

University of Information Technology  
Faculty of Computer Network and Communications



**BÁO CÁO ĐỒ ÁN CUỐI KÌ**

**Môn học: Lập trình mạng căn bản**

**Lớp: NT106.021.ANTT**

**GVHD: Trần Hồng Nghi**

**THÀNH VIÊN THỰC HIỆN (Nhóm 07):**

STT	Họ và tên	MSSV
1	Phạm Minh Tân	22521310
2	Lương Cao Thắng	22521328
3	Bùi Phương Đại	22520180
4	Phạm Xuân Tuấn Anh	22520071

## *Lời nói đầu*

*Lời nói đầu tiên cho tất cả thành viên nhóm 7 gửi lời cảm ơn tới giảng viên hướng dẫn Ths.Trần Hồng Nghi đã tận tâm dạy, hỗ trợ về phương diện kiến thức, giải đáp mọi thắc mắc của sinh viên, hỗ trợ giải quyết các vấn đề khó khăn của sinh viên. Qua đó, chúng em đã tích lũy được thêm nhiều kiến thức cùng với kinh nghiệm làm việc nhóm thông qua môn học.*

*Mặc dù tất cả thành viên cố gắng để hoàn thành nhiệm vụ của mình. Nhưng vẫn còn nhiều sai sót do việc còn thiếu kinh nghiệm. Chúng em rất mong được thông cảm và nhận được những lời nhận xét và góp ý đến từ thầy cô và các bạn.*

*Nhóm chúng em xin chân thành cảm ơn.*

# MỤC LỤC

MỤC LỤC.....	3
DANH MỤC HÌNH ẢNH .....	5
DANH MỤC BẢNG.....	6
CHƯƠNG 1: TỔNG QUAN.....	7
1.1 Giới thiệu đề tài:.....	7
1.2 Sơ lược về cách thức hoạt động của ứng dụng: .....	7
1.3 Công cụ hỗ trợ:.....	7
1.4 Cơ sở dữ liệu: .....	8
CHƯƠNG 2: THIẾT KẾ HỆ THỐNG.....	10
2.1 Các chức năng của ứng dụng: .....	10
2.2 Mô hình tổng quan đăng ký: .....	12
2.3 Mô hình tổng quan đăng nhập: .....	13
2.4 Sử dụng cơ sở dữ liệu:.....	15
CHƯƠNG 3: XÂY DỰNG HỆ THỐNG ỨNG DỤNG CHAT.....	17
3.1 Các chức năng hỗ trợ: .....	17
3.1.1 Chức năng đăng kí, đăng nhập.....	17
3.1.2. Giao diện chính .....	21
CHƯƠNG 4: THỰC NGHIỆM ĐỒ ÁN .....	29
CHƯƠNG 5: KẾT QUẢ VÀ HƯỚNG PHÁT TRIỂN.....	30
5.1 Kết quả: .....	30
5.2 Hạn chế và khó khăn: .....	30
5.3 Hướng phát triển: .....	30
TÀI LIỆU THAM KHẢO: .....	31

PHỤ LỤC .....	32
---------------	----

## DANH MỤC HÌNH ẢNH

1.	Hình 1.1 Cơ sở dữ liệu lưu trữ UserData	17
2.	Hình 1.2 Cơ sở dữ liệu lưu trữ	17
3.	Hình 2. Giao diện đăng ký của app	18
4.	Hình 3. Giao diện đăng nhập của app	19
5.	Hình 4. Tính năng xác thực bằng email	20
6.	Hình 5. Tính năng khôi phục lại mật khẩu bằng email.	21
7.	Hình 6. Giao diện MainMenu của app	22
8.	Hình 7. Giao diện trang chủ của app	23
9.	Hình 8. Giao diện xác nhận tên user tham gia Chat	23
10.	Hình 9 . Giao diện hiện thị tìm, tạo, xóa các nhóm	24
11.	Hình 10 . Giao diện chat của user admin	25
12.	Hình 11 . Giao diện chat của user member	26
13.	Hình 12 . Giao diện mã hóa file	27
14.	Hình 13 . Giao diện đóng góp ý kiến	28

## DANH MỤC BẢNG

# CHƯƠNG 1: TỔNG QUAN

## 1.1 Giới thiệu đề tài:

- Trong thời đại hiện nay, bảo vệ quyền riêng tư và bảo mật thông tin là ưu tiên hàng đầu. Phát triển một ứng dụng chat với mã hóa end-to-end bằng C# không chỉ đáp ứng nhu cầu bảo mật mà còn mang lại trải nghiệm người dùng tốt và hiệu suất cao. Đây là một hướng đi tiềm năng và cần thiết trong bối cảnh công nghệ ngày càng phát triển và các mối đe dọa an ninh mạng ngày càng tinh vi.

- Mã hóa end-to-end (E2EE) là một phương pháp bảo mật trong đó dữ liệu được mã hóa trên thiết bị của người gửi và chỉ được giải mã trên thiết bị của người nhận. Điều này đảm bảo rằng không ai, kể cả nhà cung cấp dịch vụ, có thể truy cập nội dung của các tin nhắn khi chúng đang truyền qua mạng.

- Nhận thấy điều đó, nhóm chúng em đã quyết định chọn đề tài ứng dụng chat, mã hóa end-to-end để phù hợp với tình hình hiện tại

## 1.2 Sơ lược về cách thức hoạt động của ứng dụng:

- Hệ thống máy chủ: nhận thấy các tiện ích mà firebase cung cấp với lưu trữ và đồng bộ dữ liệu theo thời gian thực:
  - + Sử dụng firestore để lưu các UserData sau khi đăng ký, để lấy thông tin cho các lần đăng nhập sau trong bao gồm các thông tin cơ bản như: tên hiển thị, các thông tin cá nhân như email, giới tính, ..... đặc biệt có biến isLoggedIn để kiểm tra xem đã có đăng nhập tài khoản ở ứng dụng khác chưa.
  - +Sau đó truy xuất UserData đã lưu ở firestore xuống gửi realtime database.
- Mã hóa và giải mã tin nhắn, icon và file: sau khi mã file (hình ảnh, tệp .doc, pdf,.....) thì ta sẽ không xem được, chỉ sau khi giải mã thì mới có thể xem được.
- Đóng góp ý kiến và báo lỗi: khi nhập thông tin, đóng góp ý kiến thì sẽ được gửi đến mail của admin và sẽ được khắc phục hoặc đáp ứng yêu cầu của người sử dụng.

## 1.3 Công cụ hỗ trợ:

### **1.3.1 Ngôn ngữ lập trình C#:**

- C# (C Sharp) là một ngôn ngữ lập trình hiện đại, hướng đối tượng và được xây dựng trên nền tảng hai ngôn ngữ mạnh nhất là C++ và java, được phát triển bởi Microsoft ra mắt lần đầu năm 2000 cùng với .NET Framework. Nó được thiết kế để đơn giản, mạnh mẽ và an toàn, với mục tiêu chính là phát triển các ứng dụng trên nền tảng .NET.

- C# với hỗ trợ mạnh mẽ từ .NET Framework giúp cho việc tạo ứng dụng Windows Forms, WPF (Windows Presentation Foundation),..... trở nên rất dễ dàng.

#### **a) Ưu điểm của ngôn ngữ C#:**

- Là một trong những ngôn ngữ mã nguồn mở thuần hướng đối tượng, được sử dụng để lập trình cho windows, dễ học và sử dụng.
- Được hỗ trợ bởi Microsoft, đảm bảo cập nhật liên tục và tài liệu phong phú.
- C# hỗ trợ thư viện .NET nhẹ, dễ cài đặt và hoàn toàn miễn phí.
- Hỗ trợ phát triển ứng dụng trên mọi nền tảng.

#### **b) Nhược điểm của ngôn ngữ C#:**

- Cần có .NET runtime để chạy ứng dụng, gây khó khăn trên các hệ thống không hỗ trợ .NET.
- C# chỉ đem lại hiệu quả tốt nhất với các lập trình viên trên nền Windows.
- Mặc dù mạnh mẽ, C# không đa dụng và dễ dàng như Python trong một số tình huống, đặc biệt là trong khoa học dữ liệu và machine learning.

### **1.4 Cơ sở dữ liệu:**

Firebase là một nền tảng phát triển ứng dụng di động và web được Google cung cấp, bao gồm nhiều dịch vụ như cơ sở dữ liệu thời gian thực, xác thực, lưu trữ, và các công cụ phân tích. Trong bối cảnh phát triển ứng dụng chat với mã hóa end-to-end (E2EE), Firebase có thể đóng vai trò quan trọng như sau:

- Cơ Sở Dữ Liệu Thời Gian Thực (Firebase Realtime Database và Firestore)
- Đồng Bộ Dữ Liệu Tức Thì: Firebase Realtime Database và Firestore cho phép đồng bộ hóa dữ liệu giữa các thiết bị trong thời gian thực, giúp



tin nhắn được cập nhật ngay lập tức trên tất cả các thiết bị của người dùng.

- Lưu Trữ Tin Nhắn: Dữ liệu chat được lưu trữ dưới dạng JSON (với Realtime Database) hoặc tài liệu (với Firestore), dễ dàng quản lý và truy xuất.
- Quyền Truy Cập và Bảo Mật: Firebase cung cấp các quy tắc bảo mật để kiểm soát quyền truy cập vào cơ sở dữ liệu, đảm bảo chỉ những người dùng được ủy quyền mới có thể đọc hoặc ghi dữ liệu.

- Firebase Storage và Firebase Realtime Database: Lưu trữ tệp đính kèm: Firebase Storage có thể được sử dụng để lưu trữ các tệp đính kèm như hình ảnh, video, và tài liệu được chia sẻ trong cuộc trò chuyện.

**a) Ưu điểm của realtimedatabase và firestore:**

- Firebase Realtime Database và Firestore đều cung cấp khả năng đồng bộ hóa dữ liệu tức thì giữa các thiết bị. Điều này rất hữu ích cho các ứng dụng chat yêu cầu cập nhật tin nhắn ngay lập tức.
- Cả hai dịch vụ đều hỗ trợ hoạt động ngoại tuyến, cho phép người dùng tiếp tục sử dụng ứng dụng và tự động đồng bộ dữ liệu khi có kết nối lại.
- Firebase cung cấp SDK tích hợp cho nhiều nền tảng (Web, Android, iOS), giúp việc tích hợp vào ứng dụng chat trở nên dễ dàng.
- Cả Realtime Database và Firestore đều cung cấp các quy tắc bảo mật để kiểm soát quyền truy cập dữ liệu, đảm bảo an toàn cho dữ liệu người dùng.

**b) Nhược điểm của realtimedatabase và firestore:**

- Cả hai dịch vụ đều yêu cầu phụ thuộc vào hạ tầng của Google. Điều này có thể dẫn đến khó khăn nếu muốn chuyển đổi sang các giải pháp khác và gây lo ngại về vấn đề bảo mật và quyền riêng tư dữ liệu.
- Khi ứng dụng mở rộng và sử dụng nhiều tài nguyên, chi phí cho cả Realtime Database và Firestore có thể tăng cao, làm cho việc duy trì ứng dụng trở nên đắt đỏ.
- Cả hai cơ sở dữ liệu đều có thể gặp phải vấn đề về phức tạp trong thiết kế và quản lý dữ liệu khi ứng dụng phát triển lớn hơn. Đối với các truy

vấn phức tạp và dữ liệu lớn, việc thiết kế cấu trúc dữ liệu đúng cách là rất quan trọng để đảm bảo hiệu suất.

## CHƯƠNG 2: THIẾT KẾ HỆ THỐNG

### **2.1 Các chức năng của ứng dụng:**

### **2.1.1 Đăng ký:**

- Khi người dùng mới đăng kí, thì cái thông tin cơ bản như: tên đăng nhập, mật khẩu, email, số điện thoại..... sẽ được lưu trên firestore, giúp hỗ trợ các lần đăng nhập lần sau, cũng như khôi phục mật khẩu bị quên.

### **2.1.2 Đăng nhập.**

- Khi người dùng đã đăng kí, phải nhập đúng tên tài khoản, mật khẩu và captcha để có thể truy vấn vào cơ sở dữ liệu, có quyền truy cập vào form main menu.

### **2.1.3 Xác thực hai lớp.**

- Dùng để tăng cường bảo mật hạn chế truy cập trái phép vào tài khoản giúp tăng tính bảo vệ thông tin dữ liệu của người dùng.

### **2.1.4 Quản lí tài khoản, tên hiển thị.**

- Hệ thống sẽ cho mình giữ nguyên tên ban đầu mình đăng kí và cũng có cho phép người dùng có khả năng đổi tên hiển thị của mình trên ứng dụng. Người dùng có thể tạo, xóa các phòng chat.

### **2.1.5 Quản lí phòng chat.**

- Chỉ có người tạo ra phòng chat (admin) mới có quyền xóa room phòng.

- Ở mỗi form sau khi tạo hoặc tìm kiếm các phòng chat sẽ được lưu lại cho lần đăng nhập lại lần sau sẽ hiện thị mà không cần phải tìm kiếm lại

### **2.1.6 Quản lí tin nhắn, file.**

- Sau khi vào được phòng chat nhập key mã hóa, khi cập nhật khóa thì các tin nhắn lưu sẽ được mã hóa dưới dạng Thuật toán SHA-256

- Các tin nhắn và file sẽ được lưu lên realtimedatabase và các người dùng khác sẽ truy cập vào và chuyển tiếp được tin nhắn

- Chỉ có người dùng tạo ra phòng chat (hay admin) thì mới có quyền xóa phòng, xóa toàn bộ tin nhắn trong đoạn chat và kiểm tra đã có bao nhiêu tên người dùng đã nhắn tin vào phòng chat.

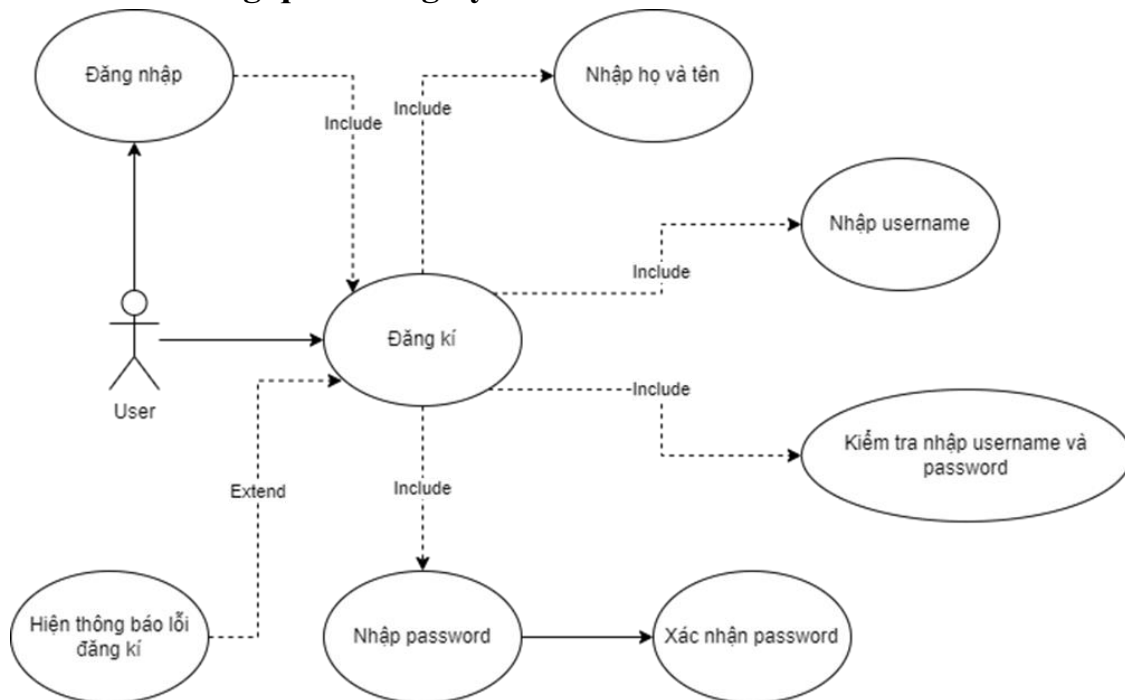
### **2.1.7 Mã hóa file, tin nhắn.**

- Có thể dùng để mã hóa file, tin nhắn, icon.... và gửi cho người khác và dùng lại form mã hóa để giải mã các file, tin nhắn, icon.... để có thể xem.

### 2.1.8 Góp ý.

- Nếu có ý kiến hay lỗi cần báo cáo sẽ điền vào form, các ý kiến sẽ được gửi đến mail của quản trị viên hệ thống (admin). Sau đó sẽ được kiểm tra và giải quyết các nhu cầu và lỗi cần sửa chữa.

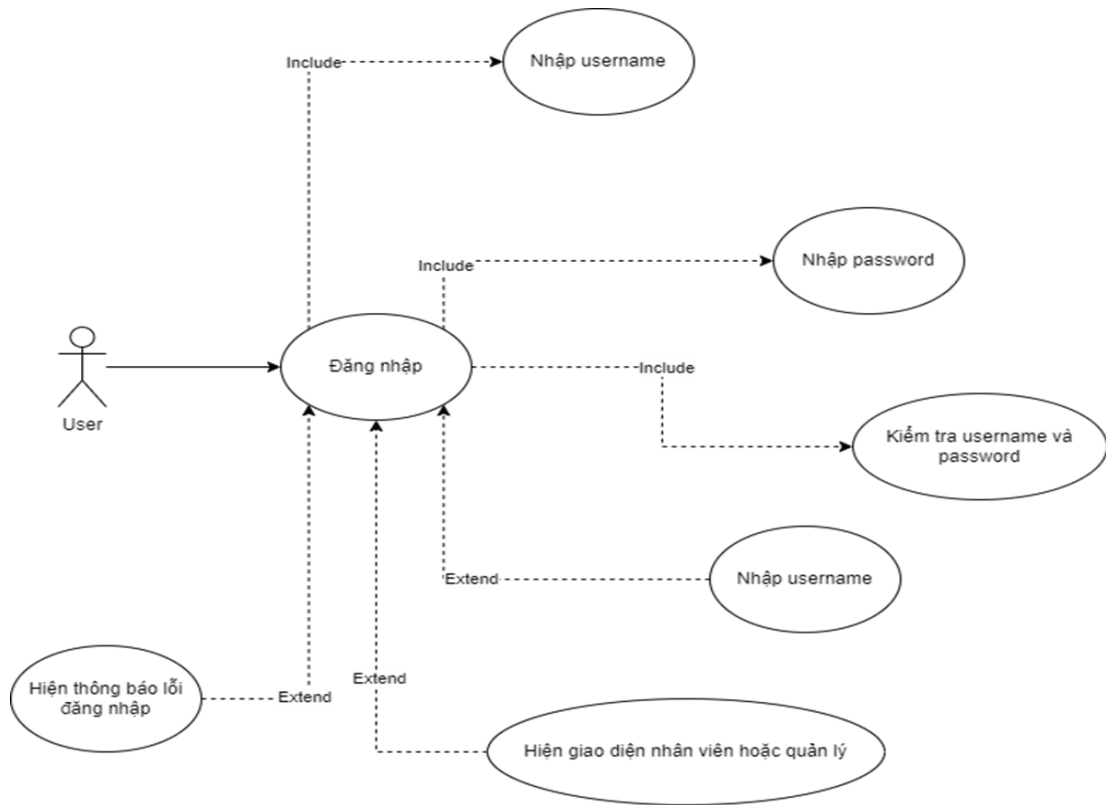
## 2.2 Mô hình tổng quan đăng ký:



Mô tả	Cho phép người dùng tạo một tài khoản mới
-------	---

Điều kiện	<ul style="list-style-type: none"> <li>- Tài khoản mà người dùng muốn tạo chưa tồn tại trong cơ sở dữ liệu.</li> <li>- Người dùng phải nhập đúng định dạng các thông tin cá nhân.</li> </ul>
Hậu điều kiện	Người dùng đăng ký thành công
Sự kiện chính	<ul style="list-style-type: none"> <li>- Dùng được ứng dụng chỉ khi người dùng muốn một tài khoản mới.</li> <li>- Hệ thống yêu cầu người dùng nhập họ tên, username và password, các thông tin cá nhân khác.</li> <li>- Người dùng phải nhập lại password để xác nhận.</li> <li>- Trong quá trình nhập password, người dùng có thể chọn hiển thị mật khẩu để tránh các sai sót.</li> </ul>
Sự kiện phụ	Nếu trong sự kiện chính người dùng nhập sai định dạng các thông tin cá nhân hoặc tài khoản vừa khởi tạo đã tồn tại trong cơ sở dữ liệu thì hệ thống sẽ báo lỗi. Sau đó, người dùng quay về đầu dòng sự kiện hoặc không thực hiện đăng ký.
Yêu cầu đặc biệt	Sẽ có gửi OTP trong 1 khoảng thời gian cần phải nhập trong khoảng thời gian đó.

### 2.3 Mô hình tổng quan đăng nhập:



Mô tả	Cho phép người dùng truy cập vào ứng dụng
-------	---

Điều kiện	<ul style="list-style-type: none"> <li>- Tài khoản người dùng phải tồn tại trước đó.</li> <li>- Phải nhập đúng định dạng, chính xác mật khẩu đã tạo trước đó</li> </ul>
Sự kiện chính	<ul style="list-style-type: none"> <li>- Hệ thống yêu cầu người dùng nhập username và password đã được cấp trước đó.</li> <li>- Người dùng nhập username, password.</li> <li>- Hệ thống kiểm tra thông tin và cho người dùng đăng nhập vào hệ thống..</li> </ul>
Sự kiện phụ	Nếu trong sự kiện chính người dùng nhập sai username hoặc password thì hệ thống sẽ báo lỗi. Sau đó, người dùng quay về đầu dòng sự kiện hoặc không thực hiện đăng nhập.
Yêu cầu đặc biệt	Nhập đúng captcha, đăng nhập trong 1 khoảng thời gian nhất định mà ứng dụng quy định, nếu mà quá thời gian sẽ bị thoát ra khỏi ứng dụng.

## 2.4 Sử dụng cơ sở dữ liệu:

### Firestore (Lưu Trữ Thông Tin Người Dùng)

Bảng UserData:

- Mỗi tài liệu trong bảng này đại diện cho một người dùng.
- Các trường thông tin có thể bao gồm: username, email, password (nếu có), avatar, status (trạng thái hiện tại của người dùng),...

Quy Tắc Bảo Mật:

- Thiết lập quy tắc bảo mật để chỉ cho người dùng cụ thể có thể truy cập vào tài liệu của họ. Lưu mật khẩu dưới dạng mã hóa, có biến kiểm tra xem có đăng nhập ở nơi khác chưa.

### Realtime Database (Lưu Trữ Tin Nhắn)

Danh Sách Tin Nhắn (Node "messages"):

- Mỗi nút con trong node này đại diện cho một cuộc trò chuyện hoặc một nhóm.

- Mỗi tin nhắn được lưu trữ dưới dạng một nút con với các thuộc tính như sendername, timestamp,...
- Sử dụng các node con bổ sung để lưu trữ thông tin về hình ảnh hoặc tệp đính kèm của tin nhắn.

### Quy Tắc Bảo Mật:

- Thiết lập quy tắc bảo mật để đảm bảo rằng chỉ những người dùng liên quan có thể đọc và ghi vào các tin nhắn.

### Tích Hợp

- Khi người dùng gửi tin nhắn mới, ứng dụng sẽ thêm tin nhắn vào Realtime Database và cập nhật giao diện người dùng để hiển thị tin nhắn mới.
- Khi người dùng đăng nhập, ứng dụng sẽ truy vấn Firestore để lấy thông tin người dùng và hiển thị trong giao diện.

### Lợi Ích

- Tối Ưu Hóa Hiệu Suất: Firestore được sử dụng để lưu trữ thông tin người dùng vì tính linh hoạt và hiệu suất tốt trong việc truy xuất dữ liệu người dùng. Realtime Database được sử dụng cho các tin nhắn vì tính chất thời gian thực và khả năng đồng bộ nhanh chóng.
- Quản Lý Dữ Liệu Hiệu Quả: Sử dụng mỗi loại cơ sở dữ liệu cho mục đích cụ thể giúp tổ chức dữ liệu một cách có tổ chức và dễ quản lý.
- Tùy Chỉnh Bảo Mật: Thiết lập quy tắc bảo mật riêng biệt cho mỗi loại dữ liệu để đảm bảo tính bảo mật và quyền riêng tư.

(default)	UserData	Bánh Khá
+ Start collection	+ Add document	+ Start collection
UserData >	Bánh Khá >	+ Add field
	dai test test2 tunanh Đại Bùi Ph...	DisplayName: "Bánh Khá" Email: "khabanhpro135@gmail.com" Gender: "Unknown" Password: "x5oUhhPWjwDjihEPLIMU" Phone: "Unknown" Username: "Bánh Khá" isLoggedIn: false



Hình 1.1 Cơ sở dữ liệu lưu trữ UserData.

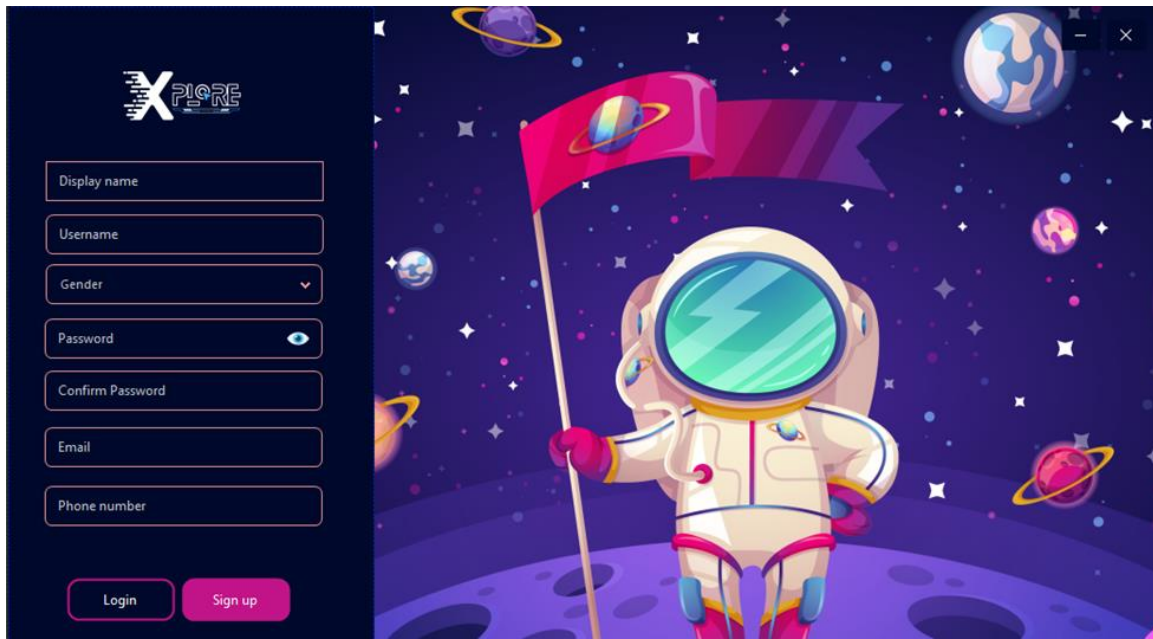


Hình 1.2 Cơ sở dữ liệu lưu trữ phòng chat, tin nhắn và file

## CHƯƠNG 3: XÂY DỰNG HỆ THỐNG ỨNG DỤNG CHAT

### 3.1 Các chức năng hỗ trợ:

#### 3.1.1 Chức năng đăng kí, đăng nhập.



Hình 2. Giao diện đăng ký của app

- Khi người dùng mới muốn tạo 1 tài khoản mới, hãy điền toàn bộ thông tin bao gồm tên hiển thị, tên đăng nhập, mật khẩu, các thông tin cá nhân khác.

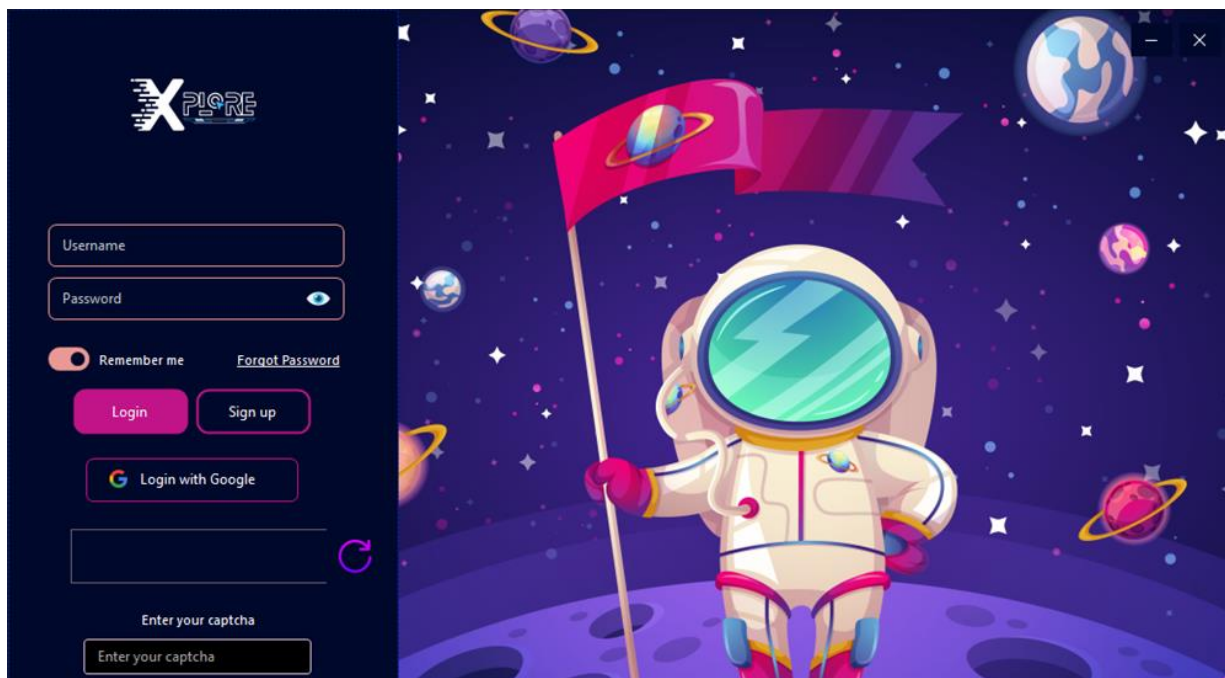
```

DisplayName: "tunanhcheck"
Email: "22520071@gm.uit.edu.vn"
Gender: "Male"
Key: null
Password: "bn9LPG3j6OyrETalsjOvjA=="
Phone: "1234567890"
Username: "tunanh"
isLoggedIn: false

```

- Sau tạo thành công tất cả sẽ được lưu lên firestore, và đặc biệt mật khẩu được lưu dưới dạng mã hóa.

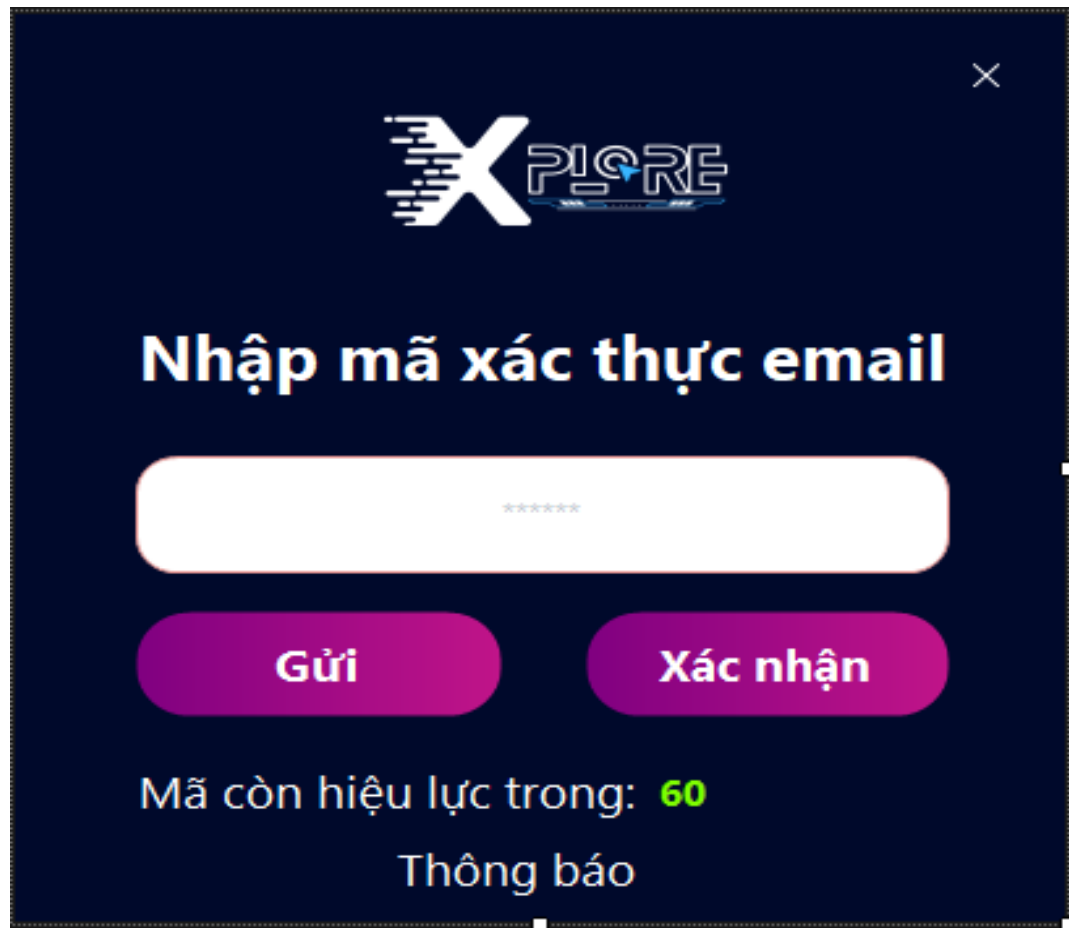
- Đặc biệt có biến isLoggedIn sẽ trả về biến true nếu mà tài khoản đã đăng nhập ở một nơi khác và không thể đăng nhập tiếp, nếu thoát ra thì biến isLoggedIn trả về false sau đó có thể tiếp tục đăng nhập.



*Hình 3. Giao diện đăng nhập của app*

- Khi đã có tài khoản chỉ cần đăng nhập lại đúng tên Username, mật khẩu đã tạo trước đó.

#### 3.1.1.1. Các tính năng hỗ trợ:



✕

**XPIRE**

**Nhập mã xác thực email**

\*\*\*\*\*

**Gửi**

**Xác nhận**

Mã còn hiệu lực trong: **60**

Thông báo

Hình 4. Tính năng xác thực bằng email

-Hỗ trợ chức năng xác thực email, khi không nhận được sẽ click vào button Gửi mã sẽ được gửi đến email đã đăng ký trước đó, nếu không nhập mã trong thời gian cố định sẽ bị thoát. Sau đó click Xác nhận.

**Nhập mail đã đăng ký**

**Gửi mã**

**Nhập mã phục hồi**

**Xác nhận**

**60**

**Không thể truy cập Mail?**

**Quay lại đăng nhập**

*Hình 5. Tính năng khôi phục lại mật khẩu bằng email.*

-Hỗ trợ khôi phục tài khoản nếu người dùng quên.

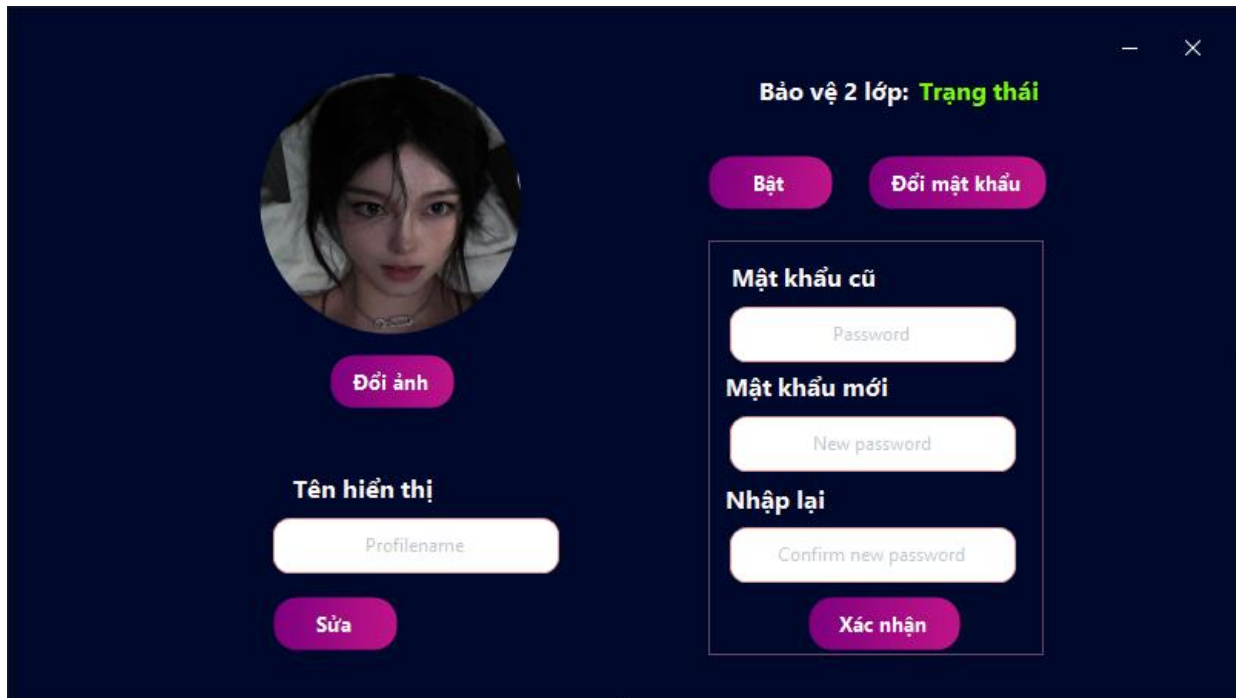
### 3.1.2. Giao diện chính



*Hình 6. Giao diện MainMenu của app*

- Khi đăng nhập tài khoản thành công sẽ hiện ra form MainMenu. Khi click vào các button sẽ hiện ra các chức năng tương ứng

#### 3.1.2.1. Giao diện trang chủ



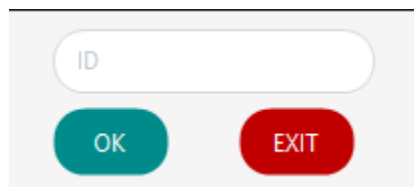
Hình 7. Giao diện trang chủ của app

- Khi chuyển vào click vào button Trang chủ sẽ chuyển đến giao diện của trang chủ:

- + Có khả năng đổi được tên hiển thị của User
- + Chức năng bảo vệ hai lớp và đổi mật khẩu

### 3.1.3. Chức năng chat.

#### 3.1.3.1. Xác nhận tên User tham gia:



Hình 8. Giao diện xác nhận tên user tham gia Chat

- Khi click vào button Nhấn tin nhóm sẽ hiển thị ra đầu tiên, có chức năng xác nhận User đã đăng kí trước đó được truy xuất từ firestore.

#### 3.1.3.2. Tạo các room chat:

Hình 9 . Giao diện hiện thị tìm, tạo, xóa các nhóm

- Ở form tạo phòng, khi :

- + Tạo phòng, chỉ cần nhập tên nhóm và mật khẩu click tạo nhóm, nhóm vừa tạo sẽ được lưu lên realtimedatabase

- + Khi tìm phòng đã có sẵn, cần phải nhập đúng tên phòng và mật khẩu, nếu sai thì sẽ hiện thông báo không tìm được phòng

- + Xóa phòng, chỉ có các user đã tạo ra phòng (admin) mới có quyền xóa phòng, các user khác không phải người đã tạo ra phòng chỉ xóa được sự xuất hiện của phòng ở panel1( Phải nhập đúng tên phòng và mật khẩu.)

### 3.1.3.3. Room có quyền Admin:





Hình 10 . Giao diện chat của user admin

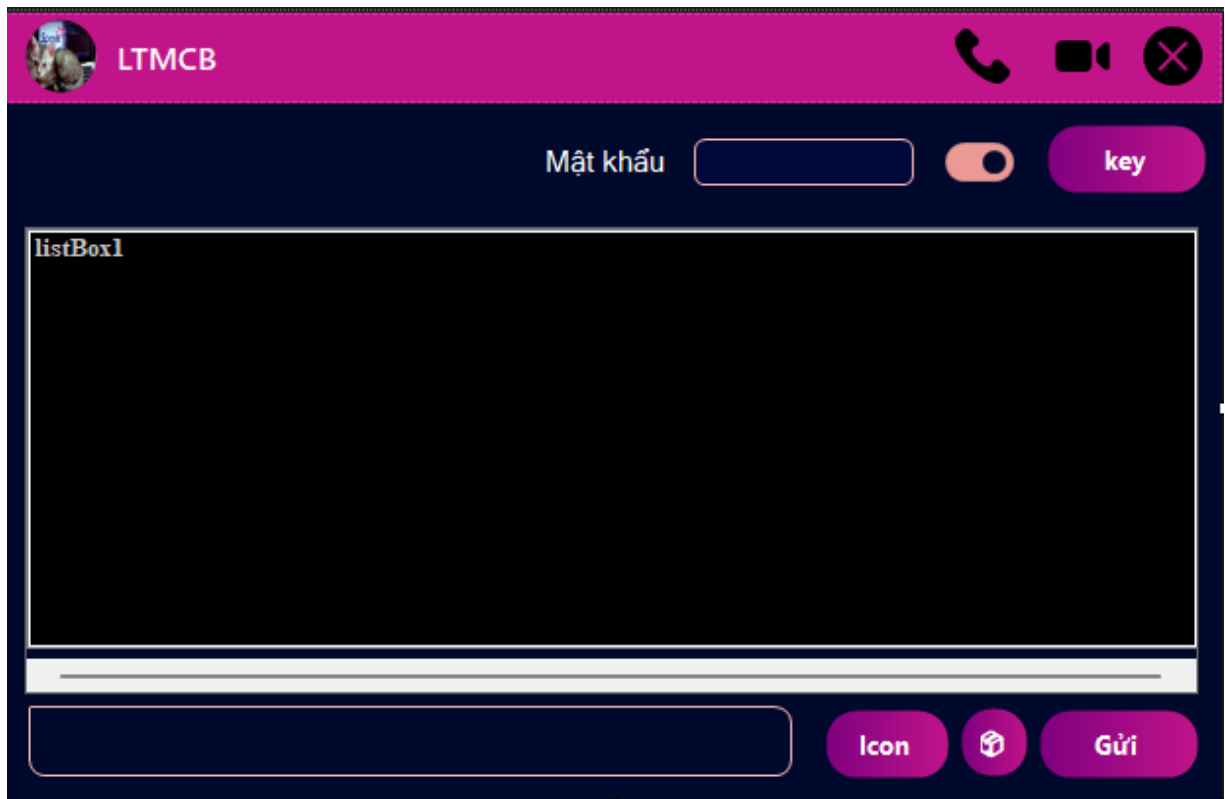
- Ở form Admin:

+Sau khi nhập key click vào button key, khóa sẽ được cập nhật và tin nhắn hiển thị ở listBox1 dưới dạng mã hóa.

+Có hiển thị tên user đã nhắn tin vào nhóm ở datagridview

+Chỉ có admin mới có quyền xóa toàn bộ lịch sử đoạn chat và cả hiển thị đoạn chat trên listBox1

#### 3.1.3.4. Room có quyền Member:



Hình 11 . Giao diện chat của user member

- Ở form Member:

+ Nhập đúng key thì tin nhắn mã hóa sẽ được giải mã và hiển thị trên listBox1

+ Ở hai form Admin và Member đều có khả năng gửi được icon và file, cùng với mã hóa được chúng.

#### 3.1.4. Chức năng mã hóa file.

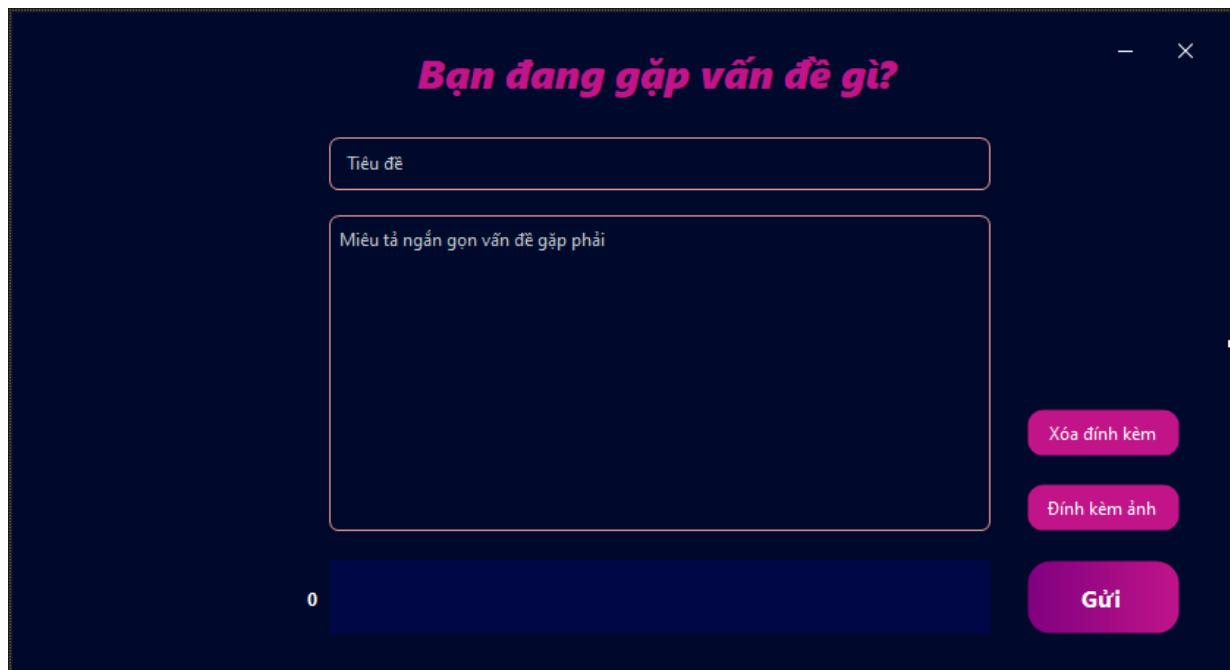


Hình 12 . Giao diện mã hóa file

- Form Mã hóa có khả năng mã hóa được các file mà không xem được chúng, chỉ khi giải mã lại thì mới có khả năng xem được.

=> Giúp tạo sự an toàn khi gửi file tránh để người khác muốn xem file không cần thiết.

### 3.1.5.Chức năng góp ý.



**Bạn đang gặp vấn đề gì?**

Tiêu đề

Miêu tả ngắn gọn vấn đề gặp phải

0

Xóa đính kèm

Đính kèm ảnh

Gửi

Hình 13 . Giao diện đóng góp ý kiến

- Có chức năng nhận các tin nhắn góp ý, các lỗi cần khắc phục và sẽ được gửi tới email của quản trị viên. Sau đó quản trị viên sẽ kiểm tra và khắc phục các lỗi sớm nhất.

## CHƯƠNG 4: THỰC NGHIỆM ĐỒ ÁN

Link github:

<https://github.com/KelvinThangg/X-Plore.git>

Link demo video:

<https://drive.google.com/drive/folders/1qAra7uwSDFNFncSEgLWVIg1iY-4d0dvv>

## CHƯƠNG 5: KẾT QUẢ VÀ HƯỚNG PHÁT TRIỂN

### 5.1 Kết quả:

- Vận dụng được các kiến thức của môn Lập trình mạng căn bản, biết sử dụng firebase, ứng dụng được kiến thức của môn Cơ sở dữ liệu.
- Học hỏi biết thêm nhiều kinh nghiệm làm việc nhóm quản lí thời gian, tập được kĩ năng tự học và nghiên cứu.
- Cải thiện được kĩ năng lập trình.

### 5.2 Hạn chế và khó khăn:

- Ứng dụng còn thiếu nhiều tính năng, khá khó để sử dụng rộng rãi.
- Còn nhiều chỗ thiếu logic, chưa kịp khắc phục.
- Vẫn còn nhiều chỗ dùng không được tiện dụng phù hợp với người dùng.
- Giao diện còn đơn giản.

### 5.3 Hướng phát triển:

- Xây dựng lại giao diện bắt mắt hơn phù hợp với người dùng hiện tại.
- Lấy thông tin hơi để phát triển rộng rãi cho nhiều đối tượng khác hàng.
- Bổ sung là nhiều phần dùng còn chưa tiện dụng.
- Chỉnh sửa một số lỗi logic.
- Tối ưu code và cơ sở dữ liệu để có dễ dàng sửa chữa, nâng cấp.

***TÀI LIỆU THAM KHẢO:***

- [https://vi.wikipedia.org/wiki/C\\_Sharp\\_\(ng%C3%B4n\\_ng%E1%BB%AF\\_1%E1%BA%ADp\\_tr%C3%ACnh\)](https://vi.wikipedia.org/wiki/C_Sharp_(ng%C3%B4n_ng%E1%BB%AF_1%E1%BA%ADp_tr%C3%ACnh))
- <https://www.imic.edu.vn/tin-tuc-cong-nghe/30518/uu-nhuoc-diem-cua-c-net-nen-hoc-lap-trinh-c-o-dau-thi-tot-nhat-ha-noi-.html>
- <https://lanit.com.vn/c-sharp-la-gi-ung-dung-uu-diem.html>
- <https://fr.slideshare.net/slideshow/n-tt-nghiep-v-tm-hiu-ngn-ng-lp-trnh-c-sharp-v-vit-ng-dng-chat-trong-mng-landoc/264690303>

## PHỤ LỤC