

Hình: Code minh họa Form Post

Ở chức năng post – đăng tải bài viết, cho phép thực hiện đăng bài với 2 lựa chọn: tìm kiếm thú cưng thất lạc (Post for Finding) hoặc cho phép người khác nhận nuôi thú cưng (Post for adopting). Nếu chọn cho phép nhận nuôi button “MoreDetail” sẽ kích hoạt cho phép người dùng điền thêm thông tin như sức khỏe, người chủ trước,... của thú cưng, tất cả thông tin sẽ được gửi lên firebase. Còn với lựa chọn “Post for Finding”, chức năng của button “MoreDetail” sẽ không hoạt động.

```

private async void Adopt_ButtonClick(object sender, EventArgs e)
{
    //MessageBox.Show("ID = " + id_petAdopt.ToString());
    FirebaseResponse response = await client.GetTaskAsync("Post/" + id_petAdopt.ToString());
    PostData postData = response.ResultAs<postData>();
    postData postData = new postData();
    postData = obj;
    postData.isAdopted = true;
    FirebaseResponse res = await client.UpdateTaskAsync("Post/" + id_petAdopt.ToString(), postData);
    PostData result = res.ResultAs<postData>();
    isCheck = true;
    listPost.Clear();
    testOn();
    isCheck = false;
    MessageBox.Show("This Pet Adopted", "Congarts");

    string postEmail = result.Email;
    IFirebaseConfig config = new FirebaseConfig
    {
        AuthSecret = "BFp0TAb8sUuV5tkaRZDWLk5Nz0drFLJWr2NkqPxt",
        BasePath = "https://registerandlogin-31e76-default-rtdb.firebaseio.com/"
    };
    IFireBaseClient Client;
    Client = new FireSharp.FirebaseClient(config);
    MailMessage message = new MailMessage();
    string from, pass, messbody;
    from = "adoptionpet818@gmail.com";
    pass = "nqrhurlflgaphvqr";
    string to = postEmail;
    string petName = PetName.Text;
    string user = ShareVariable.Username;
    FirebaseResponse response1 = await Client.GetTaskAsync("Users/" + user);
    User usr = response1.ResultAs<User>();
    string usremail = usr.Email;
    string usrphone = usr.Phone;
    string usrlocation = usr.Location;
    messbody = $"Your pet named {petName} has been adopted!\nHere is the information of Adopter\nName: {user}\nEmail: {usremail}\n";
    message.To.Add(to);

    message.From = new MailAddress(from);
    message.Body = messbody;
    message.Subject = "Pet Adopted";
    SmtpClient smtp = new SmtpClient("smtp.gmail.com");
    smtp.EnableSsl = true;
    smtp.Port = 587;
    smtp.DeliveryMethod = SmtpDeliveryMethod.Network;
    smtp.Credentials = new NetworkCredential(from, pass);
    try
    {
        smtp.Send(message);
        MessageBox.Show("Your info has been sent to the post user");
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    Adopt_Button.Hide();
    guna2Button1.Show();
}

```

```

private async void next_ButtonClick(object sender, EventArgs e)
{
    try
    {
        //MessageBox.Show("Phan tu trong listPost" + listPost.Count);
        if (current_num_id > listPost.Count - 1) current_num_id = 0;

        PostData obj = listPost[current_num_id];
        PetName.Text = obj.PetName;
        PetBirth.Text = obj.PetDob;
        PetSex.Text = obj.PetSex;
        PetType.Text = obj.PetSubtype;
        Height.Text = obj.Health_Pet.Height;
        Weight.Text = obj.Health_Pet.Weight;
        Vaccine.Text = obj.Health_Pet.Vaccinations;
        preOwner.Text = obj.Health_Pet.PreviousOwner;
        label2.Text = obj.DatePost;
        label1.Text = obj.NameClient;
        string imagestring = obj.imgstr;
        Image image = Base64StringIntoImage(imagestring);
        petImage.Image = image;
        petImage.SizeMode = PictureBoxSizeMode.StretchImage;
        id_petAdopt = obj.id;
        current_num_id++;
    } catch { }
    Adopt_Button.Show();
    guna2Button1.Hide();
}

```

Hình: Code minh họa Form Adopt

Form Adopt có 2 button thực hiện 2 chức năng chính:

- next_Button thực hiện duyệt và hiển thị thông tin thú cưng được đăng cho nhận nuôi lên màn hình.
- Adopt_Button thực hiện xác nhận nhận nuôi thú cưng có thông tin đang được hiển thị, đồng thời gửi thông báo và các thông tin cần thiết cho người đăng bài qua email

```

1 reference
private async void next_ButtonClick(object sender, EventArgs e)
{
    try
    {
        if (current_num_id > listPost.Count - 1) current_num_id = 0;

        PostData obj = listPost[current_num_id];
        PetName.Text = obj.PetName;
        PetBirth.Text = obj.PetDob;
        PetSex.Text = obj.PetSex;
        PetType.Text = obj.PetSubtype;
        label2.Text = obj.DatePost;
        label1.Text = obj.NameClient;
        string imagestring = obj.imgstr;
        Image image = Base64StringIntoImage(imagestring);
        petImage.Image = image;
        petImage.SizeMode = PictureBoxSizeMode.StretchImage;
        id_petFind = obj.id;
        current_num_id++;
    }
    catch { }
    Adopt_Button.Enabled = true;
}

2 references
public async void curentListPetnotFound()
{
    for (int i = 1; i <= total_id; i++)
    {
        FirebaseResponse response = await client.GetTaskAsync("Post/" + i.ToString());
        PostData obj = response.ResultAs<postData>();
        if (obj.isFinded == false)
            listPost.Add(obj);
    }
}

1 reference
private async void find_Load(object sender, EventArgs e)
{
    client = new FireSharp.FirebaseClient(config);
    FirebaseResponse resp = await client.GetTaskAsync("Couter/Node");
    CounterClass get = resp.ResultAs<CounterClass>();
    total_id = Int32.Parse(get.id);
    curentListPetnotFound();
}

```

```

1 reference
private async void Find_ButtonClick(object sender, EventArgs e)
{
    //MessageBox.Show("ID = " + id_petAdopt.ToString());
    FirebaseResponse response = await client.GetTaskAsync("Post/" + id_petFind.ToString());
    PostData obj = response.ResultAs<PostData>();
    PostData postData = new PostData();
    postData = obj;
    postData.isFinded = true;
    FirebaseResponse res = await client.UpdateTaskAsync("Post/" + id_petFind.ToString(), postData);
    PostData result = res.ResultAs<PostData>();
    isCheck = true;
    listPost.Clear();
    testOn();
    isCheck = false;
    MessageBox.Show("You have found a lost pet and reported it to its owner!", "Thankful!");

    string postEmail = result.Email;
    IFirebaseConfig config = new FirebaseConfig
    {
        AuthSecret = "BFp0TA8sUuV5tkaRZDWlk5Nz0drFLJWr2NkqPxt",
        BasePath = "https://registerandlogin-31e76-default-rtdb.firebaseio.com/"
    };
    IFirebaseClient Client;
    Client = new FireSharp.FirebaseClient(config);
    MailMessage message = new MailMessage();
    string from, pass, messbody;
    from = "adoptionpet818@gmail.com";
    pass = "nqrhurlflgaphvqr";
    string to = postEmail;
    string petName = PetName.Text;
    string user = ShareVariable.Username;
    FirebaseResponse response1 = await Client.GetTaskAsync("Users/" + user);
    User usr = response1.ResultAs<User>();
    string usremail = usr.email;
    string usrphone = usr.phone;
    string usrlocation = usr.location;
    messbody = $"Good news! Your lost pet named {PetName} has been found!\nHere is the information of the Finder:\nName: {user}\nEmail: {usremail}\nPhone number: {usrphone}\nLocation: {usrlocation}";
    message.To.Add(to);
    message.From = new MailAddress(from);
    message.Body = messbody;
    message.Subject = "Pet Found";
    SmtpClient smtp = new SmtpClient("smtp.gmail.com");
    smtp.EnableSsl = true;
    smtp.Port = 587;
    smtp.DeliveryMethod = SmtpDeliveryMethod.Network;
    smtp.Credentials = new NetworkCredential(from, pass);
    try
    {
        smtp.Send(message);
        MessageBox.Show("Your info has been sent to the post user");
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    Adopt_Button.Enabled = false;
}

1 reference
public void testOn()
{
    while (true)
    {
        if (isCheck) currentListPetnotFound();
        break;
    }
}

```

Hình: Code minh họa Form Find

Form Find có chức năng gần giống form Adopt, cũng có 2 button thực hiện 2 chức năng chính:

- next_Button thực hiện duyệt và hiển thị thông tin thú cưng được đăng tìm kiếm lên màn hình.

- Find_Button thực hiện xác nhận tìm thấy thú cưng có thông tin đang được hiển thị, đồng thời gửi thông báo và các thông tin cần thiết cho người đăng bài qua email

```

1 reference
private async void saveButton_Click(object sender, EventArgs e)
{
    if (string.IsNullOrEmpty(usernametextbox.Text) || string.IsNullOrEmpty(emailtextbox.Text))
    {
        // Check if textbox is Empty
        MessageBox.Show("Please Put Username or Password.", "Notification", MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
    else
    {
        if (passtextbox.Text != cfpasstextbox.Text) MessageBox.Show("Password does not match!");
        string hashPass = ComputeSha256Hash(cfpasstextbox.Text);
        try
        {
            FirebaseResponse res = await client.GetTaskAsync("Users/" + ShareVariable.Username);
            User usr = res.ResultAs<User>();
            string usr_avatar = usr.avatar.ToString();
            if (isClickedUpload)
            {
                var user = new User
                {
                    username = usernametextbox.Text,
                    password = hashPass,
                    email = emailtextbox.Text,
                    phone = phonetextbox.Text,
                    location = locationtextbox.Text,
                    avatar = ImageIntoBase64String(newavatarpic),
                };
                FirebaseResponse response2 = await client.UpdateTaskAsync("Users/" + usernametextbox.Text, user);
            }
            else
            {
                var user = new User
                {
                    username = usernametextbox.Text,
                    password = hashPass,
                    email = emailtextbox.Text,
                    phone = phonetextbox.Text,
                    location = locationtextbox.Text,
                    avatar = usr_avatar,
                };
                isClickedUpload = false;
                FirebaseResponse response2 = await client.UpdateTaskAsync("Users/" + usernametextbox.Text, user);
            }
            MessageBox.Show("Update data successfully!");
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }
}

```

```

1 reference
private void uploadButton_Click(object sender, EventArgs e)
{
    OpenFileDialog openFileDialog = new OpenFileDialog();
    openFileDialog.Filter = "Image Files (*.jpg, *.jpeg, *.png) | *.jpg; *.jpeg; *.png";
    if (openFileDialog.ShowDialog() == DialogResult.OK)
    {
        newavatarpic.Load(openFileDialog.FileName);
        newavatarpic.SizeMode = PictureBoxSizeMode.StretchImage;
        isClickedUpload = true;
    }
}
1 reference
public static string ImageIntoBase64String(PictureBox pbx)
{
    MemoryStream ms = new MemoryStream();
    pbx.Image.Save(ms, pbx.Image.RawFormat);
    return Convert.ToString(ms.ToArray());
} //chuyen hinh sang base64
1 reference
public static System.Drawing.Image Base64StringIntoImage(string base64String)
{
    byte[] imgBytes = Convert.FromBase64String(base64String);
    using (MemoryStream ms = new MemoryStream(imgBytes))
    {
        System.Drawing.Image image = System.Drawing.Image.FromStream(ms);
        return new Bitmap(image);
    }
}

```

Hình: Code minh họa Form Setting

Chức năng setting cho phép người dùng chỉnh sửa thông tin tài khoản bao gồm email, số điện thoại, vị trí, mật khẩu, ảnh đại diện và cập nhật dữ liệu mới lên firebase.

```

1 reference
void Connect()
{
    //IP là địa chỉ của server.Khởi tạo địa chỉ IP và socket để kết nối
    IP = new IPEndPoint(IPAddress.Parse("127.0.0.1"), 8080);
    Client = new Socket(AddressFamily.InterNetwork, SocketType.Stream, ProtocolType.IP);
    //bắt đầu kết nối. Nếu ko kết nối được thì hiện thông báo
    try
    {
        Client.Connect(IP);
    }
    catch
    {
        MessageBox.Show("Lỗi kết nối", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Warning);
        return;
    }

    //tạo luồng lắng nghe server khi vừa kết nối tới
    Thread listen = new Thread(Receive);
    listen.IsBackground = true;
    listen.Start();
}

```

```

    void Send()
    {
        string text = ShareVariable.Username + ":" + guna2TextBox2.Text;
        //nếu textboc khác rỗng thì mới gửi tin
        if (guna2TextBox2.Text != string.Empty)
        {
            Client.Send(Serialize(text));
            ListViewItem item = new ListViewItem("");
            item.SubItems.Add(guna2TextBox2.Text);
            listView2.Items.Add(item);
            guna2TextBox2.Text = string.Empty;
        }
    }

    //nhận dữ liệu
    void Receive()
    {
        try
        {
            while (true)
            {
                //khai báo mảng byte để nhận dữ liệu dưới dạng byte
                byte[] data = new byte[1024 * 5000];
                Client.Receive(data);
                //chuyển data từ dạng byte sang dạng string
                string message = (string)Deserialiaze(data);
                ListViewItem item = new ListViewItem(message);
                item.SubItems.Add("");
                listView2.Items.Add(item);
            }
        }
        catch
        {
            Close();
        }
    }
}

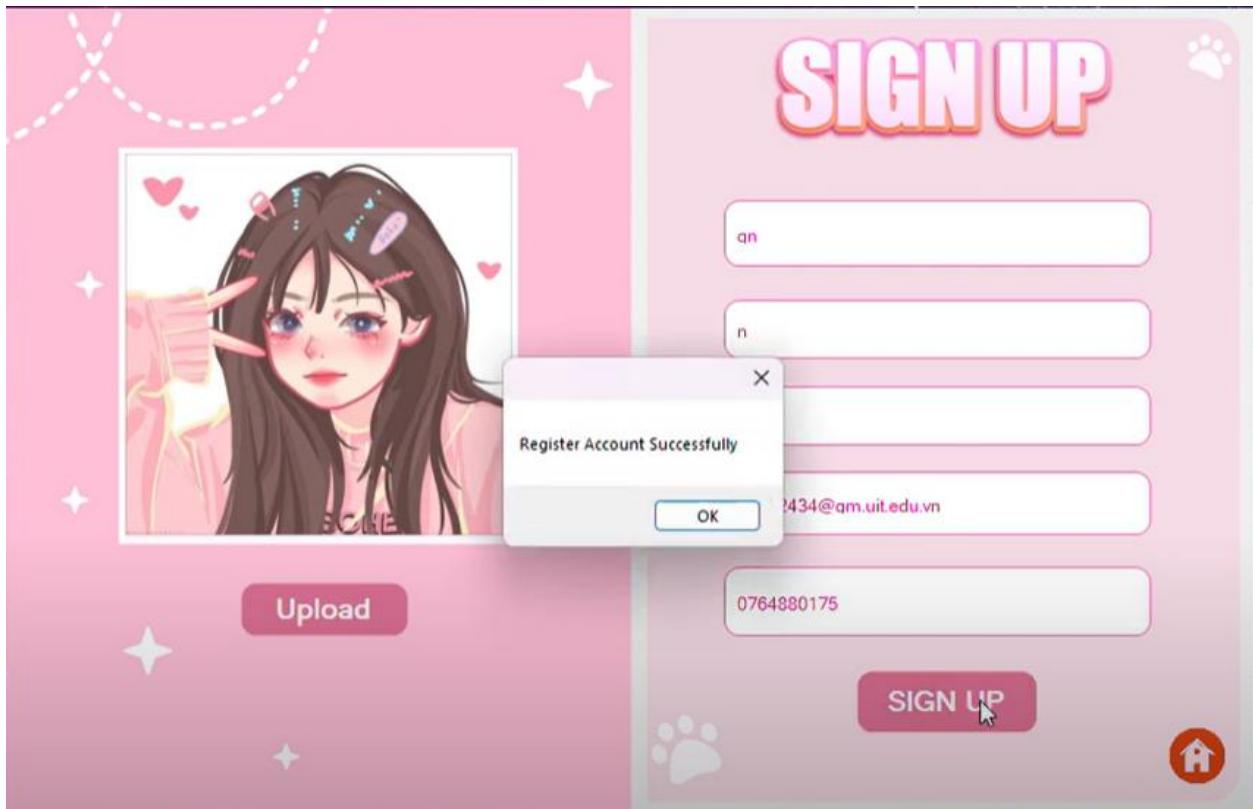
```

Hình: Code minh họa form chat box

Tính năng chat cho phép chat giữa hai người và chat với tất cả mọi người bằng cách tạo kết nối TCP theo mô hình client – server và thực hiện gửi và nhận dữ liệu giữa client và server.

CHƯƠNG 5. KIỂM THỬ

5.1. Kiểm thử chức năng đăng ký



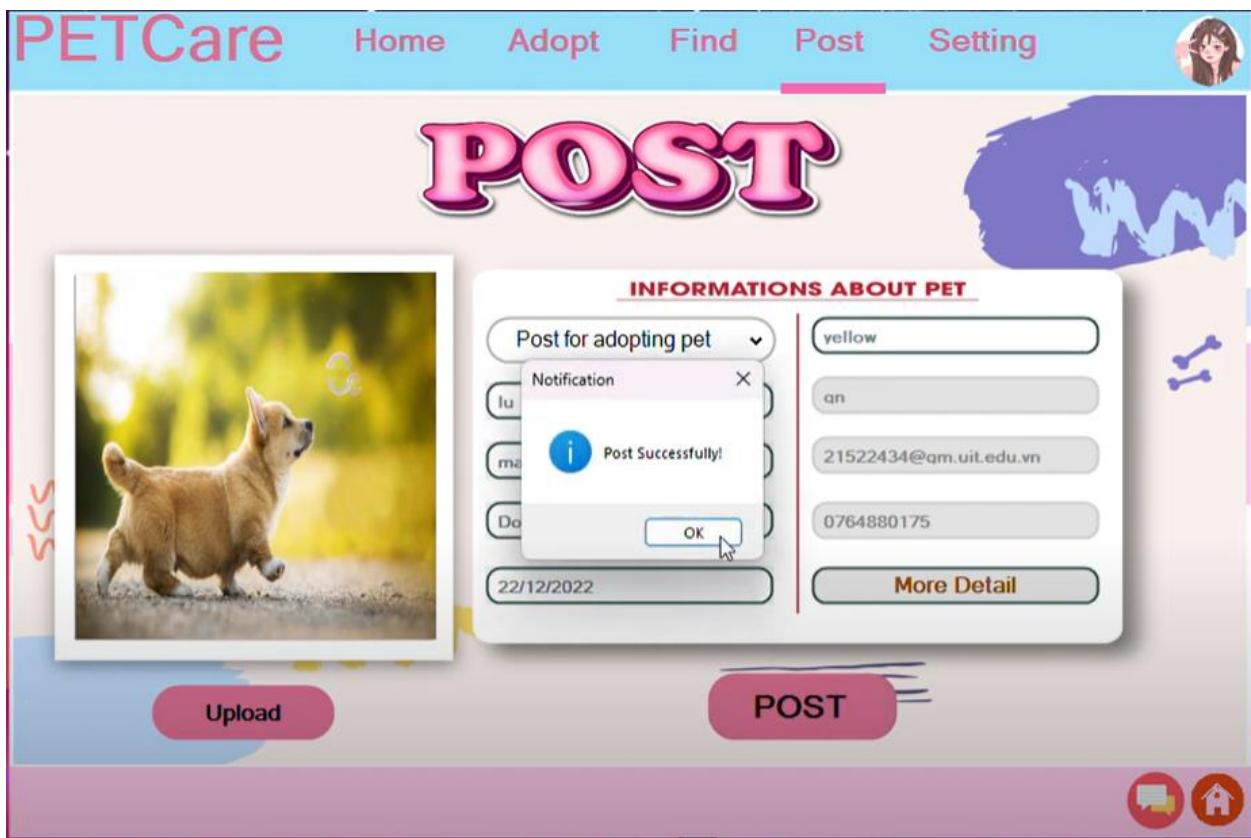
Hình 5.1 Kiểm thử chức năng đăng ký

5.2. Kiểm thử chức năng đăng nhập

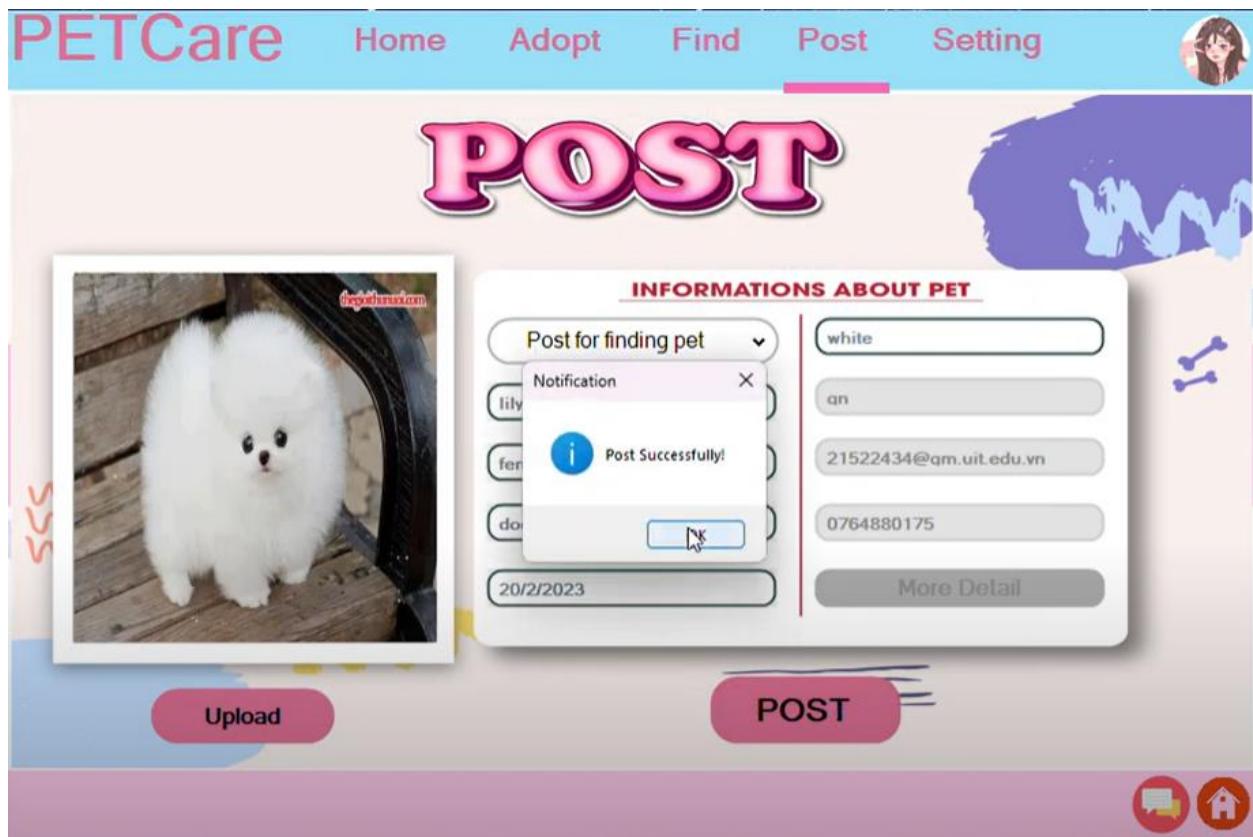


Hình 5.2 Kiểm thử chức năng đăng nhập

5.3. Kiểm thử chức năng trang đăng thông tin (Form Post)

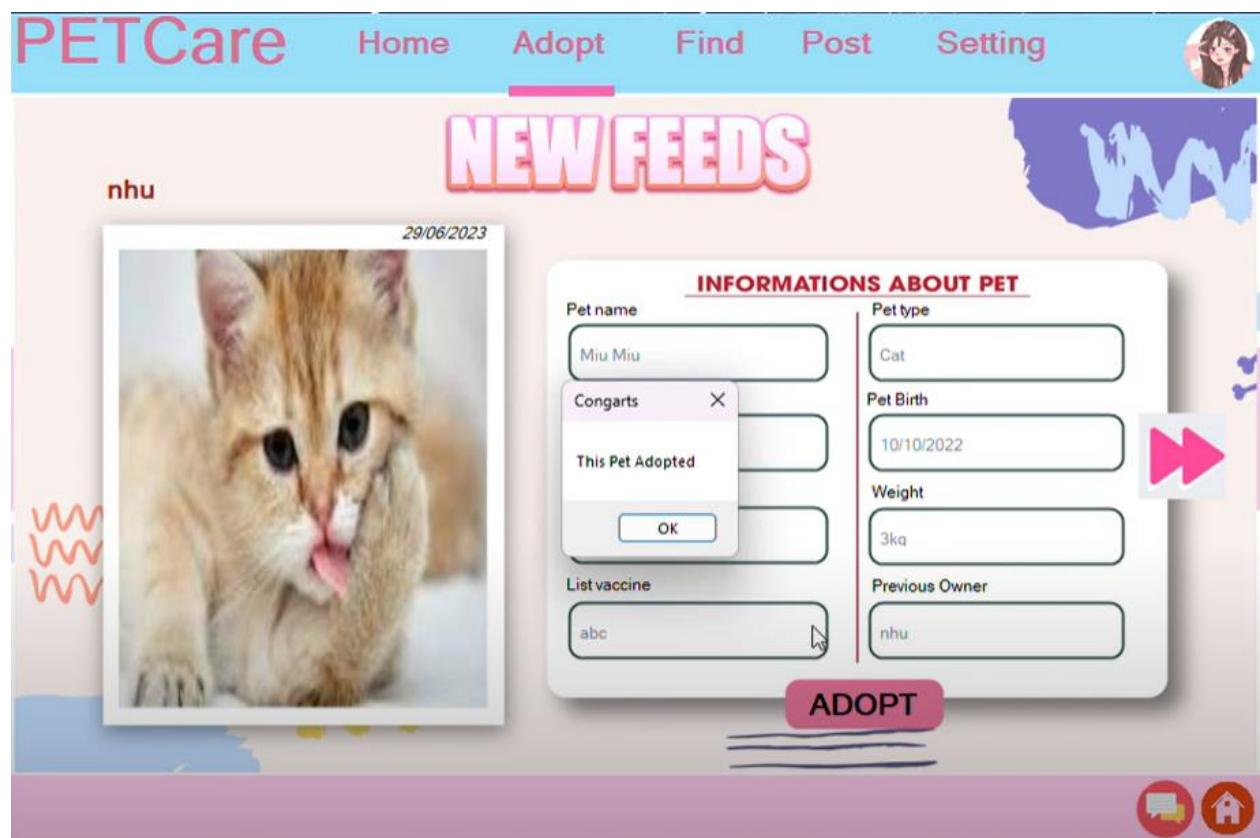


Hình 5.3 Kiểm thử chức năng trang đăng thông tin cho nhận nuôi



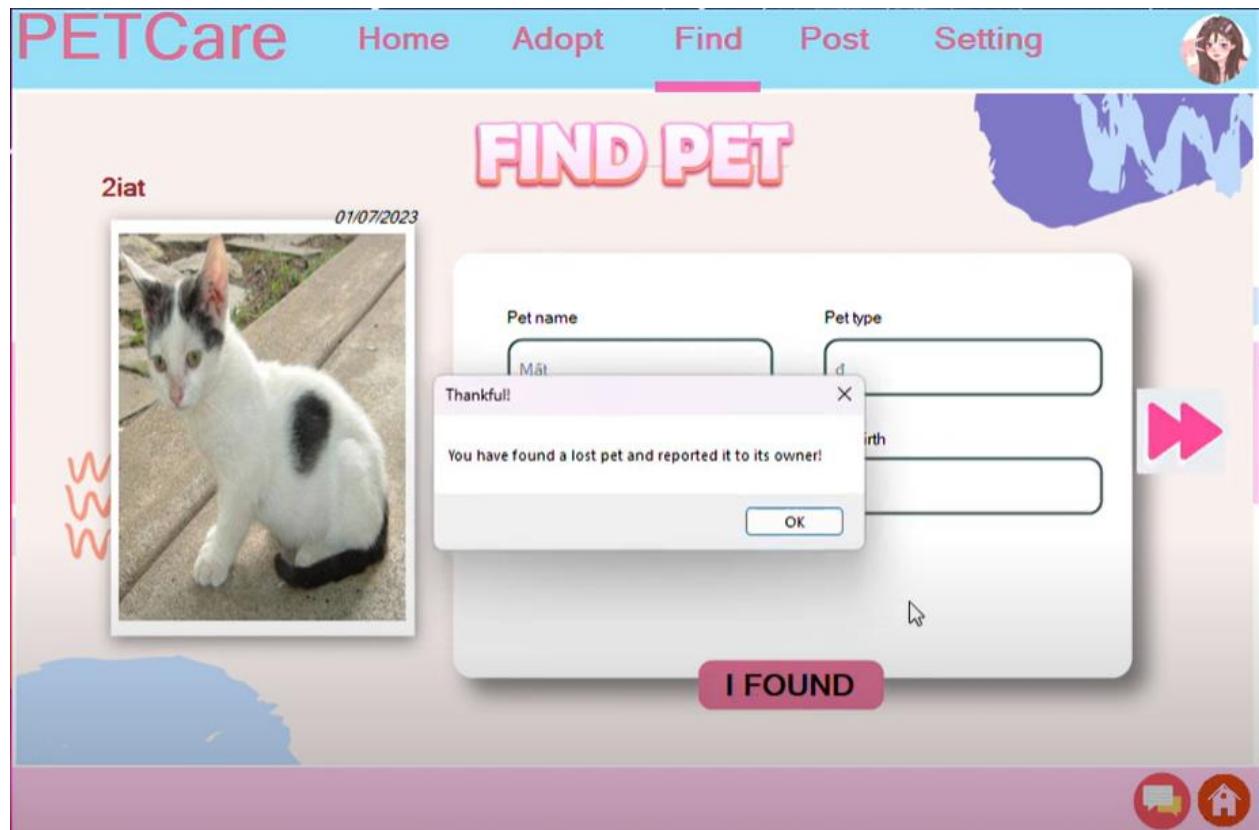
Hình 5.4 Kiểm thử chức năng trang đăng thông tin tìm pet bị mất

5.4. Kiểm thử chức năng trang nhận nuôi thú cưng (Form Adopt)



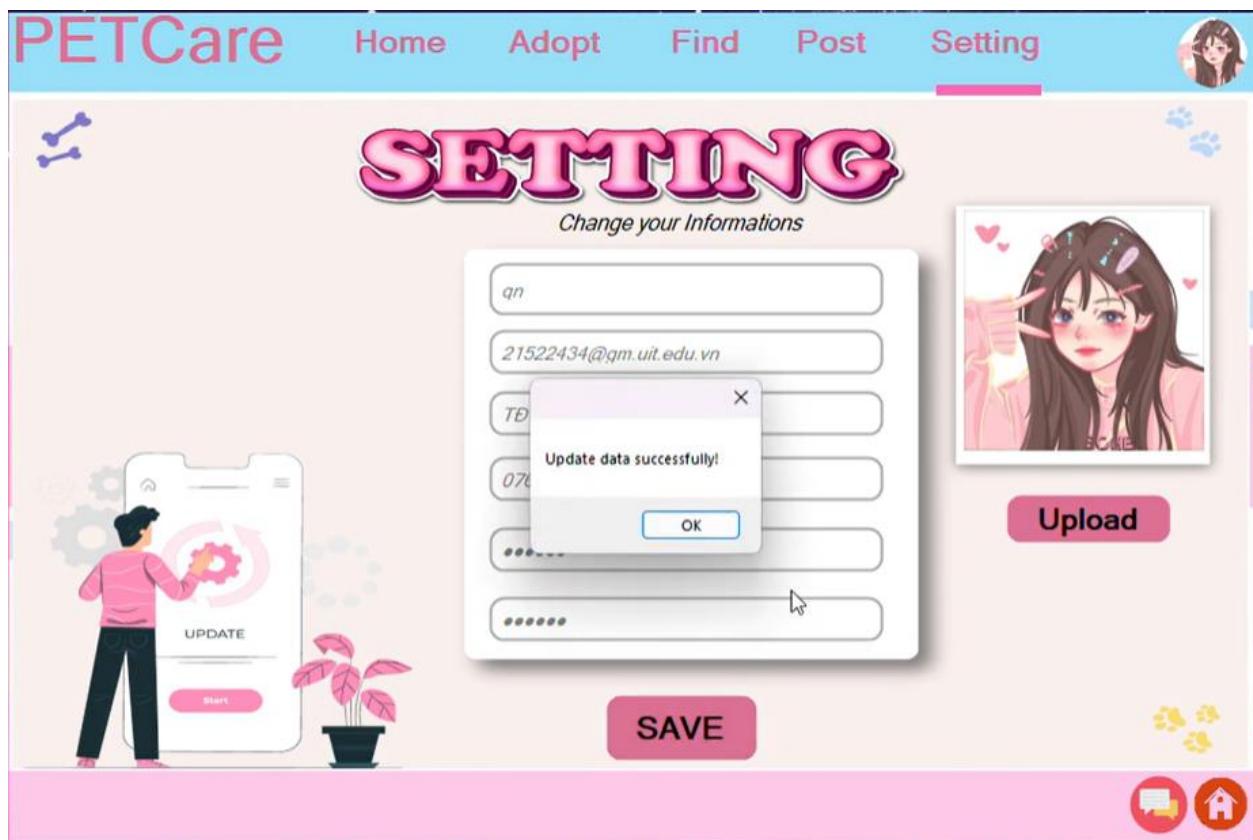
Hình 5.5 Kiểm thử chức năng trang nhận nuôi thú cưng

5.5. Kiểm thử chức năng trang tìm kiếm thú cưng (Form Find)



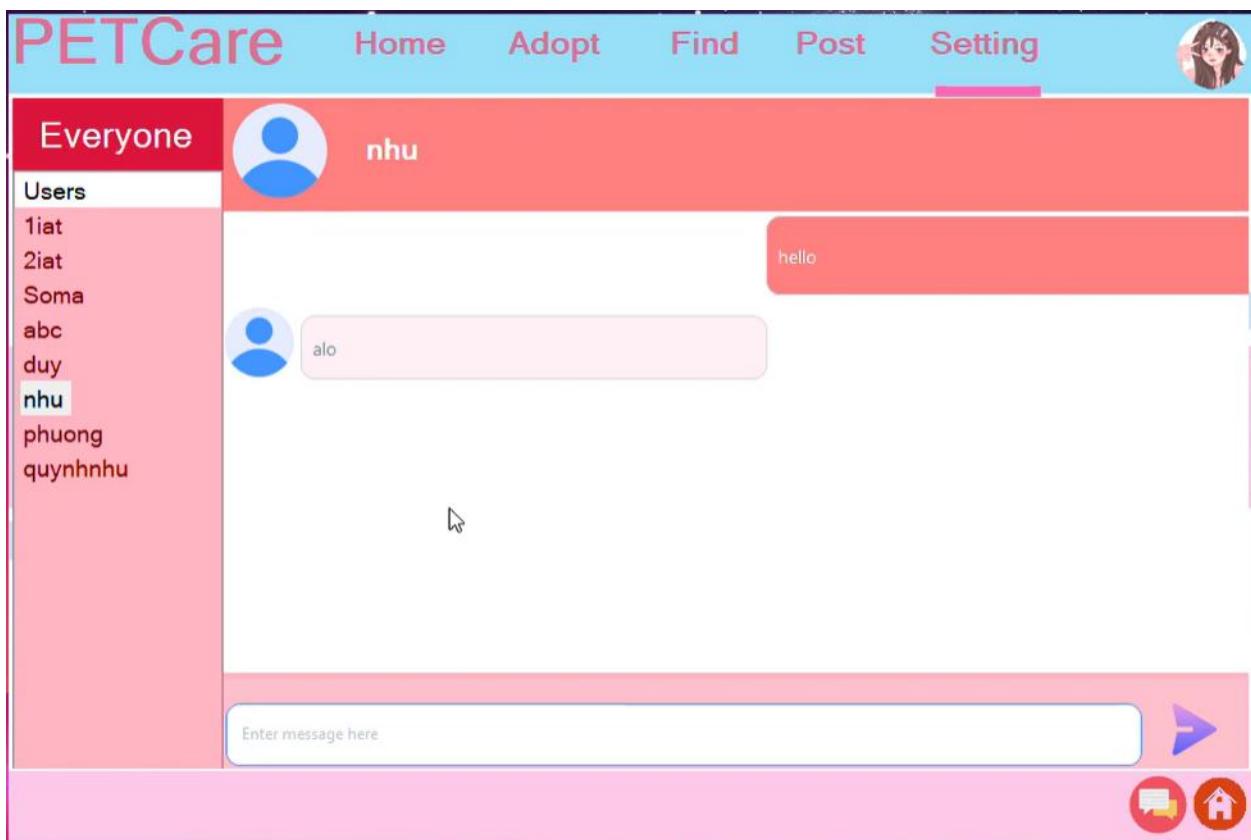
Hình 5.6 Kiểm thử chức năng trang tìm kiếm thú cưng

5.6. Kiểm thử chức năng cài đặt người dùng (Form setting)



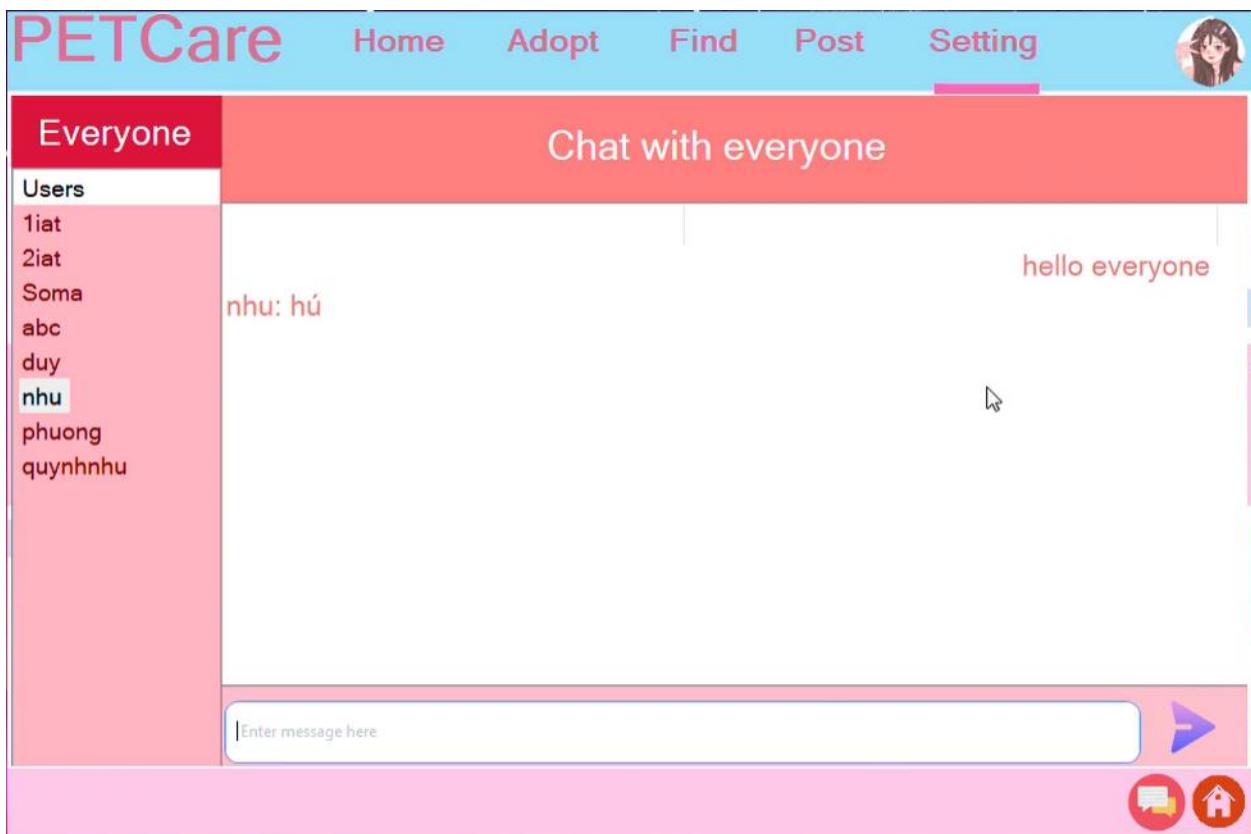
Hình 5.7 Kiểm thử chức năng cài đặt người dùng

5.7. Kiểm thử chức năng nhắn tin giữa 2 users



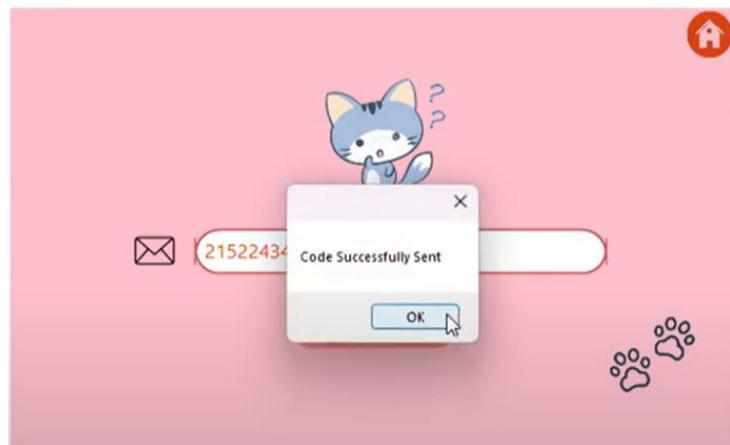
Hình 5.8 Kiểm thử chức năng nhắn tin hai người

5.8. Kiểm thử chức năng nhắn tin với tất cả users

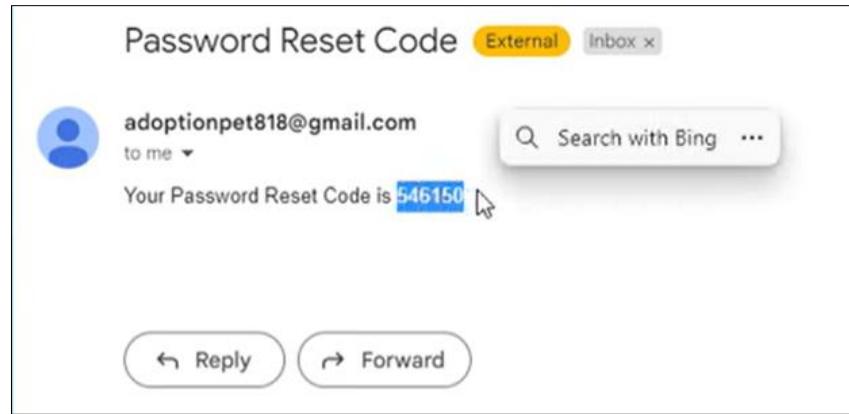


Hình 5.9 Kiểm thử chức năng nhắn tin với server (nhiều người)

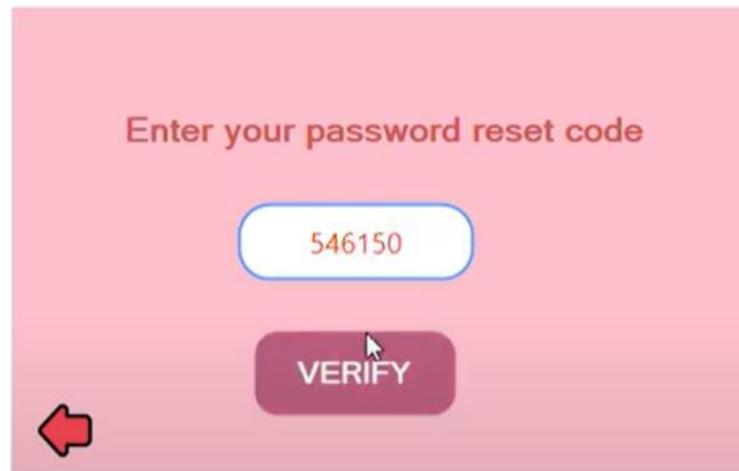
5.9. Kiểm thử chức năng quên mật khẩu



Hình 5.10 Kiểm thử chức năng gửi mã OTP



Hình 5.11 Mã OTP được gửi đến email



Hình 5.12 Nhập mã OTP



Hình 5.13 Đổi mật khẩu thành công



Hình 5.14 Đăng nhập với mật khẩu vừa đổi

5.10. Kiểm thử về hiệu suất

Chạy ổn định: Ứng dụng Pet Care đã hoạt động chạy ổn định dưới tải công việc khác nhau. Thời gian phản hồi của hệ thống không tăng đáng kể khi có nhiều người dùng truy cập cùng một lúc. Tải CPU và tải mạng đều ổn định trong phạm vi chấp nhận được.

Ôn trên môi trường kết nối mạng: Ứng dụng Pet Care đã hoạt động tốt trên các môi trường mạng khác nhau. Tốc độ truyền dữ liệu và độ trễ mạng đáp ứng yêu cầu của người dùng. Kết nối mạng đủ ổn định để đảm bảo truy cập liên tục vào ứng dụng.

Xử lý dữ liệu tăng thêm: Ứng dụng Pet Care vẫn duy trì hiệu suất tốt khi có dữ liệu tăng thêm. Khả năng xử lý và phản hồi của hệ thống không bị ảnh hưởng đáng kể khi có số lượng lớn người dùng và dữ liệu trong cơ sở dữ liệu.

CHƯƠNG 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

6.1. Kết luận

Sau quá trình phát triển, sản phẩm đã hoàn thiện các chức năng cơ bản được đề ra ban đầu. Tuy nhiên, vẫn còn một số hạn chế cần được lưu ý. Dữ liệu của ứng dụng được lưu trữ trên Firebase và sử dụng cơ sở dữ liệu non-SQL, do đó không thể thực hiện các truy vấn phức tạp. Thay vào đó, phải duyệt qua các đối tượng thông qua ID, gây giảm hiệu suất trong một số trường hợp.

Tuy nhiên, sản phẩm vẫn có thể ứng dụng thực tế và phù hợp với các môi trường có nhu cầu cao về việc nhận nuôi và tìm kiếm thú cưng.

Trong quá trình thực hiện đồ án, chúng tôi đã áp dụng và ứng dụng kiến thức lập trình mạng đã học vào các chức năng của sản phẩm. Ví dụ, chúng tôi đã xây dựng tính năng chat qua mạng LAN, cho phép người dùng gửi tin nhắn và trao đổi thông tin. Chúng tôi cũng đã triển khai tính năng gửi email đến một người dùng, tạo sự tiện lợi trong việc thông báo và liên lạc.

Ngoài ra, chúng tôi cũng đã áp dụng Firebase làm nền tảng lưu trữ dữ liệu cho sản phẩm. Điều này giúp chúng tôi lưu trữ và quản lý dữ liệu một cách linh hoạt và dễ dàng.

Thêm vào đó, chúng tôi đã sử dụng thư viện Guna2UI để thiết kế giao diện người dùng. Thư viện này cung cấp các thành phần giao diện thân thiện và tùy chỉnh, giúp tạo ra trải nghiệm sử dụng tốt cho người dùng.

Tổng kết, qua quá trình thực hiện đồ án, chúng tôi đã áp dụng thành công kiến thức lập trình mạng vào việc phát triển các chức năng trong sản phẩm. Sản phẩm có thể ứng dụng thực tế và đáp ứng nhu cầu của môi trường liên quan đến việc nhận nuôi và tìm kiếm thú cưng ở mức cơ bản.

6.2. Hướng phát triển

Đề tài này mang trong mình tiềm năng phát triển để có thể ứng dụng rộng rãi trong thực tế. Trong tương lai, chúng tôi đề xuất mở rộng ứng dụng bằng việc thêm vào các tính năng sau đây:

- **Công cụ quản lý sức khỏe thú cưng:** Tính năng này cho phép người dùng ghi chú và quản lý thông tin liên quan đến sức khỏe thú cưng như tiêm phòng, cân nặng, và

các lịch trình chăm sóc. Người dùng có thể theo dõi và nhận thông báo nhắc nhở về các hoạt động quan trọng để đảm bảo sức khỏe tốt cho thú cưng của mình.

- **Công cụ tư vấn và hỗ trợ nhanh:** Tính năng này cung cấp hỗ trợ và tư vấn nhanh chóng cho người dùng. Người dùng có thể gửi câu hỏi hoặc yêu cầu hỗ trợ và nhận được phản hồi từ các chuyên gia hoặc cộng đồng người dùng khác. Điều này giúp người dùng có thêm thông tin hữu ích và giải đáp các thắc mắc liên quan đến thú cưng của họ.
- **Công cụ tìm kiếm cơ sở thú y gần nhất:** Đây là một tính năng hữu ích cho phép người dùng tìm kiếm và xác định các cơ sở thú y gần nhất dựa trên vị trí địa lý của họ. Tính năng này giúp người dùng dễ dàng tìm được các dịch vụ y tế chăm sóc thú cưng nhanh chóng và thuận tiện.

Với việc bổ sung những tính năng này, sản phẩm sẽ trở nên linh hoạt hơn và đáp ứng được nhiều nhu cầu khác nhau của người dùng. Đồng thời, chúng tôi sẽ tiếp tục cải thiện hiệu suất và tính bảo mật của ứng dụng, đồng thời tối ưu hóa giao diện người dùng để mang lại trải nghiệm tốt nhất cho người dùng.

TÀI LIỆU THAM KHẢO

[System.Net.Sockets Namespace | Microsoft Learn](#)

[Kết nối C# với Firebase - Freetuts](#)

[Cách lấy dữ liệu từ Firebase về C# Winforms \(freetuts.net\)](#)

[Cách Insert dữ liệu từ C# Winforms lên Firebase \(freetuts.net\)](#)

[Cách Update dữ liệu từ C# Winforms lên Firebase \(freetuts.net\)](#)

[System.Net.Mail Namespace | Microsoft Learn](#)

[NuGet Gallery | Guna.UI2.WinForms 2.0.4.4](#)

PHỤ LỤC

Phụ lục 1: Hình ảnh real-time database trên Firebase

- RegisterAndLogin



- Posts

The screenshot shows the Firebase Realtime Database interface with the following structure:

```
https://doannhom7-a71b2-default-rtdb.firebaseio.com/
  Couter
  Node
    id: "16"
  Post
    1
      DatePost: "29/06/2023"
      Email: "voquynhnhu765@gmail.com"
      Health_Pet
        FavoriteFood: "fish"
        Height: "10cm"
        PreviousOwner: "nhu"
        Vaccinations: "abc"
        Weight: "3kg"
      NameClient: "nhu"
      PetColor: "yellow"
      PetDob: "10/10/2022"
      PetName: "Miu Miu"
      PetSex: "Female"
      PetSubtype: "Cat"
      Phone: "123456"
      id: 1
      imgstr: "/9j/4AAQSkZJRgABAQAAAAAAAD/2wBDAkGBwgHBgkIBwgKCgk"
      isAdopted: true
      isFinded: true
    2
    3
    4
    5 + -
```

The 'Post' node contains a child node '1' with the following data:

- DatePost: "29/06/2023"
- Email: "voquynhnhu765@gmail.com"
- Health_Pet
 - FavoriteFood: "fish"
 - Height: "10cm"
 - PreviousOwner: "nhu"
 - Vaccinations: "abc"
 - Weight: "3kg"
- NameClient: "nhu"
- PetColor: "yellow"
- PetDob: "10/10/2022"
- PetName: "Miu Miu"
- PetSex: "Female"
- PetSubtype: "Cat"
- Phone: "123456"
- id: 1
- imgstr: "/9j/4AAQSkZJRgABAQAAAAAAAD/2wBDAkGBwgHBgkIBwgKCgk"
- isAdopted: true
- isFinded: true

Below the '1' node, there are three more nodes labeled 2, 3, and 4, each with a circular arrow icon. At the bottom, there is a node labeled 5 with a rectangular input field containing the value "5", a plus sign (+), and a trash can icon.

Phụ lục 2: Đọc thêm

Timeline: [Timeline Đồ án môn học - Google Trang tính](#)

Github: https://github.com/Phuongnguyen2710/UI_PetCare