

**ỦY BAN NHÂN DÂN
THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC SÀI GÒN**



**BÁO CÁO TIỂU LUẬN
HỌC PHẦN: ĐỒ ÁN CHUYÊN NGÀNH
TÊN ĐỀ TÀI: HỆ THỐNG NHẬN DIỆN KHUÔN MẶT TRÊN
HỆ ĐIỀU HÀNH WINDOW**

Thông tin thành viên:

3121411171	Vũ Bình Phước	DCT121C3
3121411117	Nguyễn Văn Kiệt	DCT121C3
3121411154	Phạm Thanh Nhân	DCT121C3

Giảng viên: PGS.TS Phạm Thế Bảo

Thành phố Hồ Chí Minh, ngày 21 tháng 04 năm 2025

MỤC LỤC

MỞ ĐẦU	4
CHƯƠNG 1: TỔNG QUAN VỀ CREDENTIAL PROVIDER ..	5
CHƯƠNG 2: QUY TRÌNH HỆ THỐNG	6
1. Khởi động hệ thống	6
2. Kích hoạt ứng dụng nhận diện khuôn mặt	6
3. Thu thập dữ liệu khuôn mặt	6
4. So sánh dữ liệu	7
5. Xác thực người dùng	7
6. Xử lý lỗi và thông báo	7
CHƯƠNG 3: CÀI ĐẶT VÀ TRIỂN KHAI	8
1. Cài đặt	8
2. Triển khai	15
CHƯƠNG 4: KẾT LUẬN	16
1. Hoàn thiện chức năng hệ thống	16
2. Tính tự động và tiện lợi	16

DANH MỤC BẢNG BIỂU, HÌNH ẢNH

Hình ảnh:

Hình 1: Cài đặt Visual Studio với các công cụ phát triển C++	8
Hình 2: Các tệp để cấu hình Credential Provider	9
Hình 3: Khởi tạo hàm CreateNamePipeServer()	9
Hình 4: Khởi tạo hàm OnFaceIDClick()	10
Hình 5: Khởi tạo hàm GetSerialization()	10
Hình 6: Biên dịch thành file DLL	11
Hình 7: Triển khai pipe_server để lắng nghe yêu cầu từ Credential Provider	12
Hình 8: Triển khai faceid_serive để thiết lập hoạt động của các tiến trình và dịch vụ chạy song song với hệ thống.	12
Hình 9: Giao diện Lock Screen sau khi cài đặt một credential provider mới	13
Hình 10: Trạng thái quét Face ID thất bại	14
Hình 11: Trạng thái quét Face ID thành công	14

TÓM TẮT ĐỀ TÀI

Đề tài “Xây dựng hệ thống đăng nhập bằng nhận diện khuôn mặt” hướng đến việc triển khai một giải pháp xác thực người dùng hiện đại và an toàn thông qua công nghệ nhận diện khuôn mặt. Hệ thống được phát triển bằng ngôn ngữ Python, sử dụng các thư viện như OpenCV, Dlib, face_recognition để xử lý và nhận dạng khuôn mặt theo thời gian thực. Ngoài ra, hệ thống còn được tích hợp vào hệ điều hành Windows thông qua Credential Provider, cho phép người dùng đăng nhập máy tính bằng Face ID thay cho mật khẩu truyền thống.

MỞ ĐẦU

Trong bối cảnh công nghệ số phát triển mạnh mẽ, nhu cầu đảm bảo an toàn và bảo mật trong việc truy cập hệ thống ngày càng tăng cao. Các hình thức đăng nhập truyền thống như sử dụng mật khẩu đang dần trở nên lỗi thời, dễ bị đánh cắp, quên lãng hoặc sử dụng không an toàn. Điều này đặt ra yêu cầu về các phương thức xác thực mới – trong đó nhận diện khuôn mặt là một giải pháp nổi bật với nhiều ưu điểm: không cần tiếp xúc, nhanh chóng, và có độ chính xác cao.

Xuất phát từ thực tiễn đó, đề tài “Xây dựng hệ thống đăng nhập bằng nhận diện khuôn mặt” được lựa chọn nhằm ứng dụng các kiến thức về xử lý ảnh, học máy và lập trình Python để tạo nên một hệ thống có tính ứng dụng thực tế. Đề tài tập trung vào việc thu thập, xử lý, lưu trữ và so sánh dữ liệu khuôn mặt, kết hợp giao diện đồ sẵn có của Windows Login. Đặc biệt, hệ thống được tích hợp trực tiếp vào cơ chế đăng nhập của hệ điều hành Windows thông qua việc xây dựng một Credential Provider tùy chỉnh, cho phép người dùng đăng nhập bằng khuôn mặt thay cho mật khẩu.

Trong quá trình thực hiện, nhóm gặp không ít khó khăn như tối ưu hóa tốc độ nhận diện để đảm bảo thời gian đăng nhập nhanh chóng, xử lý ảnh trong điều kiện ánh sáng yếu, và vận hành chương trình ở ngoài màn hình đăng nhập. Tuy nhiên, nhờ tinh thần tự học và kiên trì, các vấn đề dần được giải quyết, giúp hoàn thiện hệ thống một cách hiệu quả và tối ưu.

CHƯƠNG 1: TỔNG QUAN VỀ CREDENTIAL PROVIDER

Credential Provider là một thành phần quan trọng trong kiến trúc xác thực người dùng của hệ điều hành Windows kể từ phiên bản Windows Vista trở về sau. Nó đóng vai trò là giao diện trung gian giữa người dùng và hệ thống đăng nhập (LogonUI), cho phép thu thập thông tin xác thực (credential) từ người dùng và chuyển tiếp chúng tới hệ thống để xử lý xác thực.

Mỗi Credential Provider có thể định nghĩa một giao diện đăng nhập riêng biệt như: nhập mật khẩu, quét vân tay, nhận diện khuôn mặt hoặc sử dụng mã PIN. Windows mặc định cung cấp sẵn một số Credential Provider cơ bản, nhưng cũng cho phép các nhà phát triển tạo mới hoặc mở rộng Credential Provider để hỗ trợ các phương thức xác thực tùy chỉnh.

Credential Provider hoạt động theo cơ chế COM (Component Object Model) với các interface tiêu chuẩn như:

- ✧ ICredentialProvider,
- ✧ ICredentialProviderCredential

Cho phép Windows lấy và xử lý thông tin đăng nhập từ các nguồn khác nhau. Khi người dùng khởi động máy tính hoặc mở khóa thiết bị, LogonUI sẽ gọi các Credential Provider đã đăng ký để thu thập credential và xác thực thông tin với hệ thống bảo mật nội bộ (LSASS - Local Security Authority Subsystem Service).

Việc xây dựng một Credential Provider tùy chỉnh cho phép tích hợp những phương pháp xác thực mới như nhận diện khuôn mặt, tăng cường mức độ bảo mật và mang đến trải nghiệm đăng nhập hiện đại hơn cho người dùng Windows.

CHƯƠNG 2: QUY TRÌNH HỆ THỐNG

Hệ thống nhận diện khuôn mặt được thiết kế theo quy trình gồm nhiều bước xử lý từ khi người dùng khởi động ứng dụng đến khi xác thực được danh tính. Quá trình này bảo đảm sự tích hợp giữa giao diện người dùng và hệ thống xử lý nhận diện.

1. Khởi động hệ thống

Khi người dùng bật máy tính hoặc truy cập màn hình đăng nhập, hệ điều hành Windows sẽ khởi động giao diện LogonUI. Tại đây, Credential Provider tùy chỉnh được đăng ký sẽ được gọi để hiển thị phương thức đăng nhập bằng nhận diện khuôn mặt bên cạnh các phương thức truyền thống.

2. Kích hoạt ứng dụng nhận diện khuôn mặt

Credential Provider sẽ kích hoạt một ứng dụng trung gian (viết bằng Python) để thực hiện quá trình nhận diện khuôn mặt. Thông qua giao tiếp IPC (Inter-Process Communication) như socket hoặc named pipe, Credential Provider truyền yêu cầu và nhận kết quả xác thực từ ứng dụng Python.

3. Thu thập dữ liệu khuôn mặt

Ứng dụng Python sử dụng webcam để thu thập hình ảnh khuôn mặt hiện tại của người dùng. Các thư viện xử lý ảnh như OpenCV và Dlib được sử dụng để phát hiện khuôn mặt trong khung hình, đồng thời trích xuất các đặc trưng sinh trắc học (face encoding).

4. So sánh dữ liệu

Hệ thống sẽ lấy mã hóa khuôn mặt (face encoding) từ ảnh thu thập được và so sánh với các mẫu dữ liệu khuôn mặt đã lưu trong cơ sở dữ liệu. Quá trình so sánh sử dụng thuật toán đo khoảng cách Euclidean giữa các vector đặc trưng để xác định độ tương đồng.

5. Xác thực người dùng

Nếu tìm thấy một mẫu dữ liệu phù hợp với ngưỡng độ chính xác định sẵn, hệ thống xác nhận thành công danh tính người dùng. Kết quả xác thực được trả về Credential Provider, từ đó gửi thông tin credential hợp lệ cho hệ thống Windows để hoàn tất quá trình đăng nhập.

6. Xử lý lỗi và thông báo

Trong trường hợp khuôn mặt không được nhận diện hoặc xảy ra lỗi trong quá trình xử lý, hệ thống sẽ hiển thị thông báo lỗi và yêu cầu người dùng thử lại hoặc sử dụng phương thức đăng nhập khác.

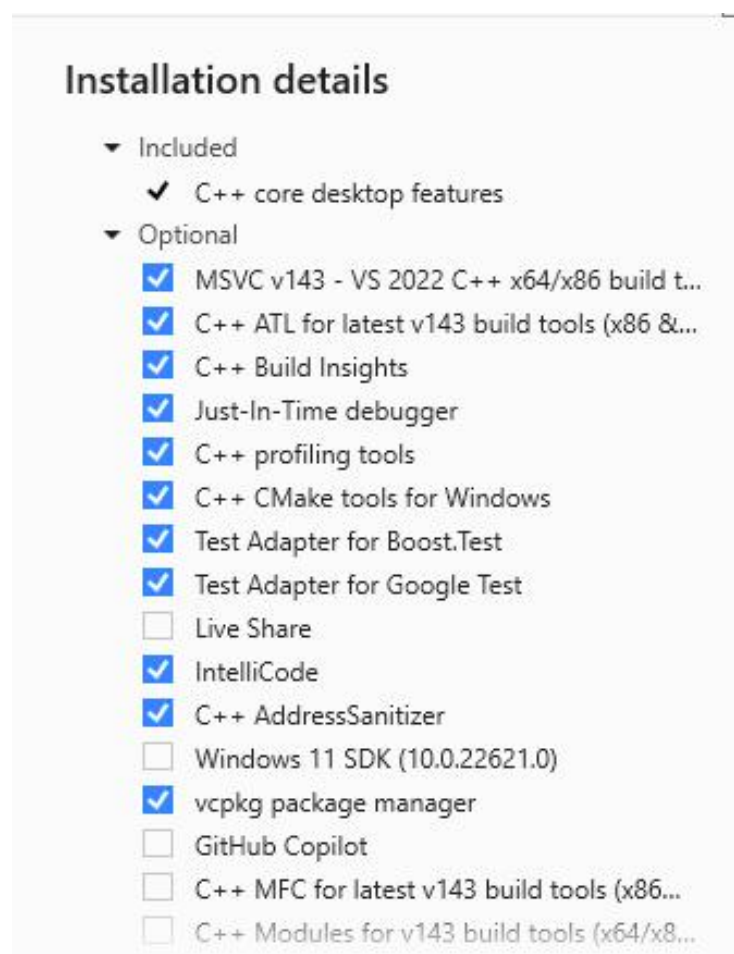
CHƯƠNG 3: CÀI ĐẶT VÀ TRIỂN KHAI

Trong phương án này, hệ thống nhận diện khuôn mặt được tích hợp trực tiếp vào quá trình đăng nhập Windows thông qua việc phát triển một Credential Provider tùy chỉnh.

1. Cài đặt

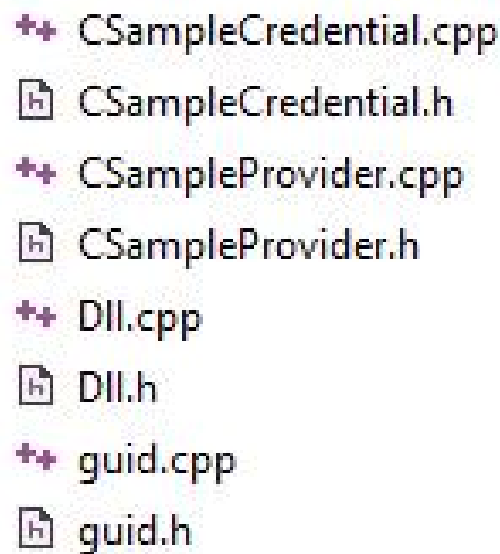
Các bước cài đặt chi tiết:

Bước 1: Cài đặt Visual Studio với đầy đủ công cụ phát triển C++ để biên dịch Credential Provider.



Hình 1: Cài đặt Visual Studio với các công cụ phát triển C++

Bước 2: Tạo các tệp để cấu hình Credential Provider.



- ++ CSampleCredential.cpp
- h CSampleCredential.h
- ++ CSampleProvider.cpp
- h CSampleProvider.h
- ++ Dll.cpp
- h Dll.h
- ++ guid.cpp
- h guid.h

Hình 2: Các tệp để cấu hình Credential Provider

Bước 3: Xây dựng các phương thức cốt lõi cho quá trình xử lý

Trong quá trình phát triển hệ thống, sẽ cần triển khai nhiều phương thức khác nhau. Tuy nhiên, ở bước này, chúng ta sẽ tập trung vào việc xây dựng và phân tích các phương thức quan trọng nhất, có vai trò then chốt trong việc điều phối và thực thi chức năng của hệ thống:



```
HRESULT CSampleCredential::CreateNamedPipeServer() { ... }
```

Hình 3: Khởi tạo hàm CreateNamePipeServer()

Mục đích: Thiết lập một kênh giao tiếp liên tiến trình (Inter-Process Communication – IPC) thông qua Named Pipe, đóng vai trò như server để giao tiếp với một tiến trình khác – chẳng hạn như một script Python xử lý nhận diện khuôn mặt.

```
HRESULT CSampleCredential::OnFaceIDClick()
{ ...
}
```

Hình 4: Khởi tạo hàm OnFaceIDClick()

Hàm CSampleCredential::OnFaceIDClick() đóng vai trò là cầu nối giữa giao diện đăng nhập người dùng (Credential Provider trong Windows) và hệ thống xác thực khuôn mặt được xây dựng bằng Python. Đây là một phần quan trọng trong quá trình tích hợp tính năng đăng nhập bằng Face ID vào hệ điều hành Windows thông qua giao diện người dùng tùy chỉnh và nhận kết quả xác thực để quyết định hiển thị trạng thái và có cho phép đăng nhập hay không trong Windows Credential Provider.

```
HRESULT CSampleCredential::GetSerialization(_Out_ CREDENTIAL_PROVIDER_GET_SERIALIZATION_RESPONSE* pcpgsr,
    _Out_ CREDENTIAL_PROVIDER_CREDENTIAL_SERIALIZATION* pcpcs,
    _Outptr_result_maybenull_ PWSTR* ppwszOptionalStatusText,
    _Out_ CREDENTIAL_PROVIDER_STATUS_ICON* pcpsiOptionalStatusIcon)
{ ...
}
```

Hình 5: Khởi tạo hàm GetSerialization()

Hàm GetSerialization() là một trong những hàm quan trọng nhất trong Credential Provider. Mục đích của hàm này là cung cấp thông tin xác thực người dùng (credential data) cho Windows Logon để tiến hành quá trình đăng nhập tự động nếu điều kiện cho phép. Trả về thông tin đăng nhập dưới dạng CREDENTIAL_PROVIDER_CREDENTIAL_SERIALIZATION.

Khi đã xác thực thành công (ví dụ sau khi xác thực bằng Face ID), hàm này sẽ được gọi để:

- ✧ Cung cấp thông tin xác thực cho hệ thống.
- ✧ Cho phép Windows đăng nhập người dùng mà không cần nhập mật khẩu bằng tay.
- ✧ Tránh hiển thị lại giao diện đăng nhập.

Bước 4: Biên dịch Credential Provider thành file DLL và copy vào đường dẫn C:\Windows\System32.



Hình 6: Biên dịch thành file DLL

Bước 5: Đăng ký Credential Provider vào Windows bằng cách thêm thông tin vào cấu hình 1 file register.reg(Credential Providers).

Mở Tools → Create GUID → copy định dạng {XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX}.

Tạo File register.reg với nội dung:

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Authentication\Credential Providers\{XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX}]
```

```
@="SampleV2CredentialProvider"
```

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\CLSID\{XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX}]
```

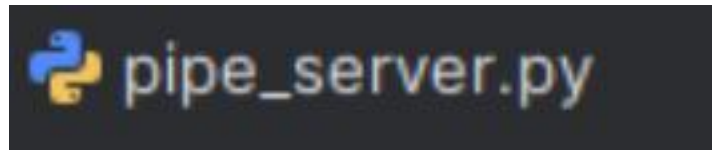
```
@="SampleV2CredentialProvider"
```

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\CLSID\XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX]\InprocServer32]
```

```
@="SampleV2CredentialProvider.dll"
```

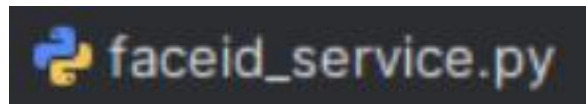
```
"ThreadingModel"="Apartment"
```

Bước 6: Triển khai pipe_server nhận diện khuôn mặt Python, đảm bảo ứng dụng tự động lắng nghe yêu cầu từ Credential Provider.



Hình 7: Triển khai pipe_server để lắng nghe yêu cầu từ Credential Provider

Bước 7: Triển khai faceid_serive để thiết lập hoạt động của các tiến trình và dịch vụ chạy song song với hệ thống.



Hình 8: Triển khai faceid_serive để thiết lập hoạt động của các tiến trình và dịch vụ chạy song song với hệ thống.

Nếu mở Command Prompt với quyền Admin, chạy lệnh sau để cài đặt dịch vụ:

```
python "C:\FaceID_Project\Source code control\faceid_service.py" install
```

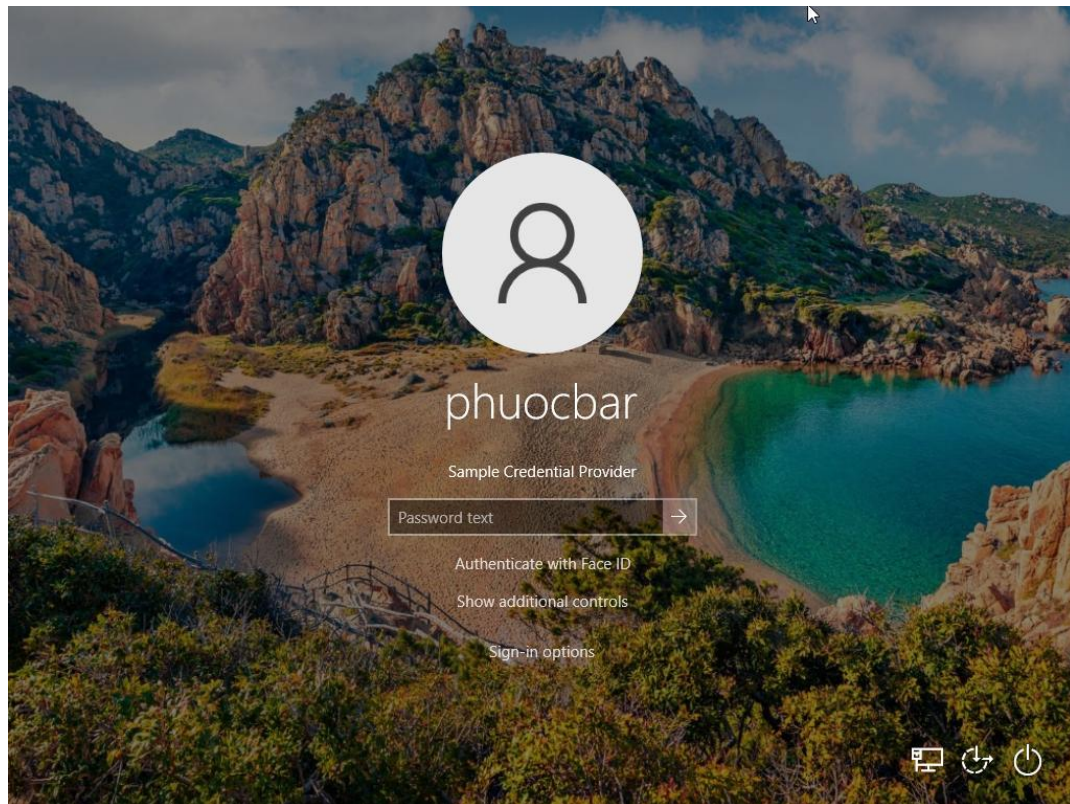
Chạy lệnh:

```
C:\FaceID_Project\Source code control>python faceid_service.py start
```

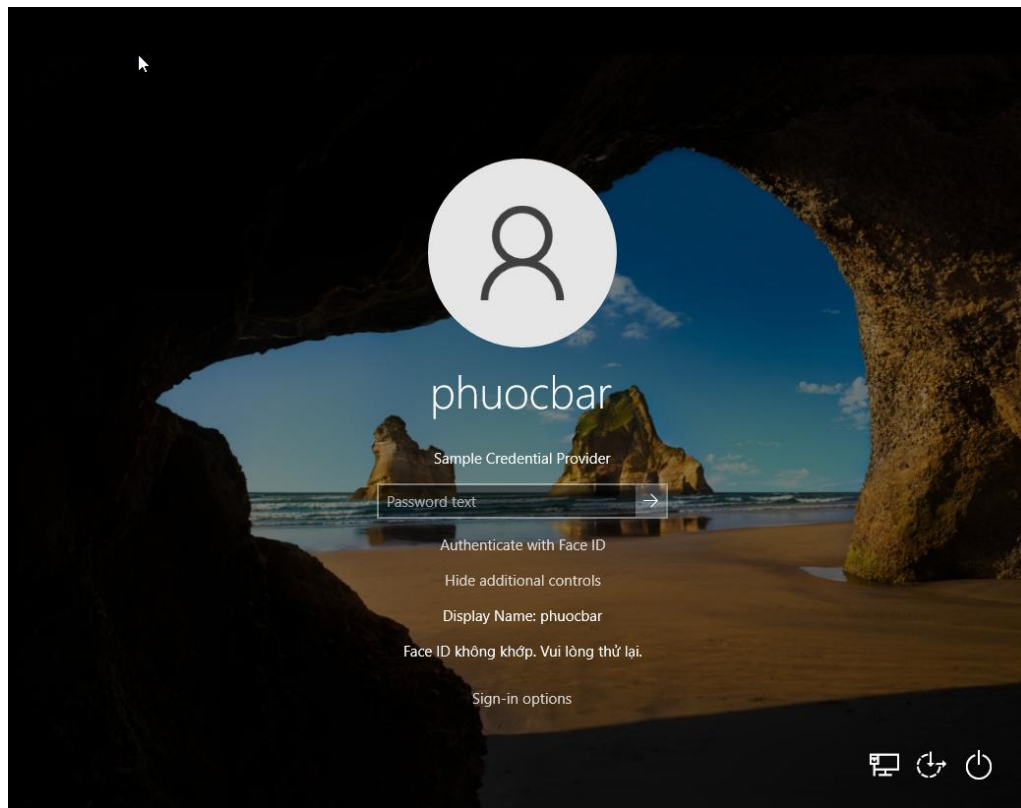
để khởi động service

Bước 8: Thử nghiệm đăng nhập: bật máy tính, chọn phương thức đăng nhập bằng khuôn mặt, hệ thống tự động kích hoạt ứng dụng và tiến hành xác thực.

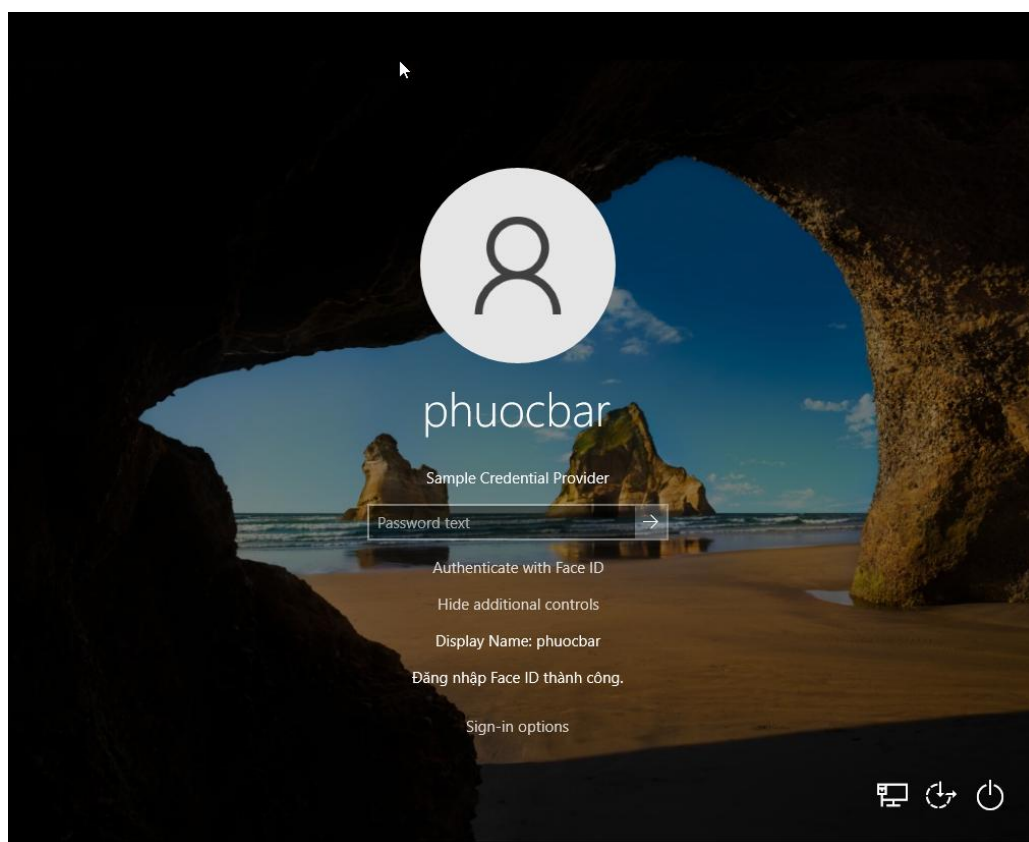
Các màn hình giao diện:



Hình 9: Giao diện Lock Screen sau khi cài đặt một credential provider mới



Hình 10: Trạng thái quét Face ID thất bại



Hình 11: Trạng thái quét Face ID thành công

2. Triển khai

Sau khi cài đặt xong, người dùng có thể triển khai hệ thống nhận diện khuôn mặt trên máy tính của mình. Điều này bao gồm việc thử nghiệm ứng dụng và đảm bảo các bước hoạt động chính xác.

Kiểm tra trước khi triển khai ứng dụng:

- Kiểm tra camera: Trước khi ứng dụng hoạt động, người dùng cần kiểm tra camera để đảm bảo hệ thống có thể nhận diện khuôn mặt qua webcam.
- Đăng ký khuôn mặt: Người dùng cần đăng ký khuôn mặt của mình bằng cách nhập tên người dùng và quét khuôn mặt. Sau khi quét thành công, thông tin khuôn mặt sẽ được lưu trữ trong cơ sở dữ liệu SQL Server.
- Đăng nhập: Khi người dùng quay lại hệ thống, họ có thể nhập tên người dùng và quét khuôn mặt để xác thực. Nếu nhận diện thành công, họ sẽ được phép truy cập vào hệ thống.
- Bảo mật và tinh chỉnh: Để nâng cao độ chính xác và bảo mật, người dùng có thể điều chỉnh các tham số trong hệ thống nhận diện khuôn mặt, chẳng hạn như tăng cường xử lý hình ảnh hoặc tối ưu hóa thuật toán nhận diện.

CHƯƠNG 4: KẾT LUẬN

Sau quá trình thiết kế, cài đặt và triển khai hệ thống nhận diện khuôn mặt, hệ thống đã đạt được một số kết quả tích cực, đáp ứng được các mục tiêu ban đầu đã đề ra. Các kết quả có thể được phân tích theo các khía cạnh sau:

1. Hoàn thiện chức năng hệ thống

Nhận diện khuôn mặt chính xác: Hệ thống sử dụng thư viện `face_recognition` kết hợp với `OpenCV` để trích xuất và so sánh đặc trưng khuôn mặt. Qua quá trình thử nghiệm, tỉ lệ nhận diện chính xác trong điều kiện ánh sáng tốt và camera rõ nét là đạt trên 90%.

Đăng ký và xác thực: Người dùng có thể dễ dàng đăng ký khuôn mặt mới với tên người dùng tương ứng. Khi xác thực, hệ thống so sánh vector khuôn mặt mới với dữ liệu trong cơ sở dữ liệu và đưa ra kết quả tức thì.

Giao diện thân thiện: Ứng dụng có giao diện đơn giản với hộp thoại nhập tên người dùng, dễ sử dụng cho cả người không rành về công nghệ.

2. Tính tự động và tiện lợi

Khởi động cùng hệ thống: Thông qua `Task Scheduler`, hệ thống tự động khởi chạy khi người dùng mở máy tính hoặc sau khi sleep, giúp quá trình xác thực diễn ra ngay khi khởi động thiết bị.

