



# BẢO MẬT VÀ TỐI ƯU ỨNG DỤNG WEB.NET

# BẢO MẬT VÀ TỐI ƯU ỨNG DỤNG WEB .NET



## GIỚI THIỆU KHOÁ HỌC



## CÁC NỘI DUNG CHÍNH

1. Mục tiêu khoá học
2. Nội dung khoá học
3. Phương pháp học tập



### 1. MỤC TIÊU KHOÁ HỌC

- ✓ Học viên nắm rõ được kỹ thuật bảo mật ứng dụng web trên nền tảng .Net.
- ✓ Học viên có thể sử dụng Identity trong xác thực và phân quyền người dùng với ứng dụng Web.Net.
- ✓ Học viên có thể tích hợp xác thực người dùng của bên thứ 3, xác thực 2 bước vào ứng dụng web .Net.



## 1. MỤC TIÊU KHOÁ HỌC

- ✓ Học viên nắm được các kỹ thuật tối ưu ứng dụng web .Net, sử dụng các ORM để kết nối và làm việc với cơ sở dữ liệu.
- ✓ Học viên nắm được quy trình thiết kế kiến trúc phần mềm theo mô hình Micro service.



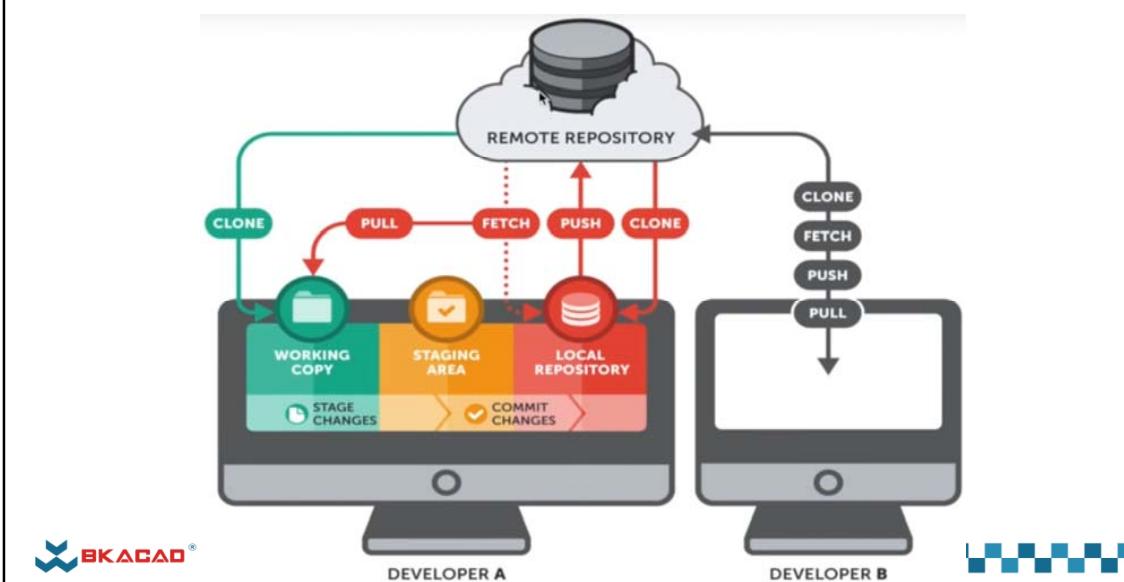
## 1. MỤC TIÊU KHOÁ HỌC

Mục tiêu: Sử dụng git và github quản lý mã nguồn dự án



# 1. MỤC TIÊU KHOÁ HỌC

Mục tiêu: Sử dụng git và github quản lý mã nguồn dự án



# 2. NỘI DUNG KHOÁ HỌC

3: Xác thực và phân quyền người dùng trong .net

2: Sử dụng git và github để quản lý mã nguồn dự án

1: Cài đặt và thiết lập môi trường lập trình

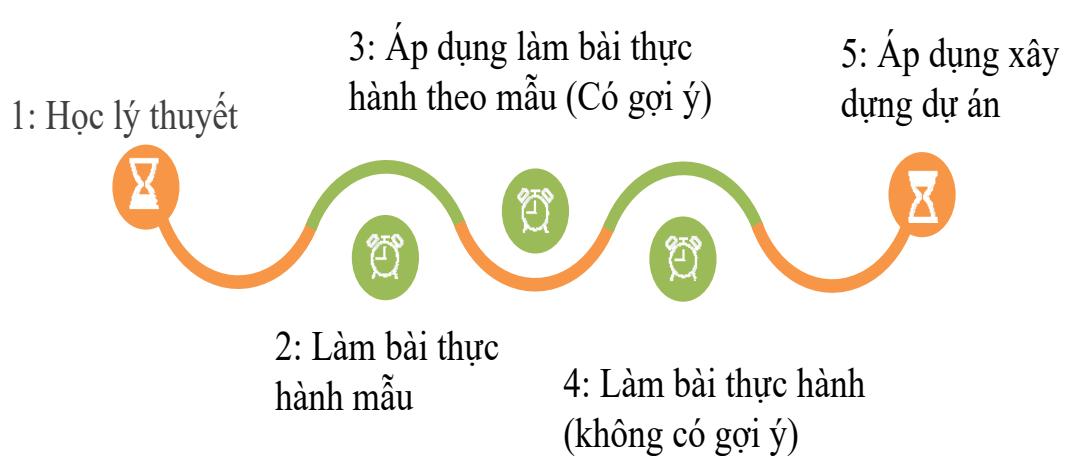
4: Đăng nhập hệ thống với ứng dụng của bên thứ 3

5: Tối ưu dung web .Net

6: Kiến trúc Microservice trong .net



### 3. PHƯƠNG PHÁP HỌC TẬP





# BẢO MẬT VÀ TỐI ƯU ÚNG DỤNG WEB .NET



# CÀI ĐẶT CÔNG CỤ VÀ THIẾT LẬP MÔI TRƯỜNG LẬP TRÌNH



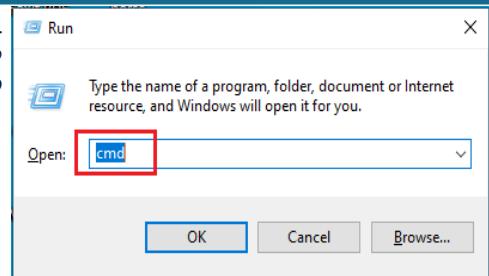
## Các nội dung chính

- I. Hướng dẫn sử dụng cmd/terminal
- II. Cài đặt .net core
- III. Cài đặt IDE/Code editor
- IV. Cài đặt sql server
- V. Cài đặt git
- VI. Đăng ký và cấu hình tài khoản Github
- VII. Cài đặt NodeJS

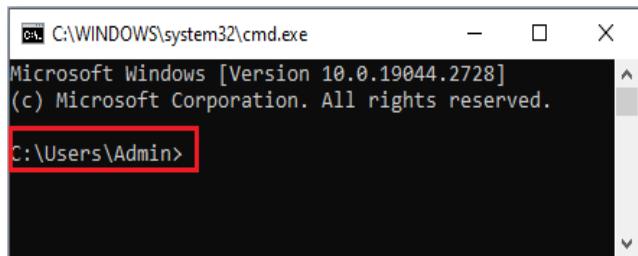


## I. HƯỚNG DẪN SỬ DỤNG CMD/TERMINAL

Đối với Windows, người dùng bấm tổ hợp phím “Windows + R” sau đó nhập “cmd”



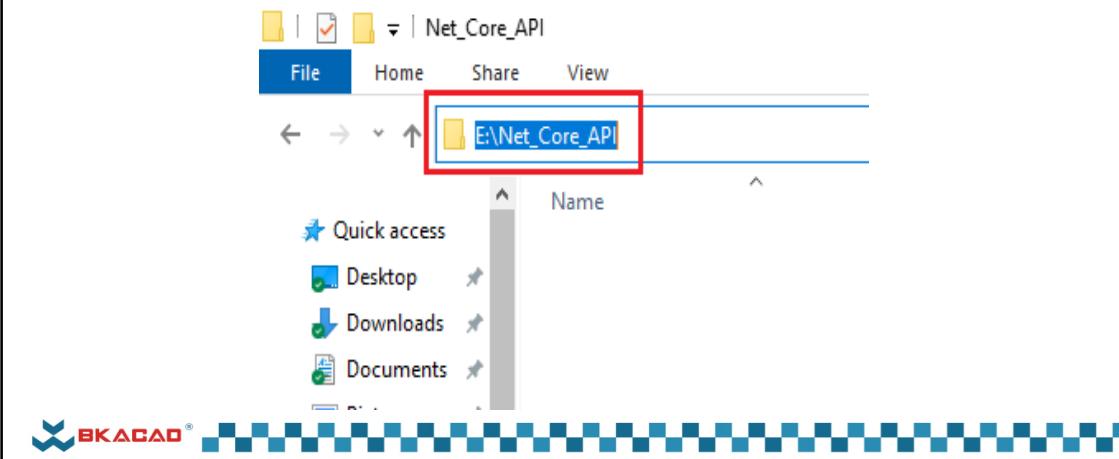
Sau đó nhấn phím “Enter”, ứng dụng CMD sẽ được mở ra:



## I. HƯỚNG DẪN SỬ DỤNG CMD/TERMINAL

Để chuyển tới địa chỉ thư mục muốn làm việc, người dùng thực hiện các bước sau:

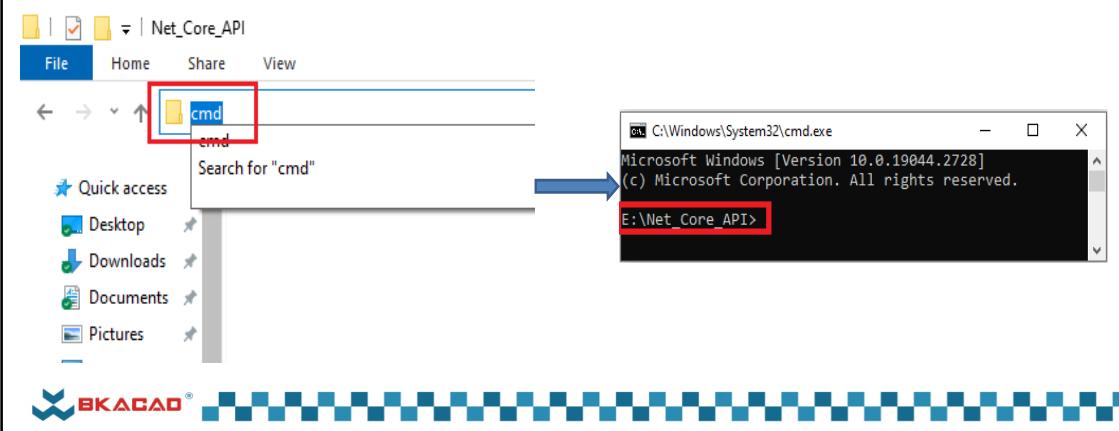
- Mở thư mục cần chuyển đến



## I. HƯỚNG DẪN SỬ DỤNG CMD/TERMINAL

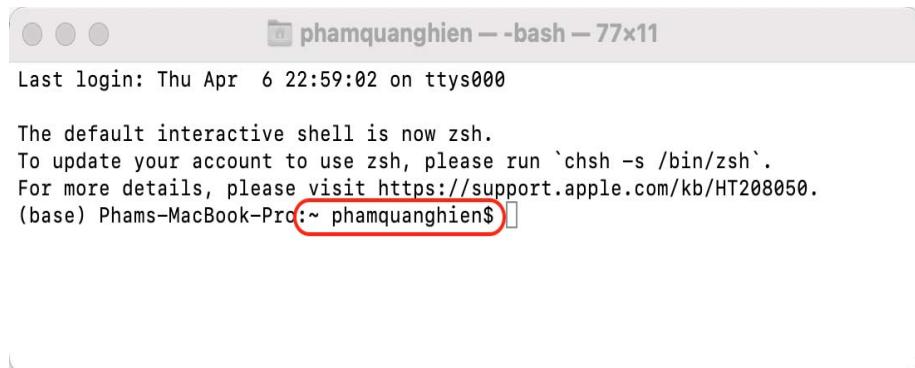
Để chuyển tới địa chỉ thư mục muốn làm việc, người dùng thực hiện các bước sau:

- Nhập “cmd” và bấm “Enter”



## I. HƯỚNG DẪN SỬ DỤNG CMD/TERMINAL

Đối với Linux (sử dụng tổ hợp phím Ctrl+Alt+T) hoặc MacOS (Spotlight Search) để mở “Terminal”:



```
phamquanghien -- bash -- 77x11
Last login: Thu Apr  6 22:59:02 on ttys000
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(base) Phams-MacBook-Pro:~ phamquanghien$
```



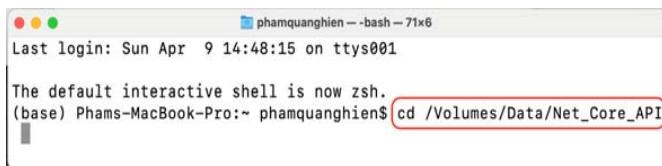
## I. HƯỚNG DẪN SỬ DỤNG CMD/TERMINAL

1.Mở Terminal

2.Nhập lệnh “cd Folder”

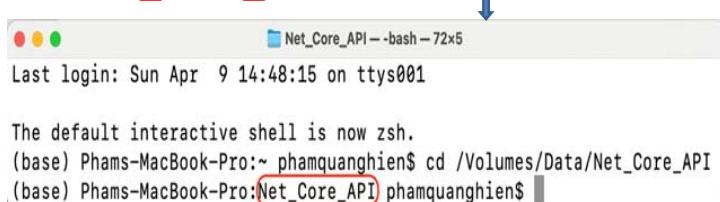
3.Folder là đường dẫn tới  
thư mục gốc sẽ làm  
việc.

4.Ví dụ: cd /Volumes/Data/Net\_Core\_API



```
phamquanghien -- bash -- 71x6
Last login: Sun Apr  9 14:48:15 on ttys001
The default interactive shell is now zsh.
(base) Phams-MacBook-Pro:~ phamquanghien$ cd /Volumes/Data/Net_Core_API
```

Nhập lệnh  
“cd Folder”  
và nhấn Enter



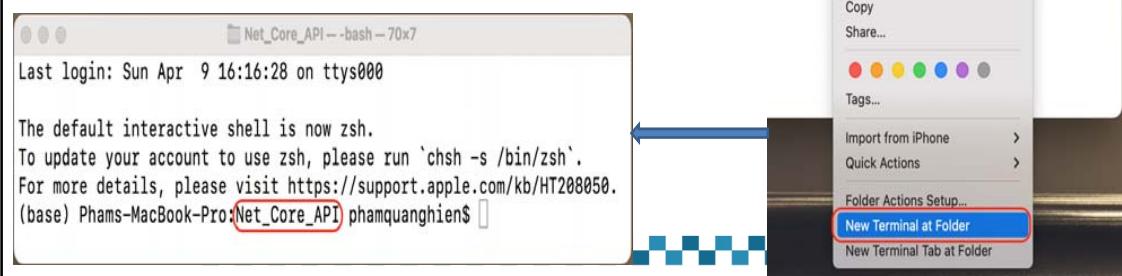
```
Net_Core_API -- bash -- 72x5
Last login: Sun Apr  9 14:48:15 on ttys001
The default interactive shell is now zsh.
(base) Phams-MacBook-Pro:~ phamquanghien$ cd /Volumes/Data/Net_Core_API
(base) Phams-MacBook-Pro:Net_Core_API phamquanghien$
```



## I. HƯỚNG DẪN SỬ DỤNG CMD/TERMINAL

Thay vì dùng lệnh “cd Folder” người dùng làm như sau:

- Chọn thư mục muốn làm việc.
- Kích phải chuột vào thư mục và chọn “New Terminal at Folder” hoặc “New Terminal Tab at Folder” như hình bên



## II. CÀI ĐẶT MÔI TRƯỜNG .NET

1. Truy cập vào google và tìm kiếm với từ khoá “Download net core”

microsoft.com  
https://dotnet.microsoft.com/en-us/d... · Dịch trang này

Download .NET (Linux, macOS, and Windows) - Microsoft .NET

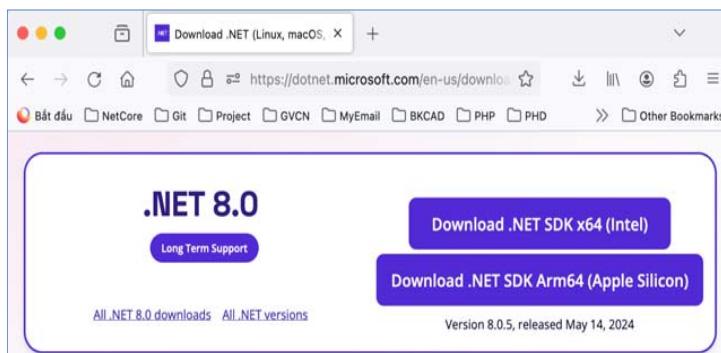
Free downloads for building and running .NET apps on Linux, macOS, and Windows. ... NET Core and cloud-native development, ARM64 support, and more.

NET versions · Download .NET 7.0 · .NET 6.0

## II. CÀI ĐẶT MÔI TRƯỜNG .NET

2. Truy cập vào link đầu tiên (<https://dotnet.microsoft.com/en-us/download>)

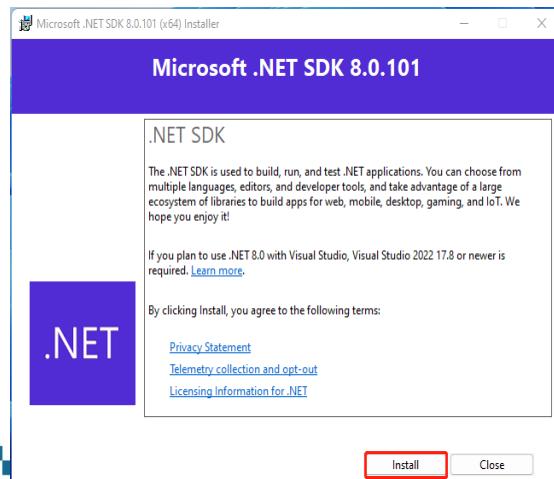
3. Chọn “.NET SDK x64” để tải về .Net SDK x64 bản 8.0



## II. CÀI ĐẶT MÔI TRƯỜNG .NET

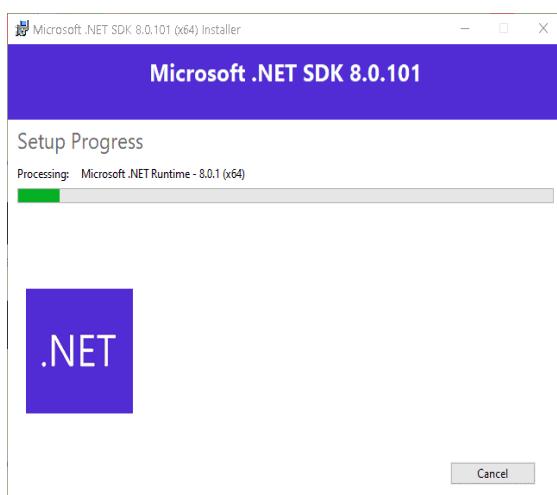
4. Sau khi tải file .Net SDK x64 thành công, mở file tải về để tiến hành cài đặt.

5. Ở cửa sổ cài đặt chọn “Install”



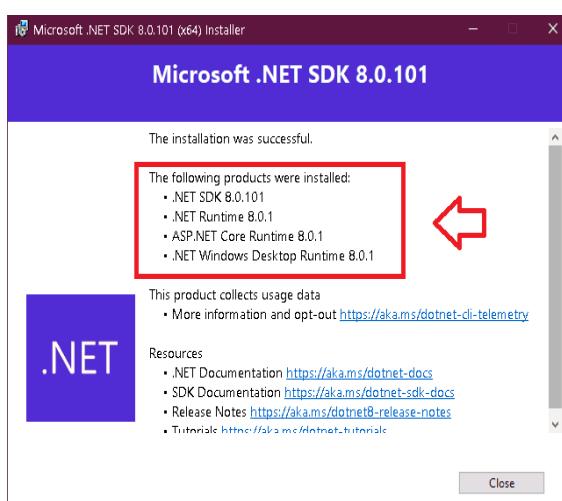
## II. CÀI ĐẶT MÔI TRƯỜNG .NET

6. Sau khi chọn “**Install**”, quá trình cài đặt được diễn ra



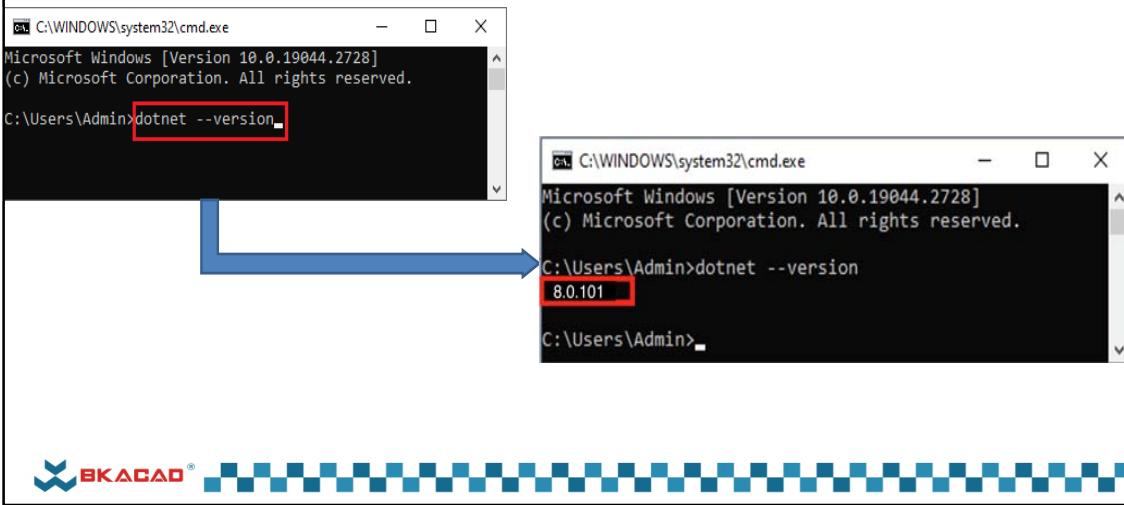
## II. CÀI ĐẶT MÔI TRƯỜNG .NET

7. Quá trình cài đặt diễn ra thành công, chọn “**Close**” để kết thúc



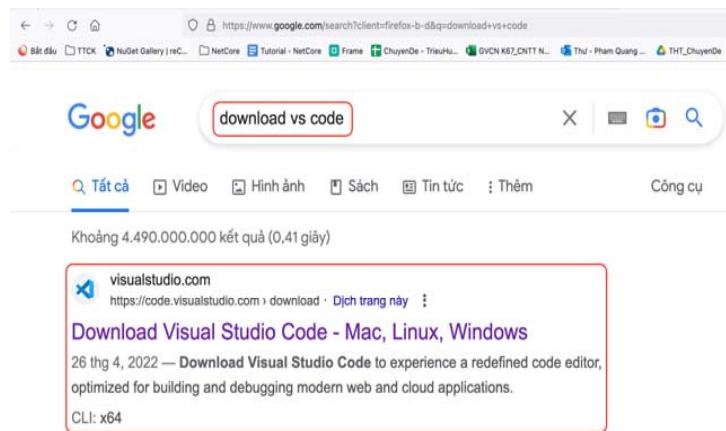
## II. CÀI ĐẶT MÔI TRƯỜNG .NET

8. Mở **CMD/Terminal**, chạy lệnh “**dotnet --version**” để kiểm tra phiên bản .Net đã được cài đặt thành công



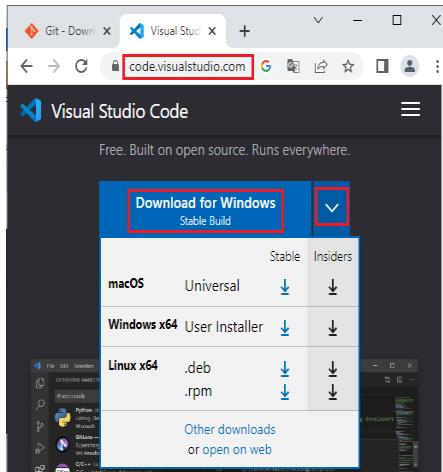
## III. CÀI ĐẶT VISUAL STUDIO CODE

1. Truy cập vào google và tìm kiếm với từ khoá “**Download VS code**”



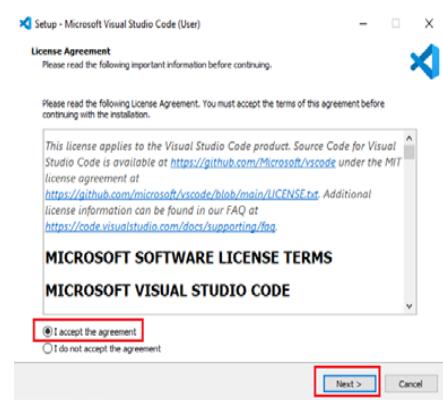
### III. CÀI ĐẶT VISUAL STUDIO CODE

2. Truy cập vào link <https://code.visualstudio.com/download>
3. Chọn phiên bản phù hợp với hệ điều hành để tải về



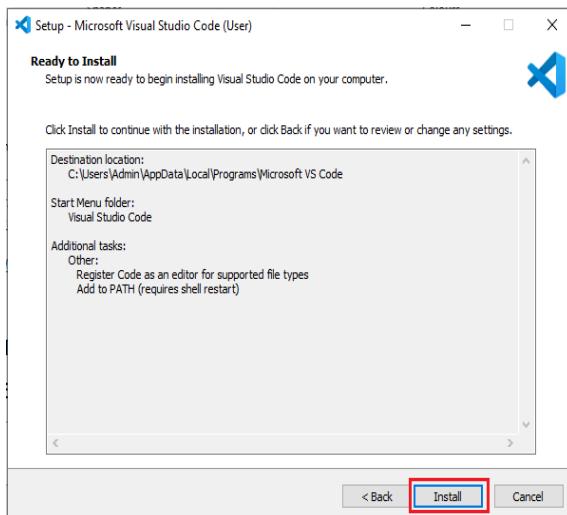
### III. CÀI ĐẶT VISUAL STUDIO CODE

4. Sau khi tải về file cài đặt (**VSCODEUserSetup.exe**), chạy file exe để tiến hành cài đặt, Ở cửa sổ cài đặt chọn “I accept the agreement” → chọn “Next”



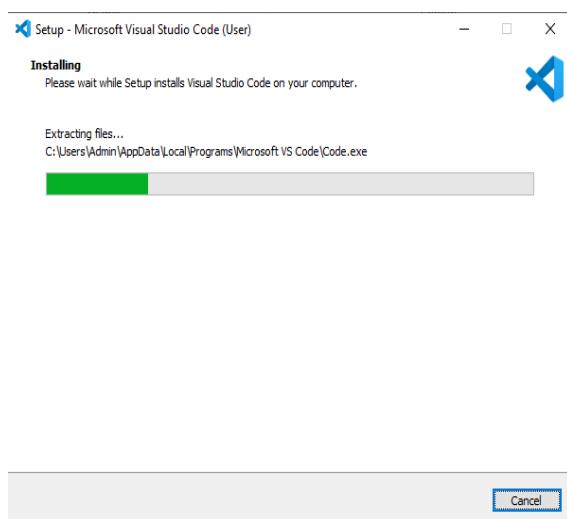
### III. CÀI ĐẶT VISUAL STUDIO CODE

#### 5. Cửa sổ tiếp theo hiện ra chọn “Install”



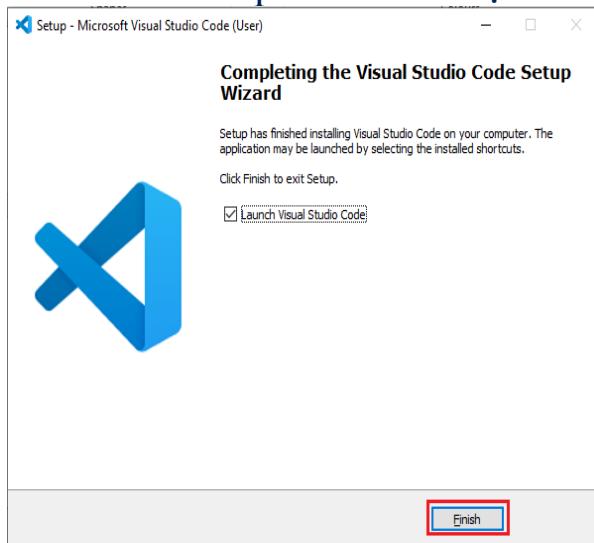
### III. CÀI ĐẶT VISUAL STUDIO CODE

#### 6. Quá trình cài đặt được diễn ra



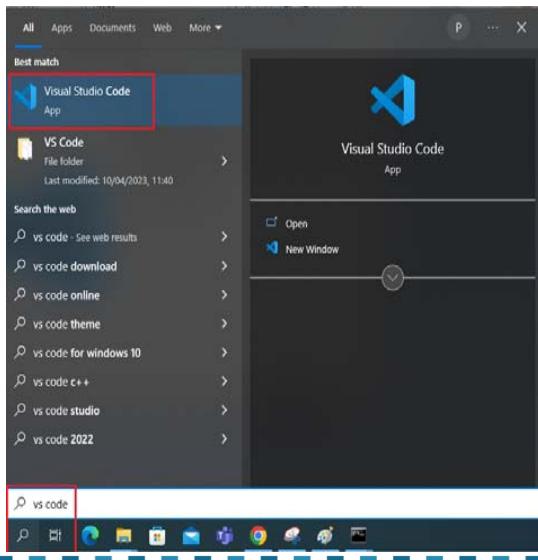
### III. CÀI ĐẶT VISUAL STUDIO CODE

#### 7. Chọn “Finish” để kết thúc quá trình cài đặt



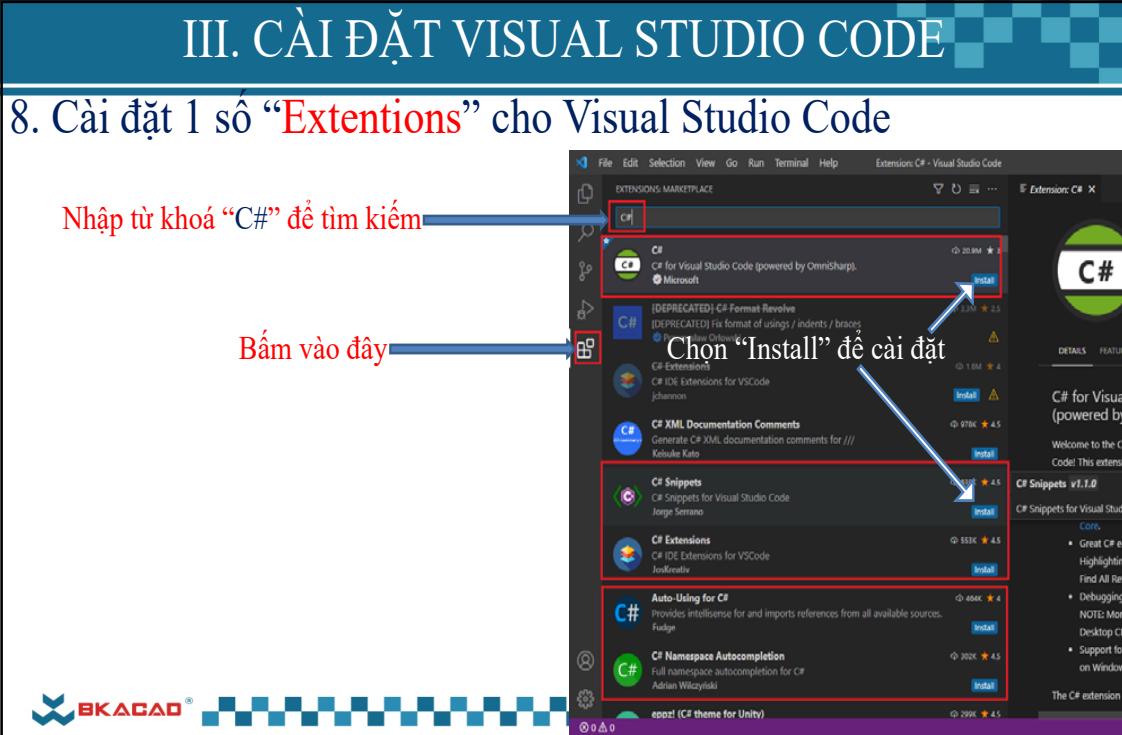
### III. CÀI ĐẶT VISUAL STUDIO CODE

#### 8. Mở ứng dụng “Visual Studio Code”



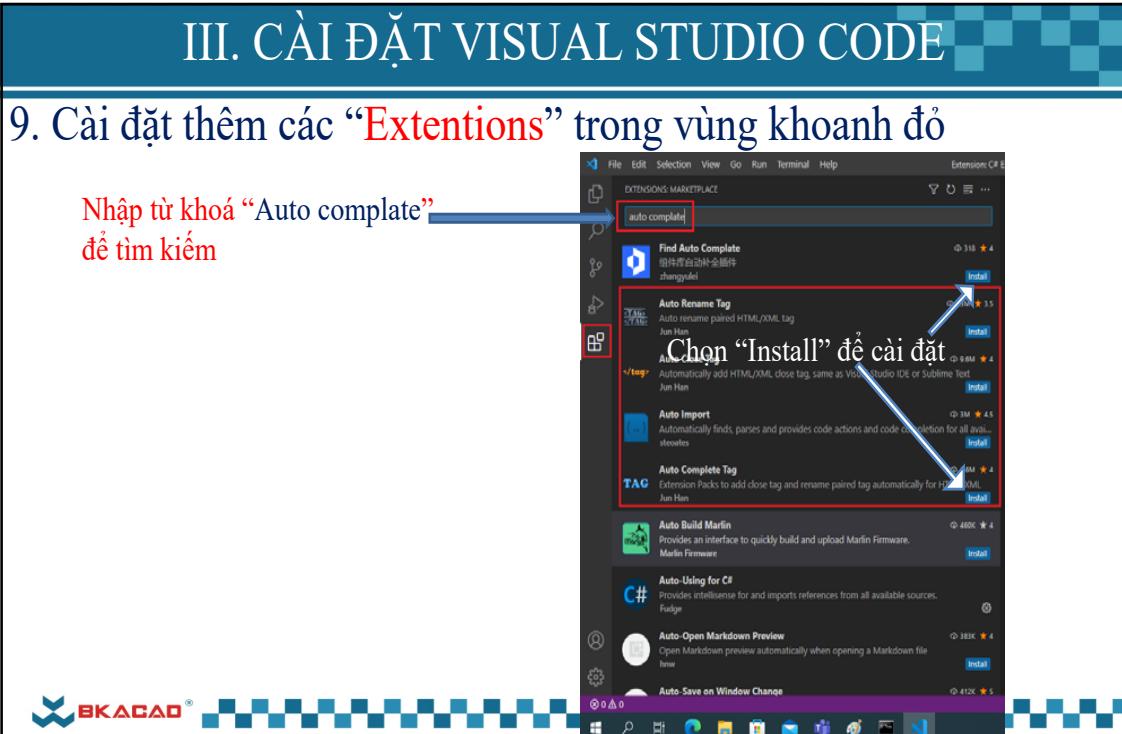
### III. CÀI ĐẶT VISUAL STUDIO CODE

#### 8. Cài đặt 1 số “Extentions” cho Visual Studio Code



### III. CÀI ĐẶT VISUAL STUDIO CODE

#### 9. Cài đặt thêm các “Extentions” trong vùng khoanh đỏ



## IV. Cài đặt SQL Server

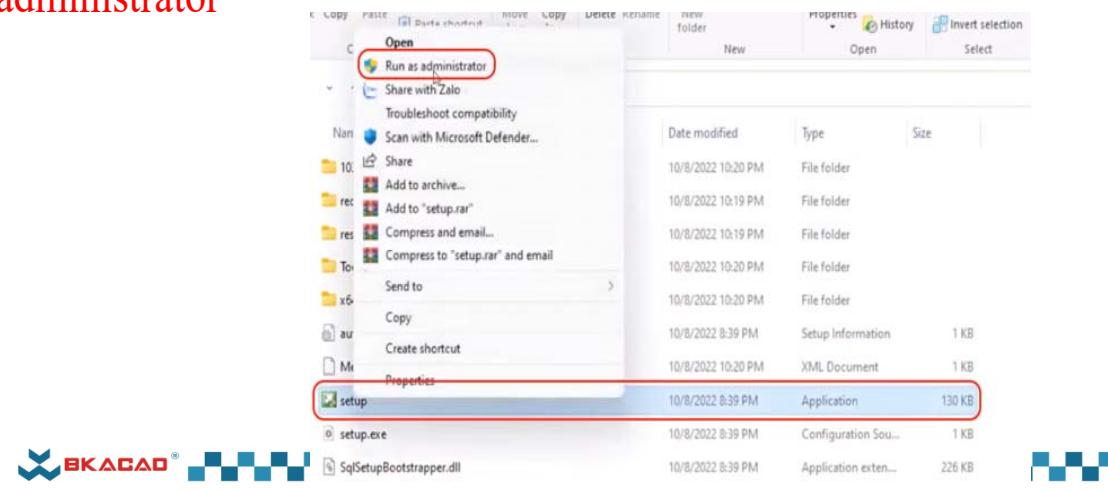
Cài đặt SQL Server bao gồm 2 phần:

- Cài đặt **SQL Server** để có database engine (sử dụng để lưu trữ, xử lý, phân tích dữ liệu)
- Cài đặt **SQL Server Management Studio** (sử dụng để quản lý database engine)



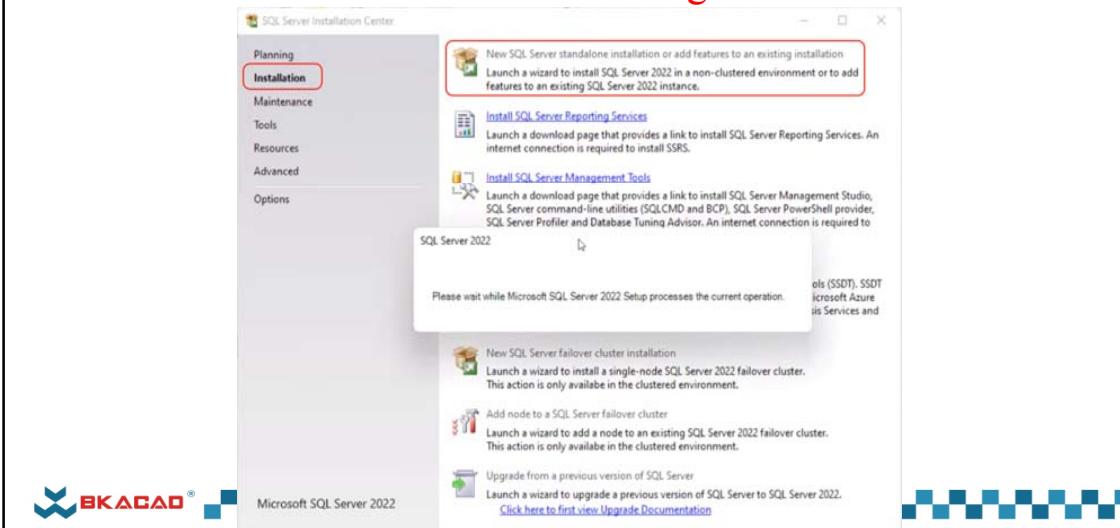
## IV. Cài đặt SQL Server

1. Giải nén file tải về, mở thư mục vừa giải nén (chứa file cài đặt), kích chuột phải vào file “**Setup**” chọn “**Run as administrator**”



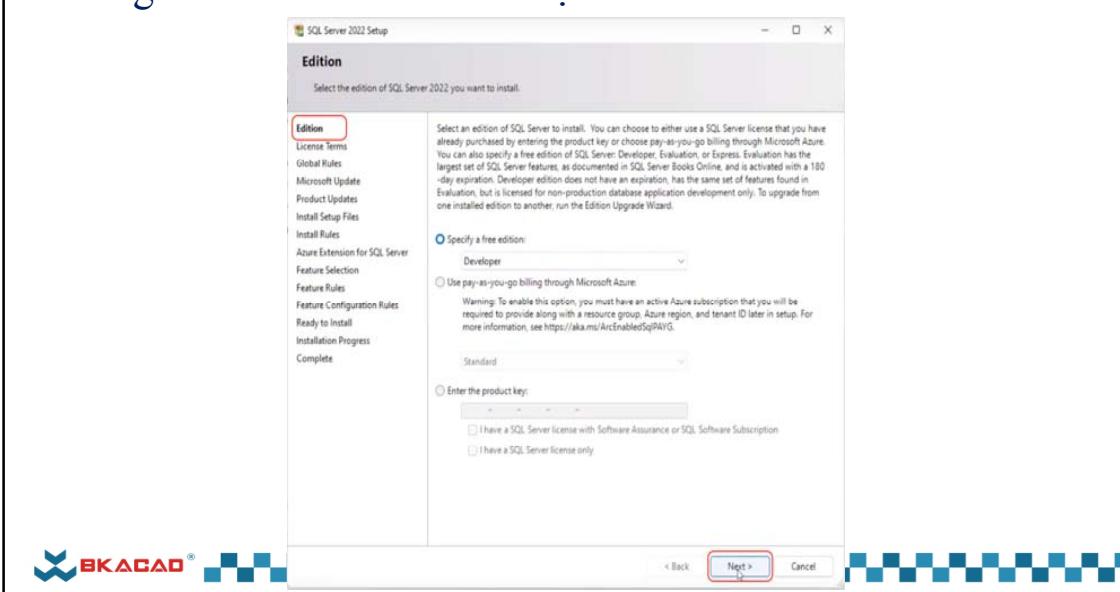
## IV. Cài đặt SQL Server

2. Trong cửa sổ hiện ra chọn “**Installation**” → chọn “**New SQL Server standalone installation ... existing installation**”



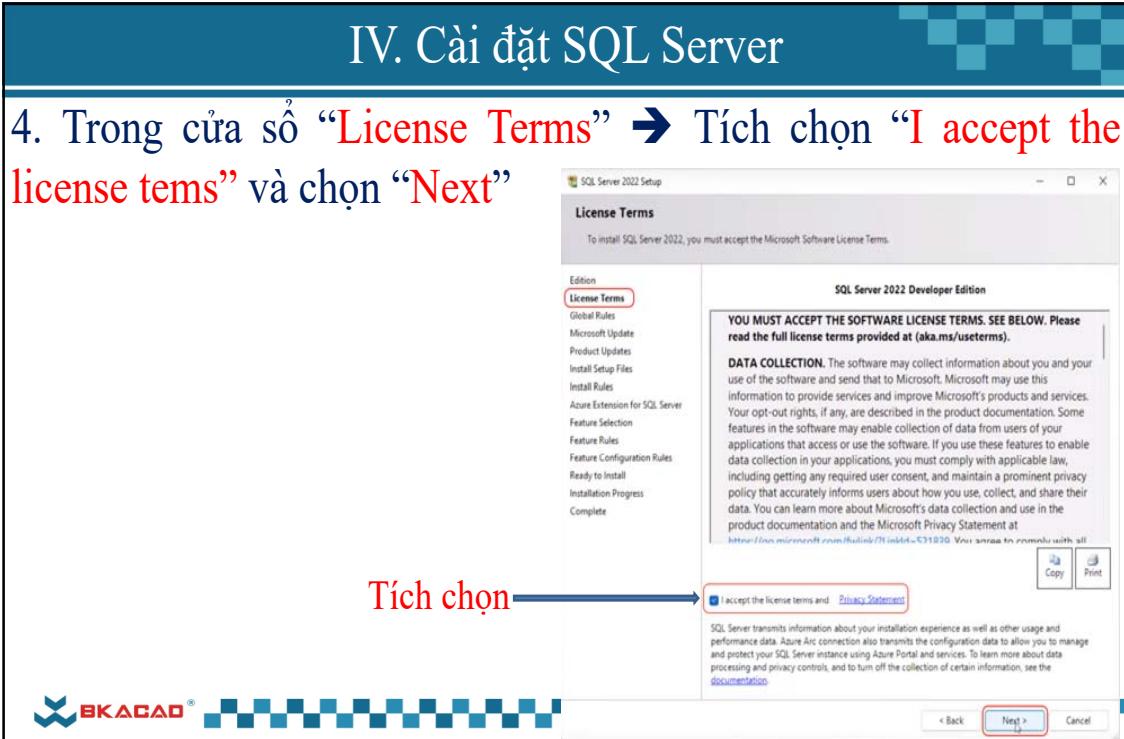
## IV. Cài đặt SQL Server

3. Trong cửa sổ “**Edition**” → chọn “**Next**”



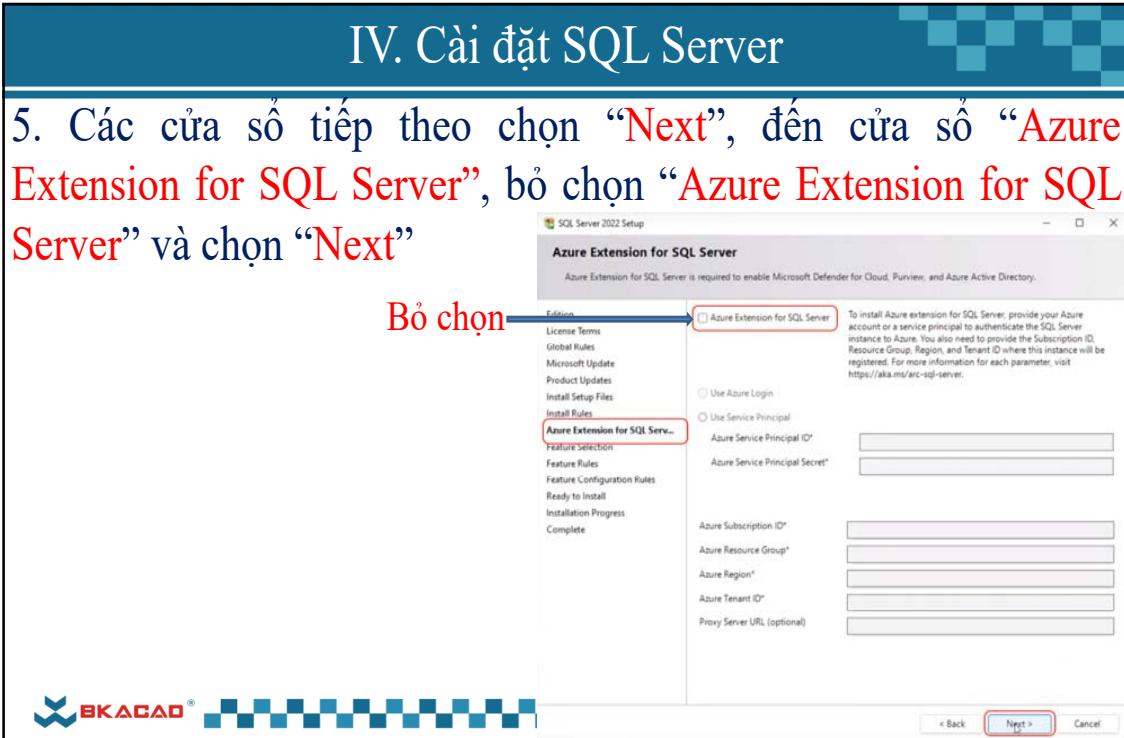
## IV. Cài đặt SQL Server

4. Trong cửa sổ “License Terms” → Tích chọn “I accept the license terms” và chọn “Next”



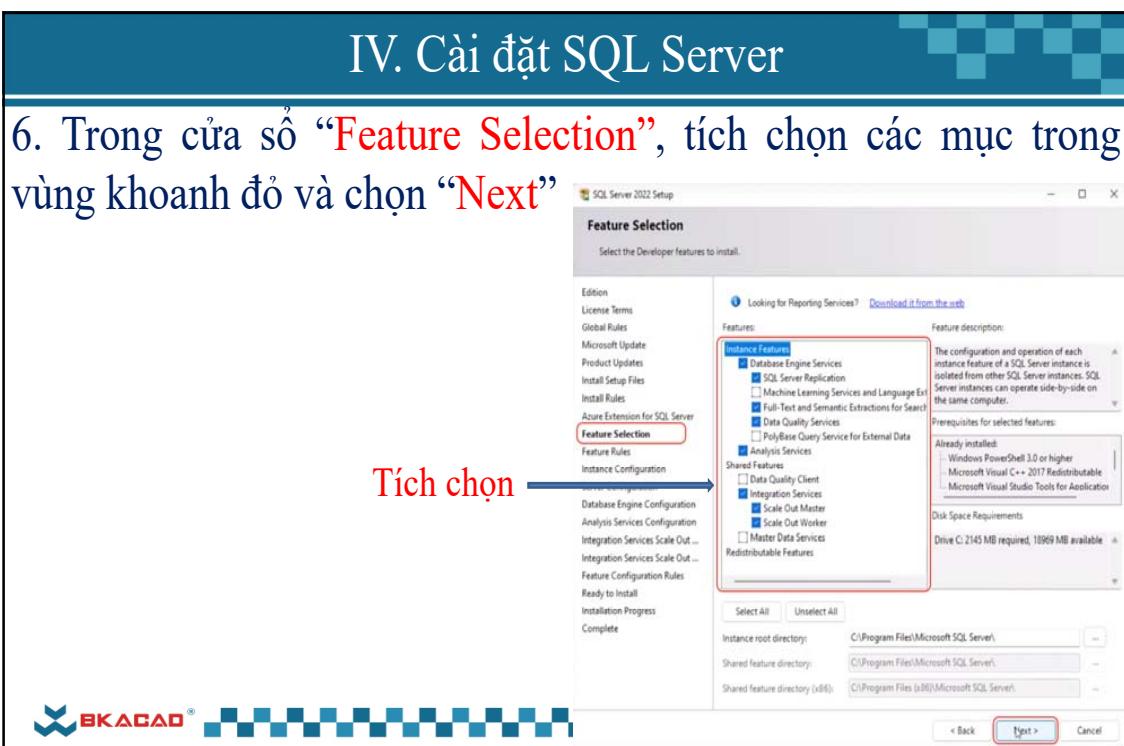
## IV. Cài đặt SQL Server

5. Các cửa sổ tiếp theo chọn “Next”, đến cửa sổ “Azure Extension for SQL Server”, bỏ chọn “Azure Extension for SQL Server” và chọn “Next”



## IV. Cài đặt SQL Server

6. Trong cửa sổ “Feature Selection”, tích chọn các mục trong vùng khoanh đỏ và chọn “Next”



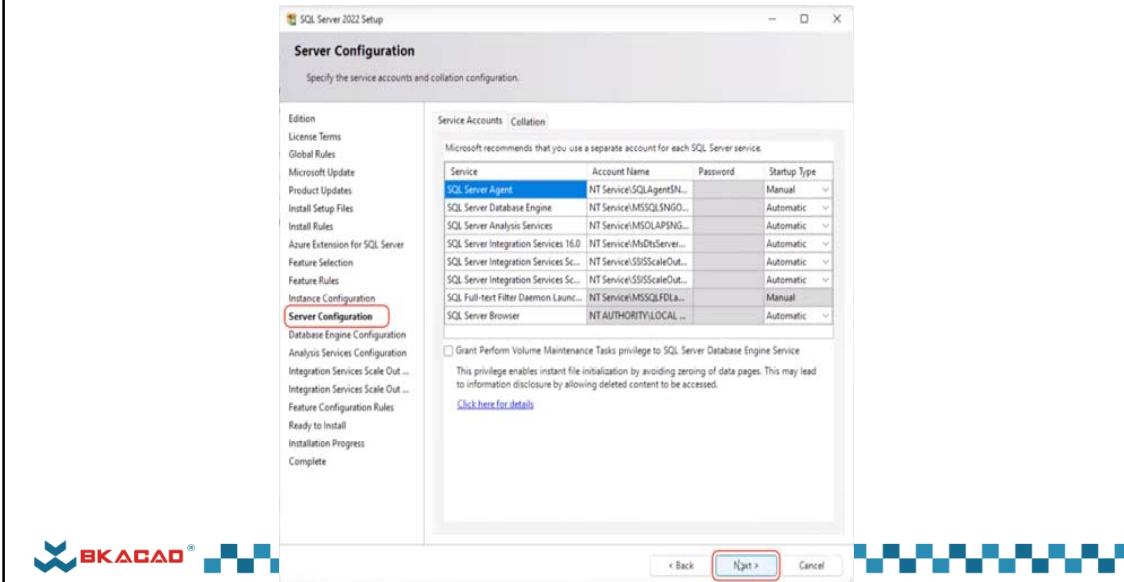
## IV. Cài đặt SQL Server

7. Trong cửa sổ “Instance Configuration”, nhập tên bạn muốn dùng (Viết không dấu) và chọn “Next”



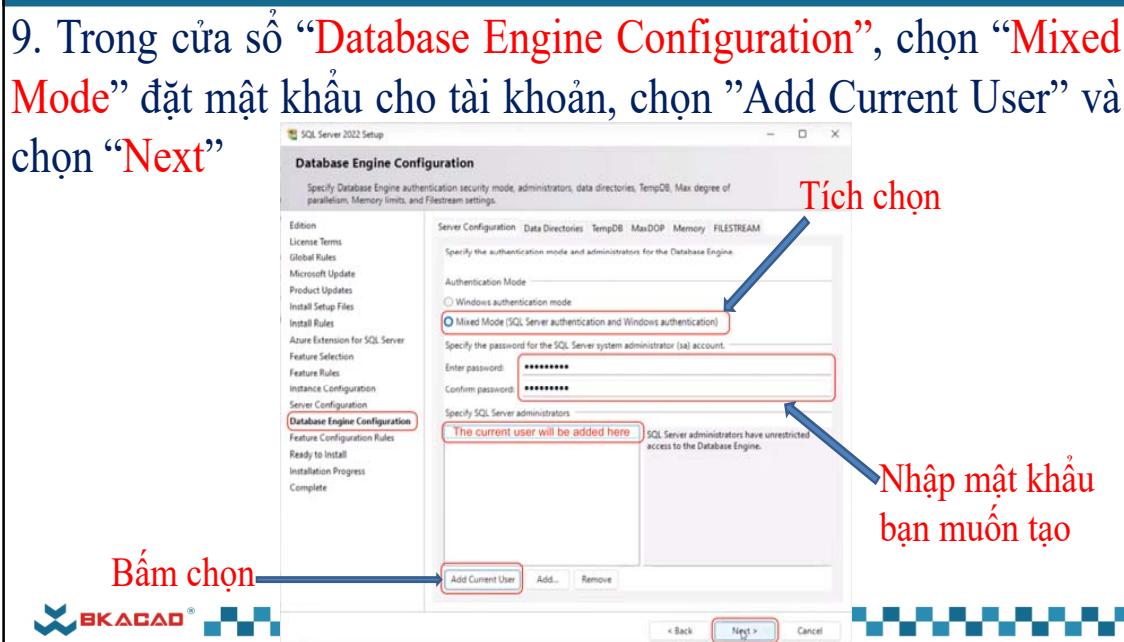
## IV. Cài đặt SQL Server

### 8. Trong cửa sổ “Server Configuration”, chọn “Next”



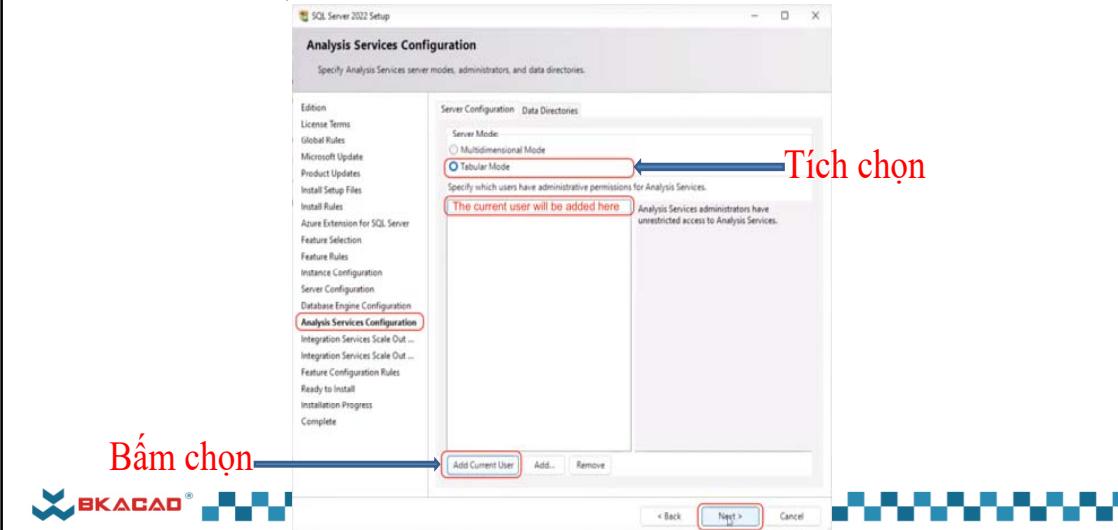
## IV. Cài đặt SQL Server

### 9. Trong cửa sổ “Database Engine Configuration”, chọn “Mixed Mode” đặt mật khẩu cho tài khoản, chọn ”Add Current User” và chọn “Next”



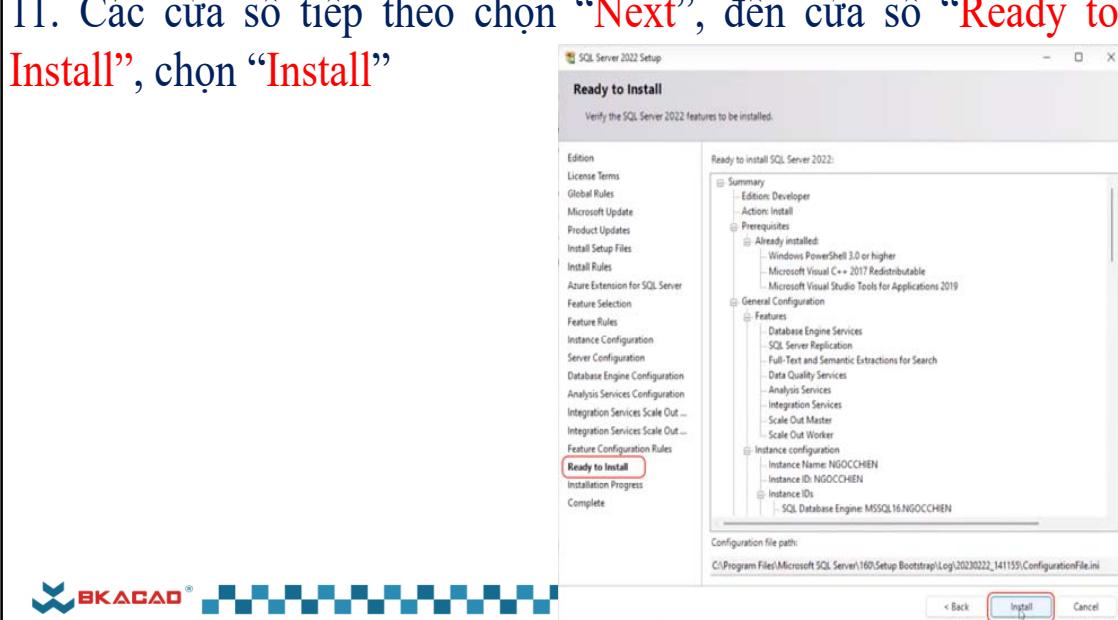
## IV. Cài đặt SQL Server

10. Trong cửa sổ “Analysis Services Configuration”, chọn “Tabular Mode”, chọn ”Add Current User” và chọn “Next”



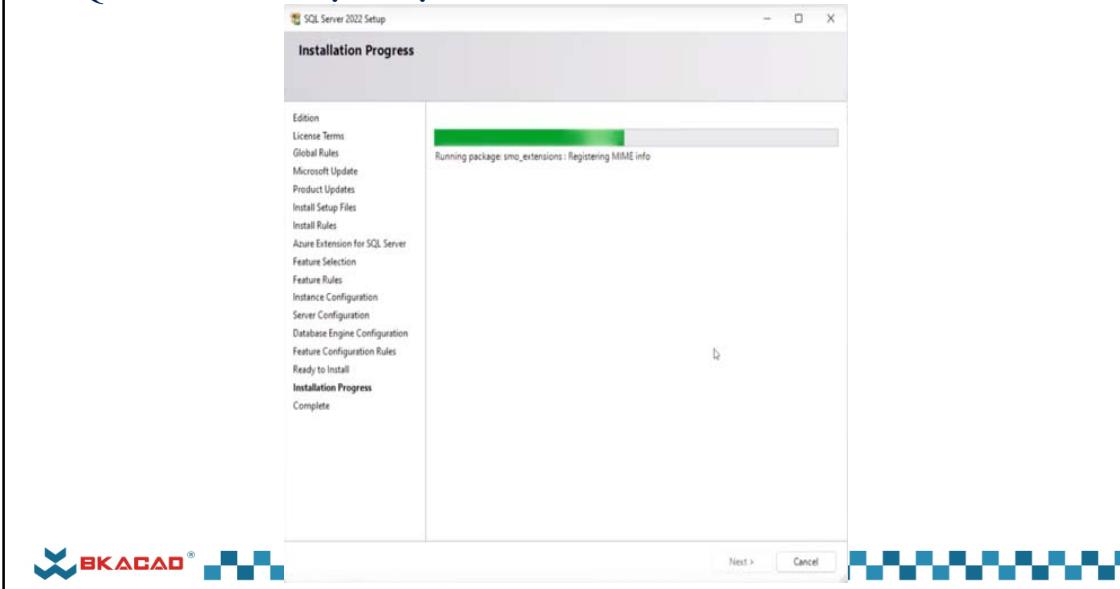
## IV. Cài đặt SQL Server

11. Các cửa sổ tiếp theo chọn “Next”, đến cửa sổ “Ready to Install”, chọn “Install”



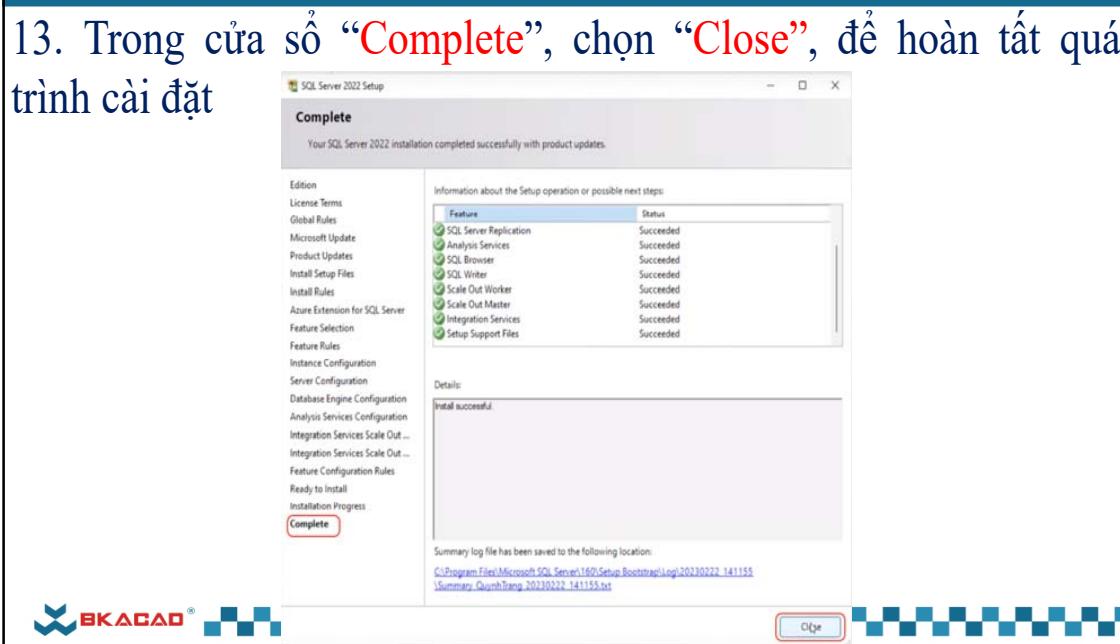
## IV. Cài đặt SQL Server

### 12. Quá trình cài đặt được diễn ra



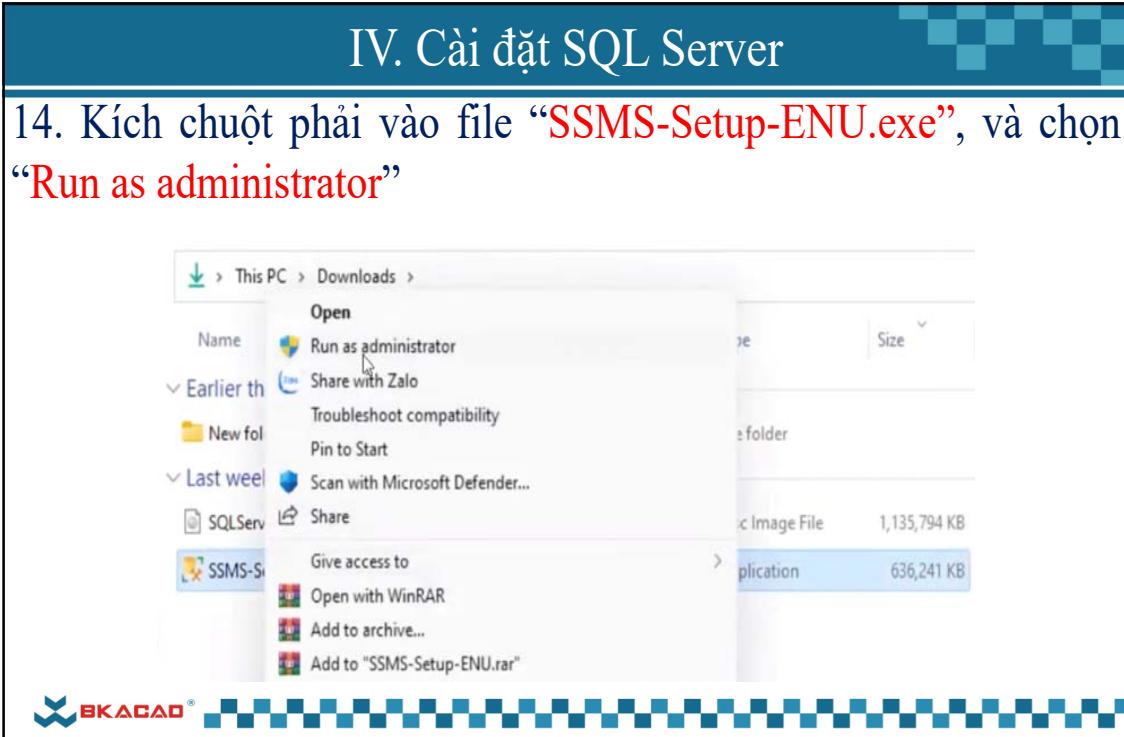
## IV. Cài đặt SQL Server

### 13. Trong cửa sổ “Complete”, chọn “Close”, để hoàn tất quá trình cài đặt



## IV. Cài đặt SQL Server

14. Kích chuột phải vào file “SSMS-Setup-ENU.exe”, và chọn “Run as administrator”



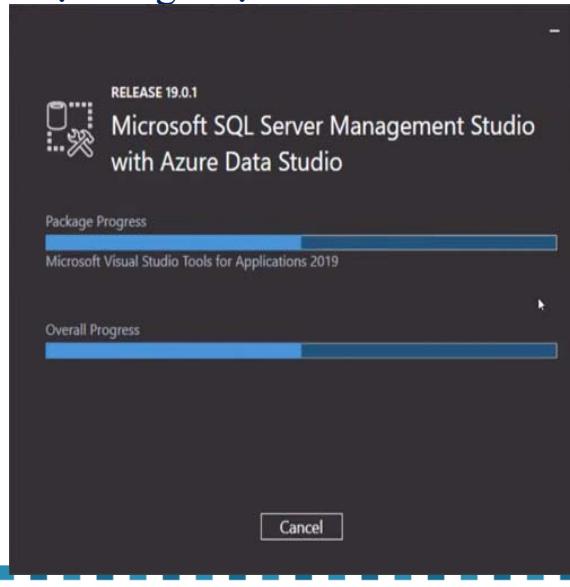
## IV. Cài đặt SQL Server

15. Chọn “Install” để tiến hành cài đặt



## IV. Cài đặt SQL Server

### 16. Quá trình cài đặt đang được diễn ra



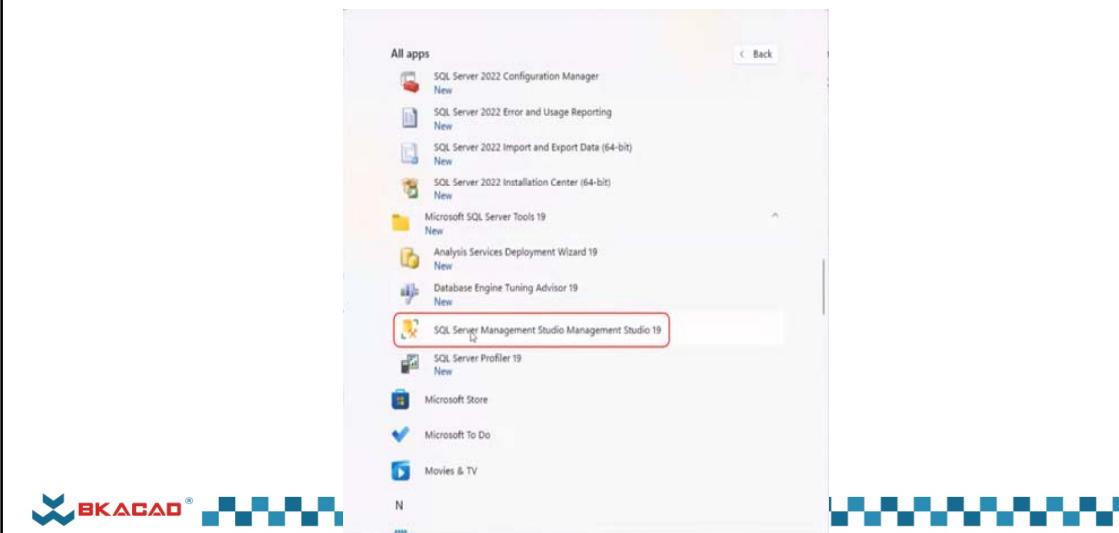
## IV. Cài đặt SQL Server

### 17. Sau khi quá trình cài đặt hoàn thành, chọn “Close” để kết thúc cài đặt



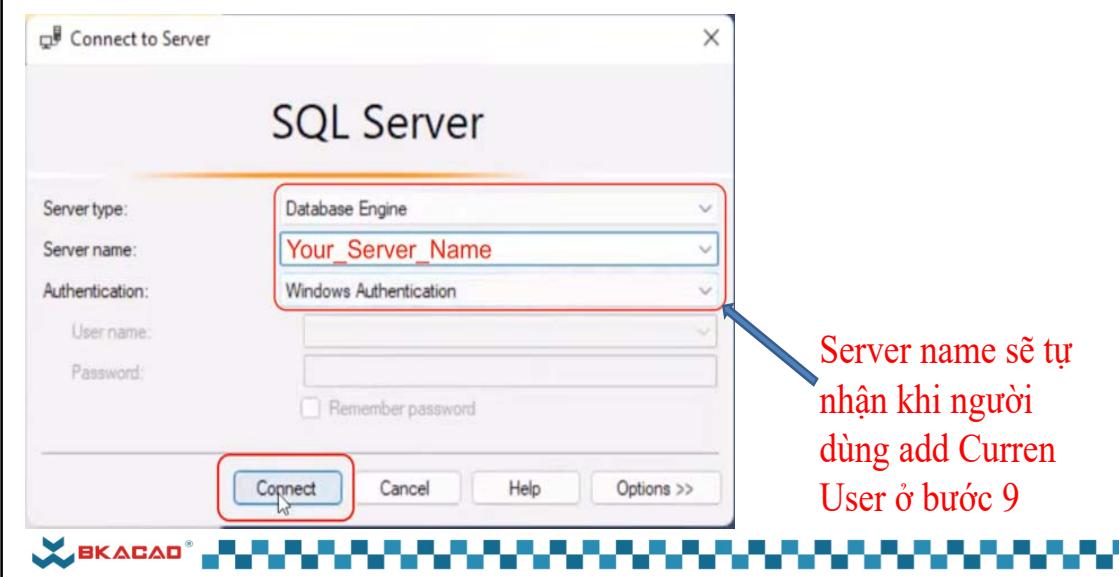
## IV. Cài đặt SQL Server

### 18. Mở ứng dụng “SQL Server Management Studio Management Studio 19”



## IV. Cài đặt SQL Server

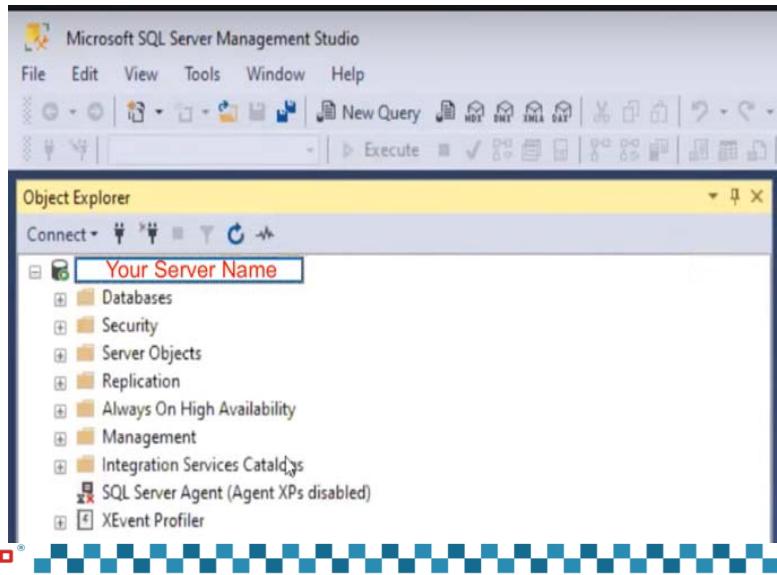
### 19. Đăng nhập vào SQL Server



Server name sẽ tự nhận khi người dùng add Current User ở bước 9

## IV. Cài đặt SQL Server

### 19. Đăng nhập vào SQL Server thành công



## V. Cài đặt git

### 1. Truy cập google và tìm kiếm với từ khóa “download git”

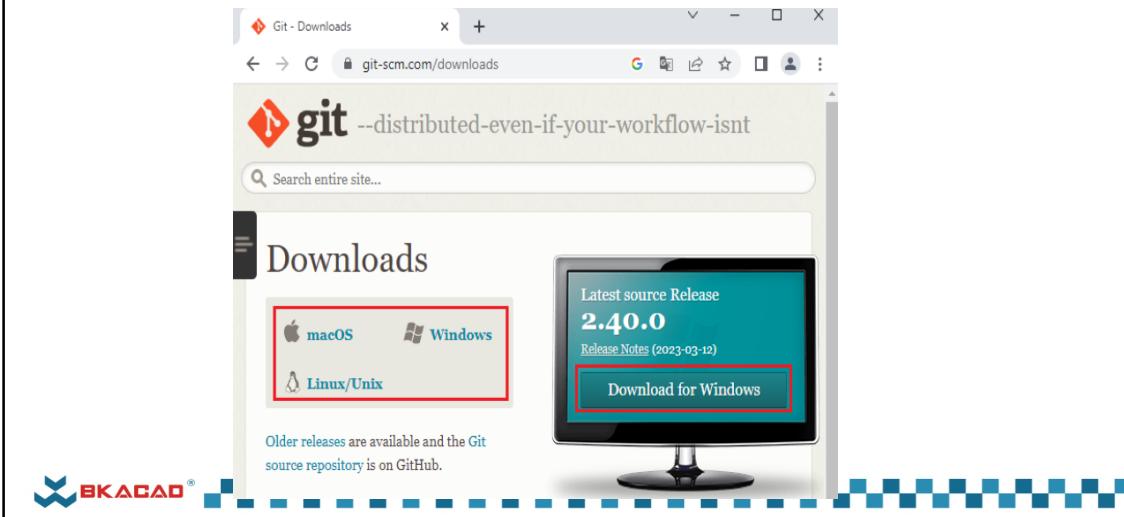
Khoảng 395.000.000 kết quả (0,29 giây)

git-scm.com  
https://git-scm.com/downloads · Dịch trang này ·  
Downloads - Git SCM

Latest source Release. 2.40.0 Release Notes (2023-03-12) Download for Linux. GUI Clients.  
Git comes with built-in GUI tools (git-gui, gitk), but there are ...  
Brew install git · Download for Linux and Unix · GUI Clients

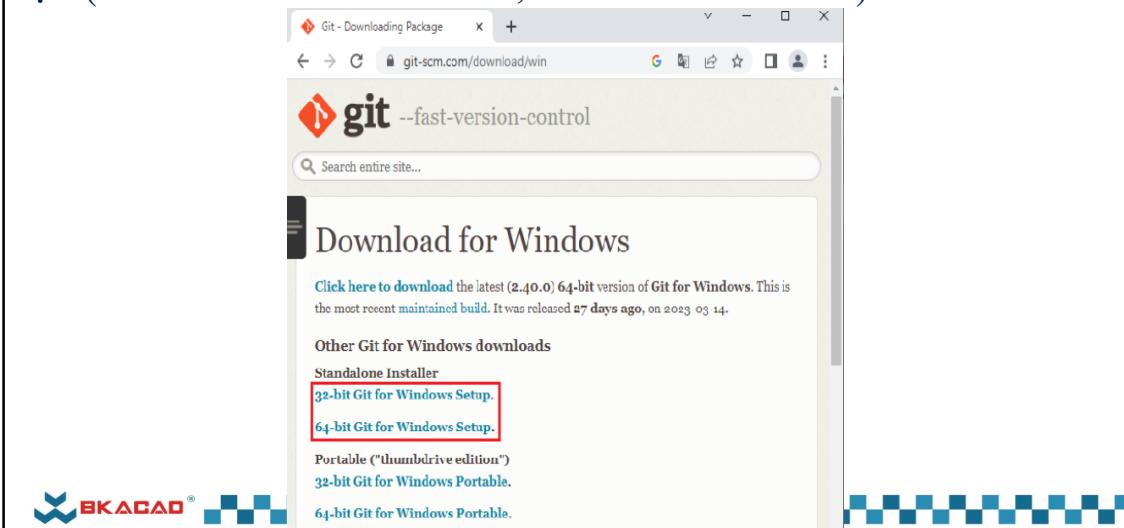
## V. Cài đặt git

2. Truy cập vào link đầu tiên (<https://git-scm.com/downloads>)
3. Chọn phiên bản phù hợp với hệ điều hành, để tải về.



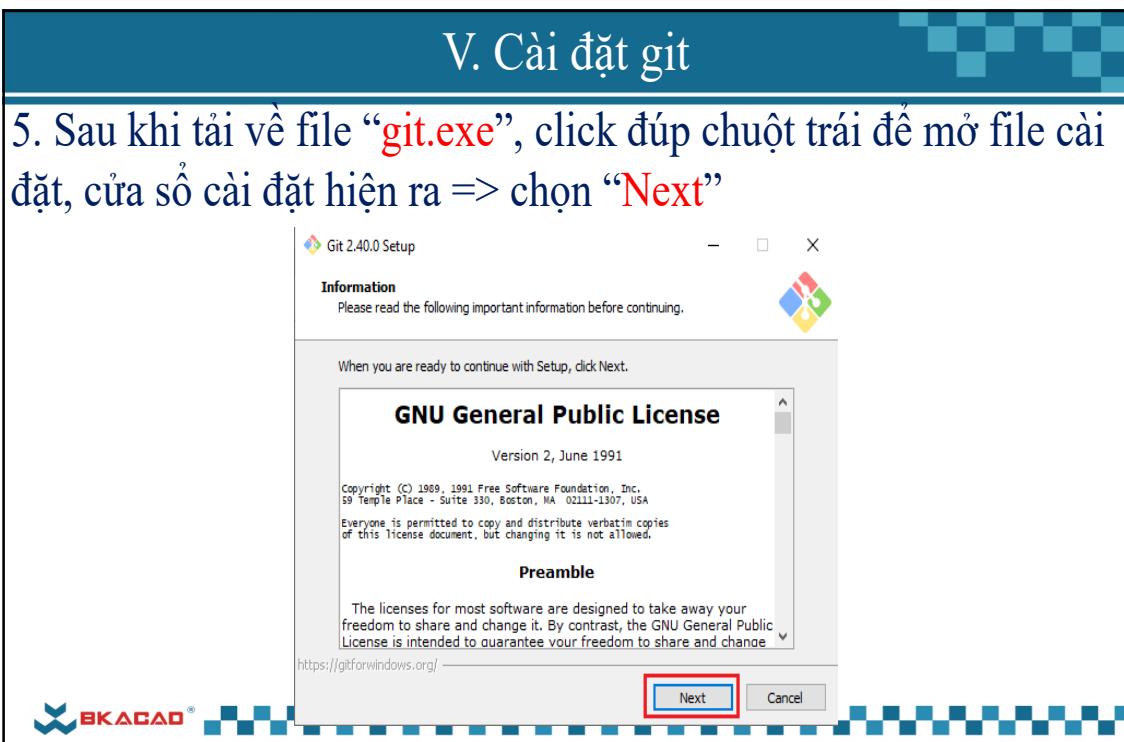
## V. Cài đặt git

4. Trong cửa sổ mới chọn phiên bản phù hợp với máy tính của bạn (32bit cho windows 32bit, 64bit cho win64bit)



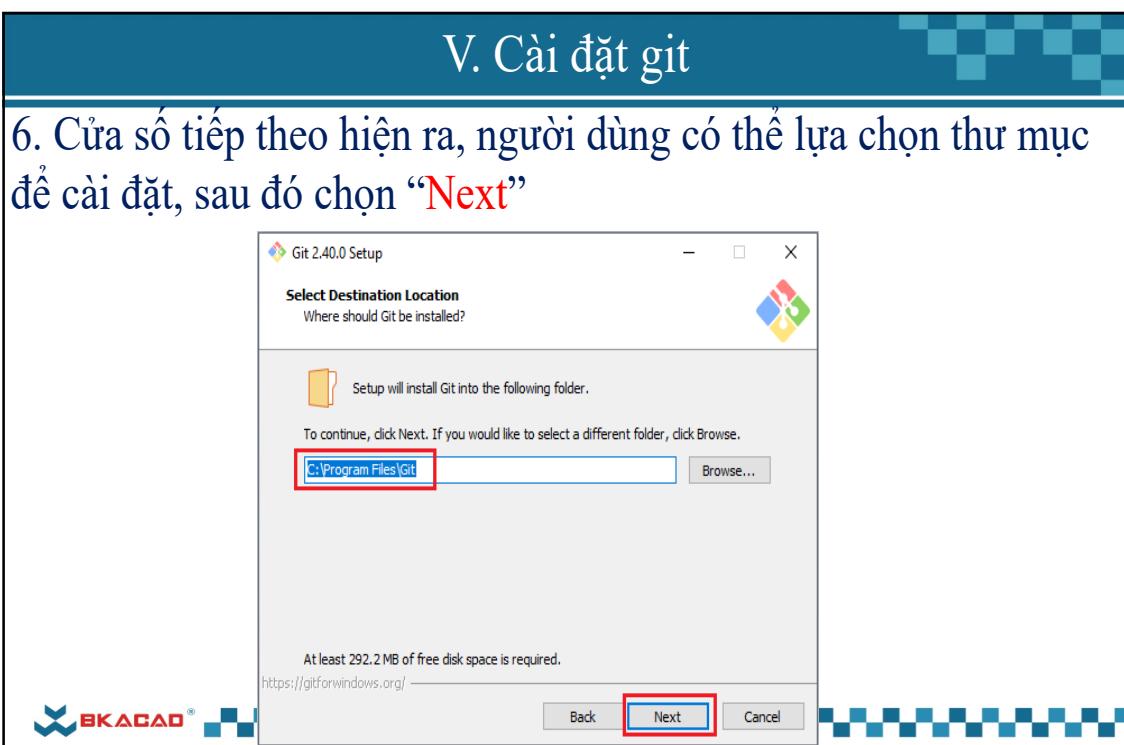
## V. Cài đặt git

5. Sau khi tải về file “git.exe”, click đúp chuột trái để mở file cài đặt, cửa sổ cài đặt hiện ra => chọn “Next”



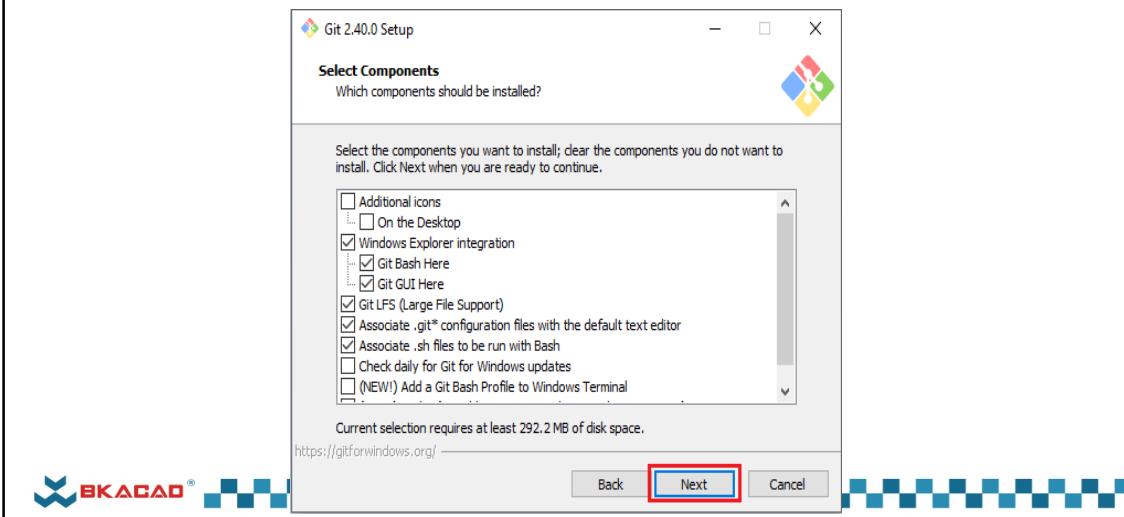
## V. Cài đặt git

6. Cửa sổ tiếp theo hiện ra, người dùng có thể lựa chọn thư mục để cài đặt, sau đó chọn “Next”



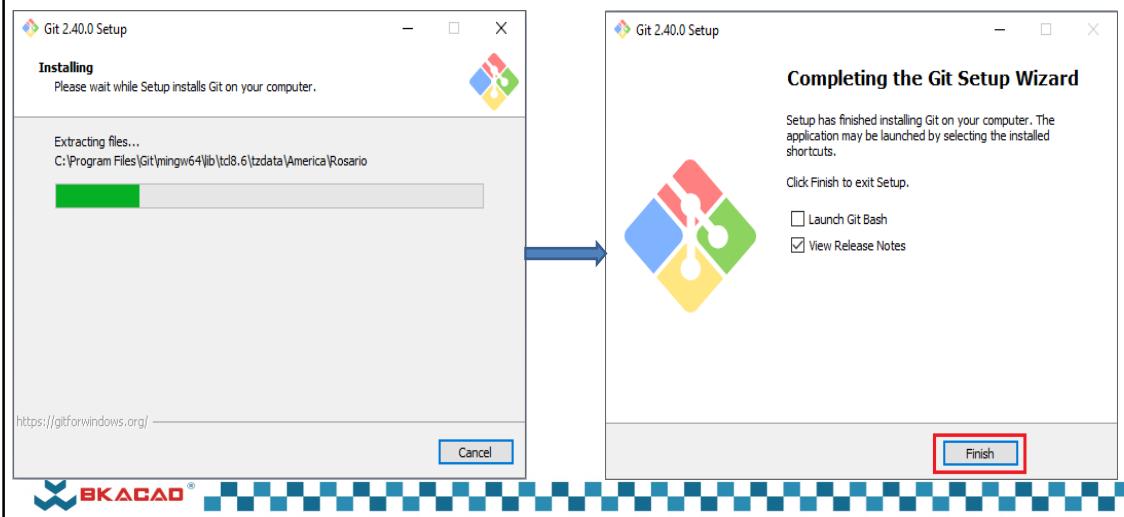
## V. Cài đặt git

7. Cửa sổ tiếp theo hiện ra, chọn “Next”



## V. Cài đặt git

8. Phần mềm đang được cài đặt, khi có nút lệnh “Finish” xuất hiện, người dùng bấm vào nút lệnh “Finish” để hoàn thành quá trình cài đặt.



## V. Cài đặt git

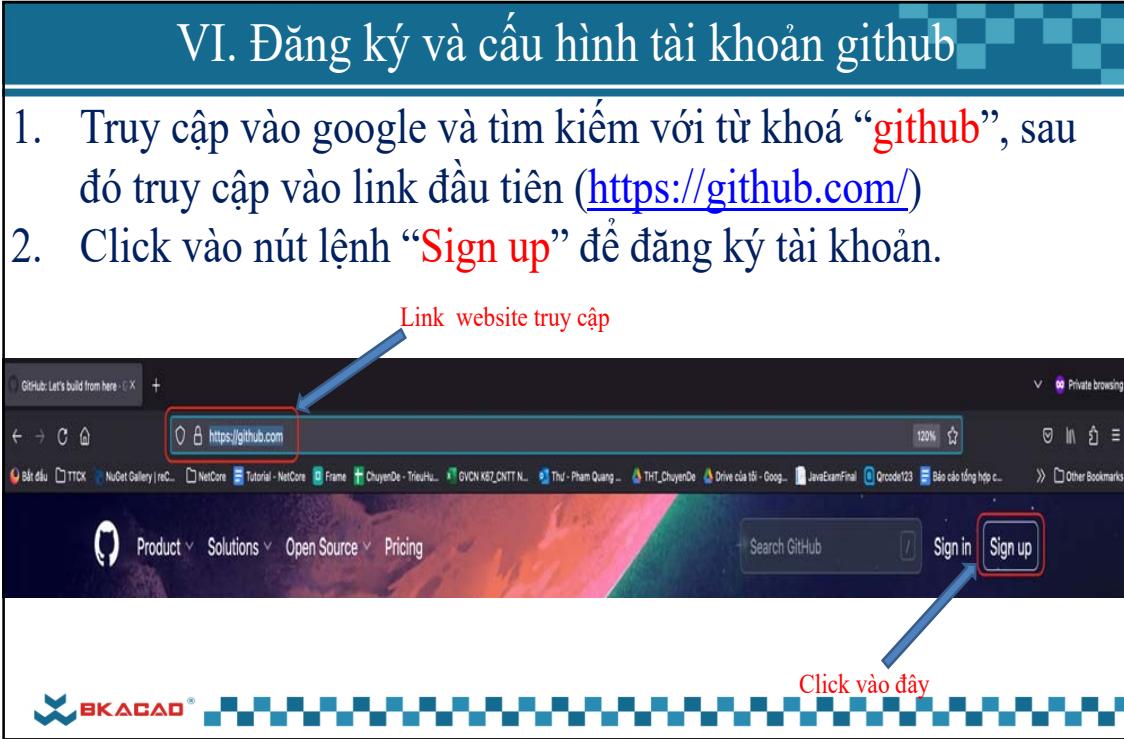
9. Mở CMD và nhập lệnh “**git version**” để kiểm tra xem cài đặt git thành công chưa, nếu cài đặt thành công sẽ hiển thị phiên bản đã cài đặt.

The image shows two side-by-side Windows Command Prompt windows. Both windows have the title 'C:\WINDOWS\system32\cmd.exe' and show the Microsoft Windows [Version 10.0.19044.2728] (c) Microsoft Corporation. All rights reserved. prompt. In the first window, the command 'git version' is entered, and the output 'git version 2.40.0.windows.1' is displayed. A red arrow points from the bottom of the first window down to the top of the second window. In the second window, the command 'git version' is also entered, and the same output 'git version 2.40.0.windows.1' is displayed. This indicates that the command was run again after the first window was closed.



## VI. Đăng ký và cấu hình tài khoản github

1. Truy cập vào google và tìm kiếm với từ khoá “**github**”, sau đó truy cập vào link đầu tiên (<https://github.com/>)
2. Click vào nút lệnh “**Sign up**” để đăng ký tài khoản.



## VI. Đăng ký và cấu hình tài khoản github

### 3. Nhập địa chỉ Email của bạn, chọn “Continue”



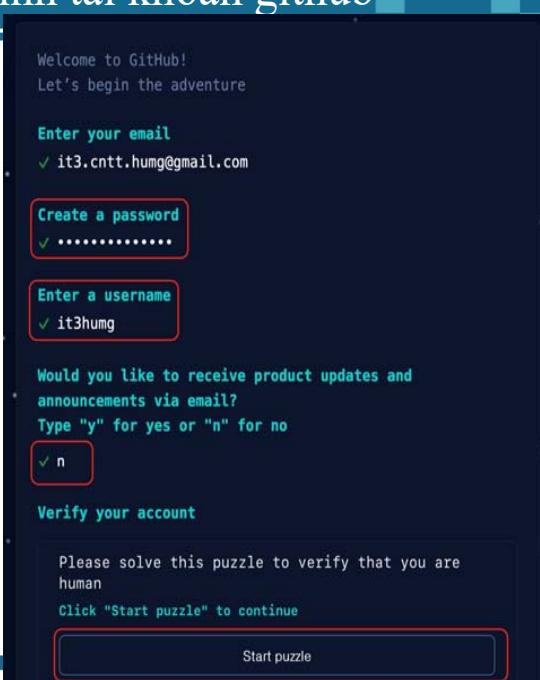
Địa chỉ email bạn muốn dùng để đăng ký tài khoản github



## VI. Đăng ký và cấu hình tài khoản github

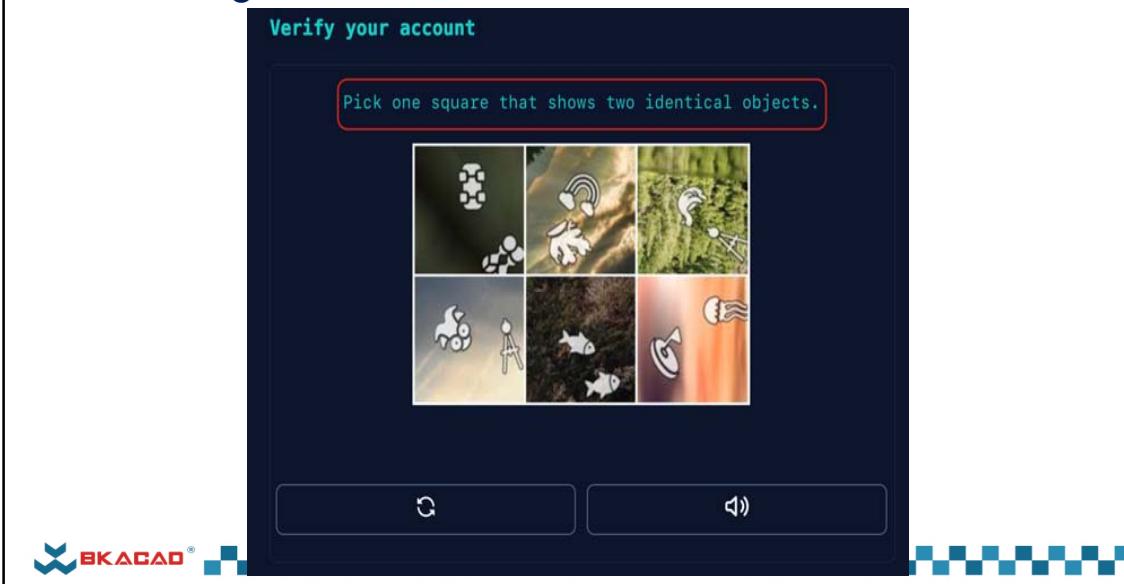
### 4. Tương tự nhập các thông tin Password, Username, n (không nhận thông báo từ github)

- Sau đó chọn “Start Puzzle” để tiến hành xác minh tài khoản.



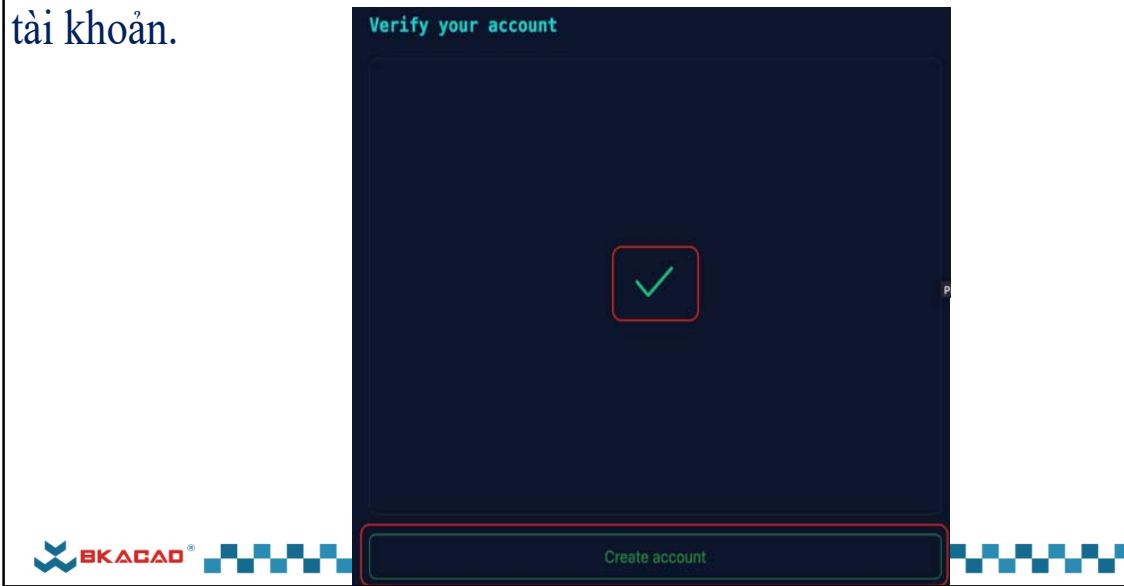
## VII. Đăng ký và cấu hình tài khoản github

5. Tiến hành giải câu đố để xác minh tài khoản.



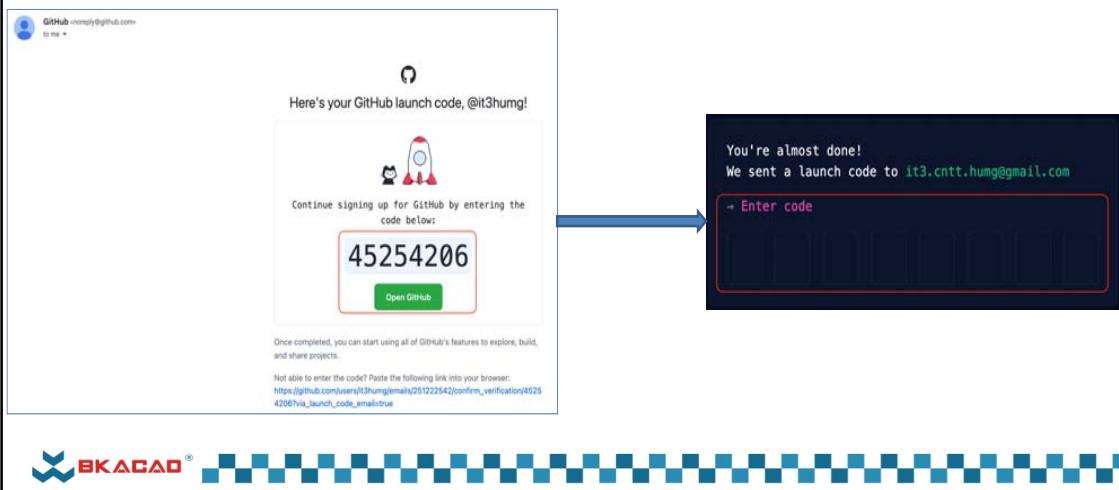
## VII. Đăng ký và cấu hình tài khoản github

6. Sau khi giải câu đố thành công, chọn “Create Account” để tạo tài khoản.



## VI. Đăng ký và cấu hình tài khoản github

7. Một email từ github gửi tới, chứa link và code để kích hoạt tài khoản, chọn “Open Github” để mở link kích hoạt tài khoản.



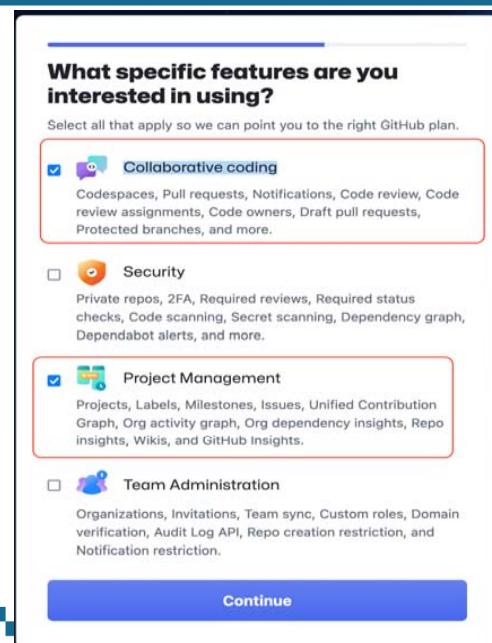
## VI. Đăng ký và cấu hình tài khoản github

8. Sau khi nhập code vào link kích hoạt tài khoản, một cửa sổ mới hiện ra, chọn “Just me” và “Student” => chọn “Continue”



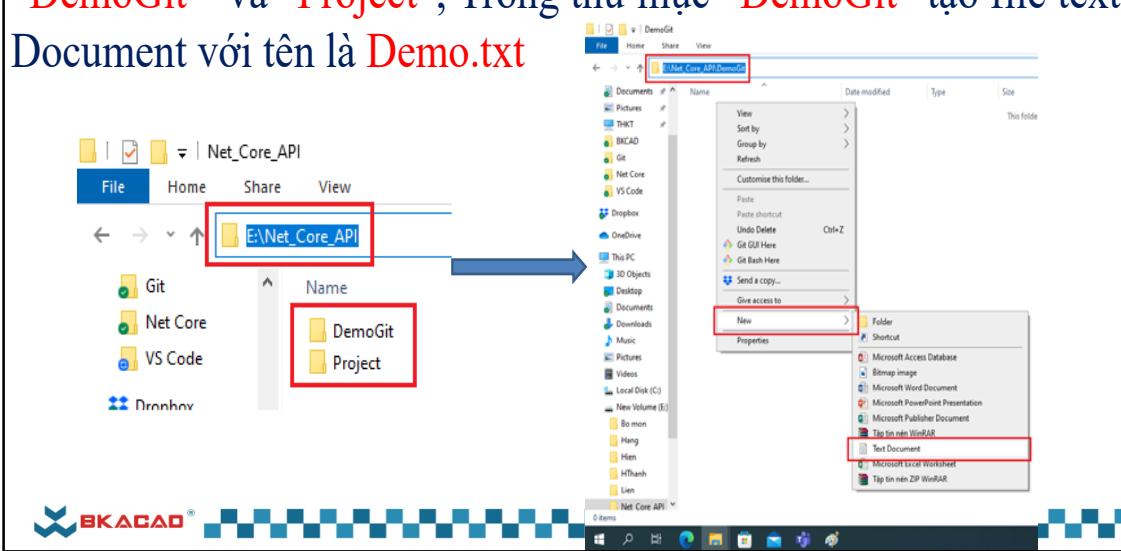
## VI. Đăng ký và cấu hình tài khoản github

9. Một cửa sổ hiện ra, người dùng chọn mục “Collaborative coding” và “Project Management” => chọn “Continue”



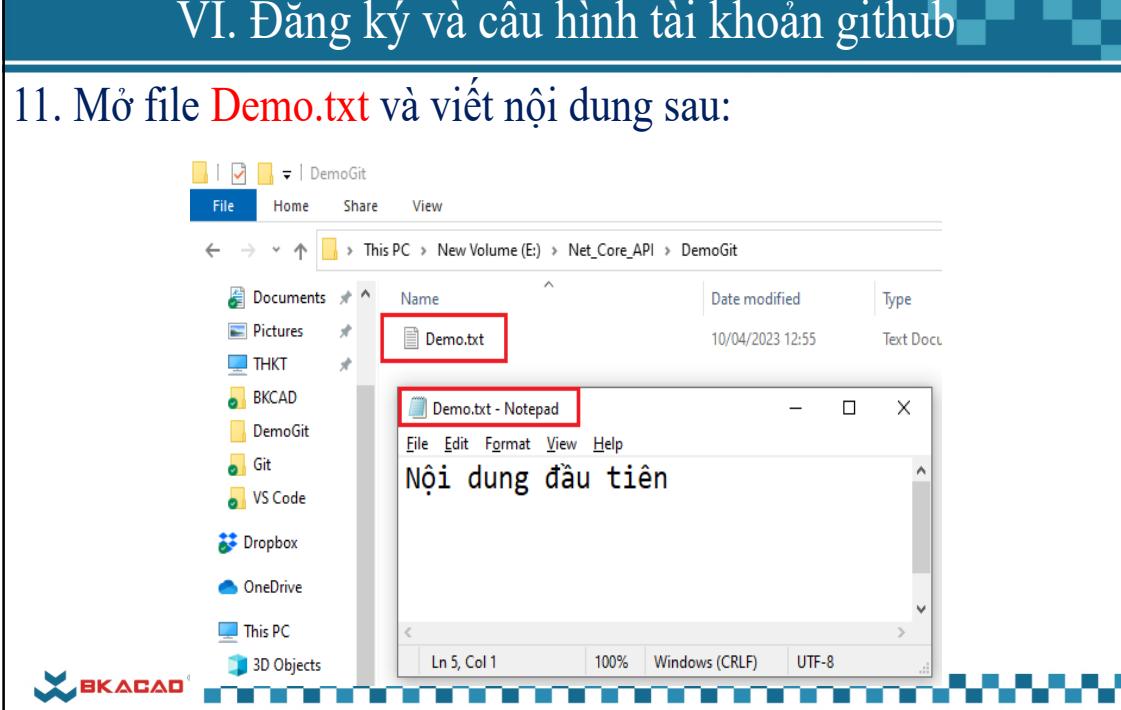
## VI. Đăng ký và cấu hình tài khoản github

10. Tạo thư mục “Net\_Core\_API”, trong đó tạo 2 thư mục “DemoGit” và “Project”, Trong thư mục “DemoGit” tạo file text Document với tên là Demo.txt



## VI. Đăng ký và cấu hình tài khoản github

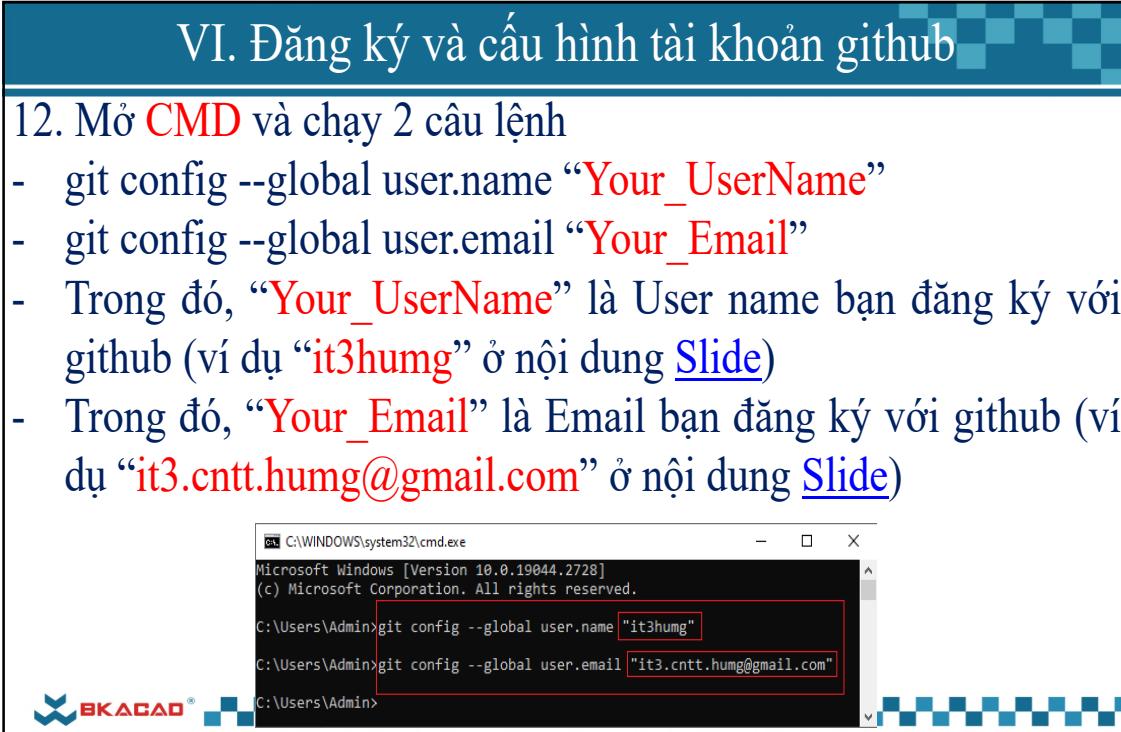
### 11. Mở file **Demo.txt** và viết nội dung sau:



## VI. Đăng ký và cấu hình tài khoản github

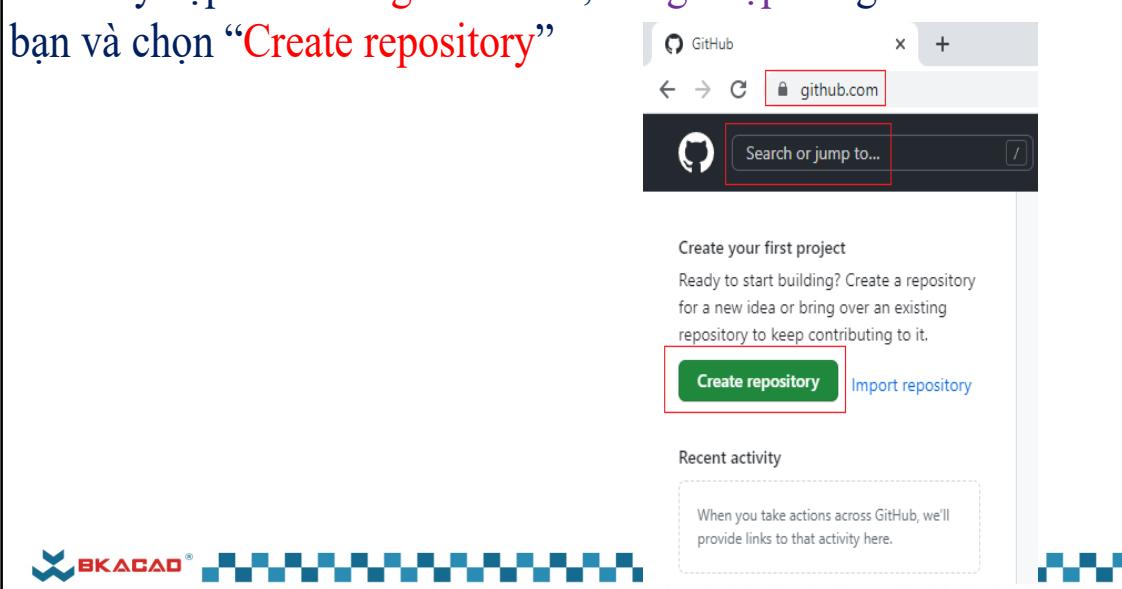
### 12. Mở **CMD** và chạy 2 câu lệnh

- git config --global user.name "**Your\_UserName**"
- git config --global user.email "**Your\_Email**"
- Trong đó, "**Your\_UserName**" là User name bạn đăng ký với github (ví dụ "**it3humg**" ở nội dung [Slide](#))
- Trong đó, "**Your\_Email**" là Email bạn đăng ký với github (ví dụ "**it3.cntt.humg@gmail.com**" ở nội dung [Slide](#))



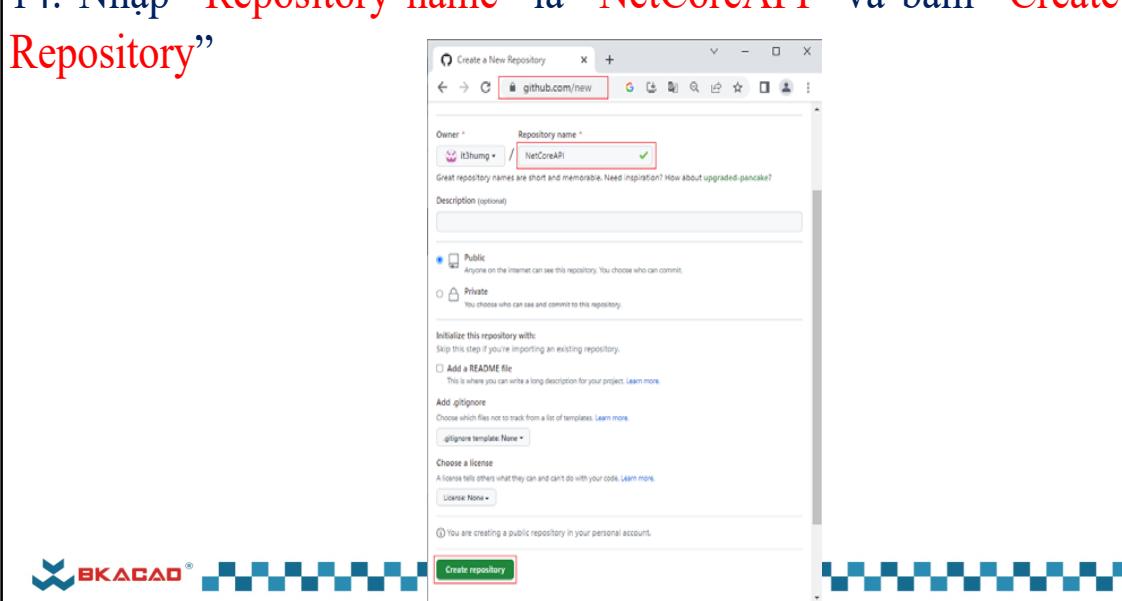
## VI. Đăng ký và cấu hình tài khoản github

13. Truy cập vào link [github.com](https://github.com), đăng nhập bằng tài khoản của bạn và chọn “Create repository”



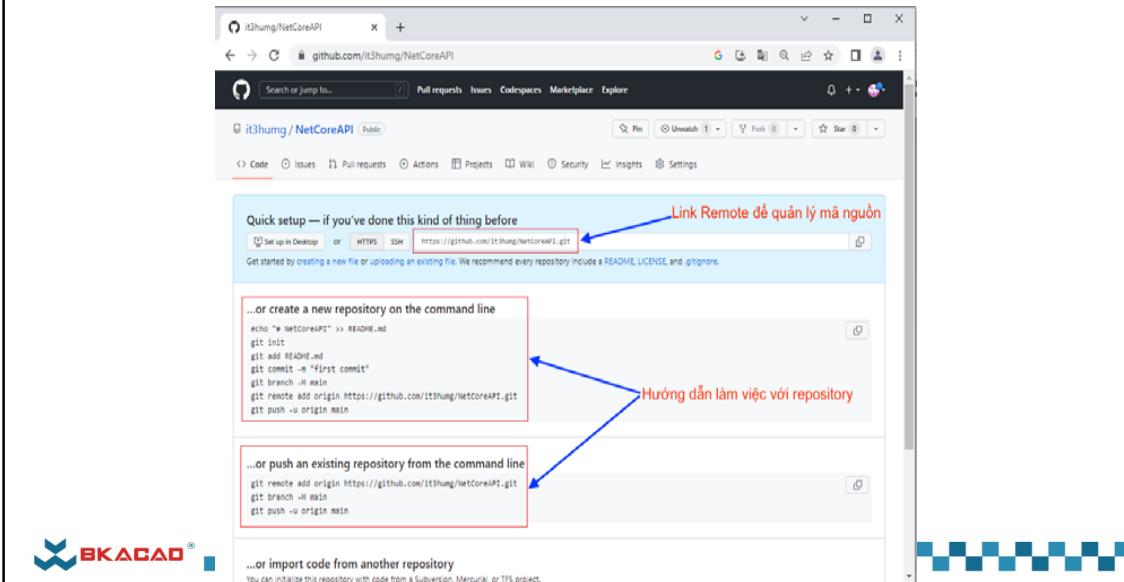
## VI. Đăng ký và cấu hình tài khoản github

14. Nhập “Repository name” là “NetCoreAPI” và bấm “Create Repository”



## VI. Đăng ký và cấu hình tài khoản github

### 15. Cửa sổ tạo “Repository” thành công



## VI. Đăng ký và cấu hình tài khoản github

### 16. Mở CMD/Terminal ở thư mục Net\_Core\_API và chạy các lệnh sau:

- git init
- git add .
- git commit -m “first commit”
- git branch -M main

```
C:\Windows\system32\cmd.exe - git push -u origin main
Microsoft Windows [Version 10.0.19044.2728]
(c) Microsoft Corporation. All rights reserved.

E:\Net_Core_API>git init
Initialized empty Git repository in E:/Net_Core_API/.git/
E:\Net_Core_API>git add .
E:\Net_Core_API>git commit -m "First commit"
[master (root-commit) 51313e6] first commit
 1 file changed, 4 insertions(+)
 create mode 100644 DemoGit/Demo.txt
E:\Net_Core_API>git branch -M main
E:\Net_Core_API>git remote add origin https://github.com/it3hung/NetCoreAPI.git
E:\Net_Core_API>git push -u origin main
```

- git remote add origin **Link\_Remote\_Github**
- git push -u origin main



## VI. Đăng ký và cấu hình tài khoản github

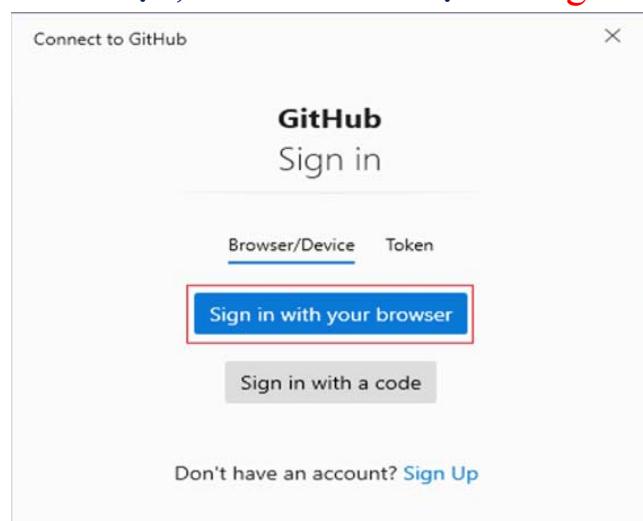
17. Sau khi chạy lệnh “`git push -u origin main`”, hệ thống yêu cầu hoàn thành xác thực trên trình duyệt (“Please complete authentication in your browser...”)

```
PS C:\Windows\System32\cmd.exe
E:\Net_Core_API>git commit -m "first commit"
[master (root-commit) 51313e6] first commit
 1 file changed, 4 insertions(+)
 create mode 100644 DemoGit/Demo.txt
E:\Net_Core_API>git branch -M main
E:\Net_Core_API>git remote add origin https://github.com/it3humg/NetCoreAPI.git
E:\Net_Core_API>git push -u origin main
info: please complete authentication in your browser...
```



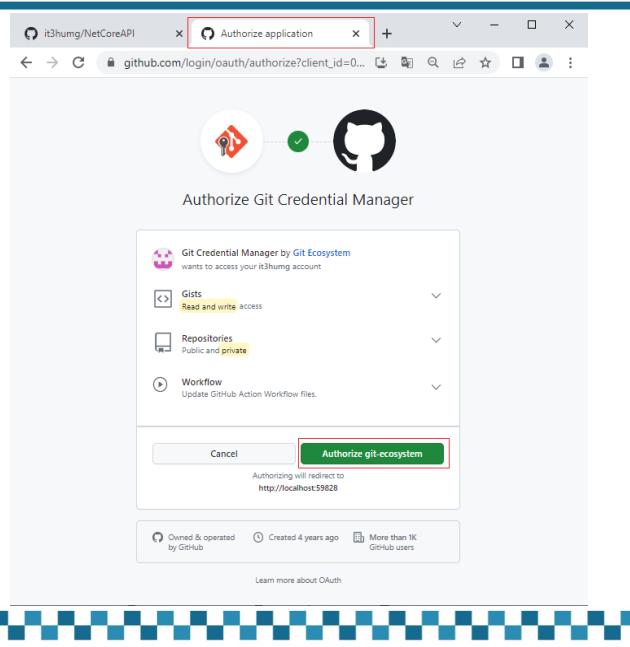
## VI. Đăng ký và cấu hình tài khoản github

18. Trong cửa sổ xác thực, bấm vào nút lệnh “`Sign in with your browser`”



## VI. Đăng ký và cấu hình tài khoản github

19. Một cửa sổ xác thực trên web được mở ra, chọn “Authorize git-ecosystem”



20. Sau khi xác thực thành công, mã nguồn của project đã được đẩy lên github với thông báo “Writing objects: 100%”

```
C:\Windows\System32\cmd.exe
E:\Net_Core_API>git commit -m "first commit"
[master (root-commit) 51313e6] first commit
 1 file changed, 4 insertions(+)
 create mode 100644 DemoGit/Demo.txt

E:\Net_Core_API>git branch -M main

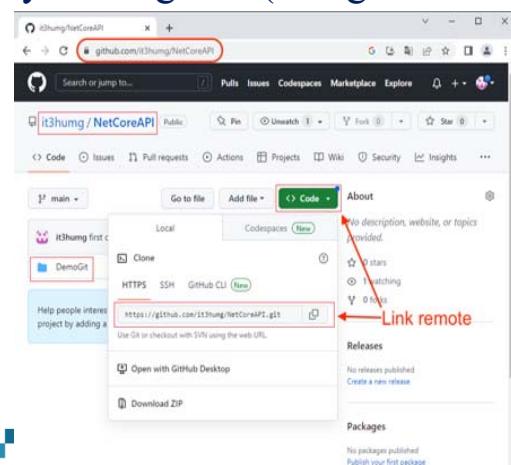
E:\Net_Core_API>git remote add origin https://github.com/it3humg/NetCoreAPI.git

E:\Net_Core_API>git push -u origin main
info: please complete authentication in your browser...
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Writing objects: 100% (4/4), 279 bytes | 139.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/it3humg/NetCoreAPI.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
E:\Net_Core_API>
```

## VI. Đăng ký và cấu hình tài khoản github

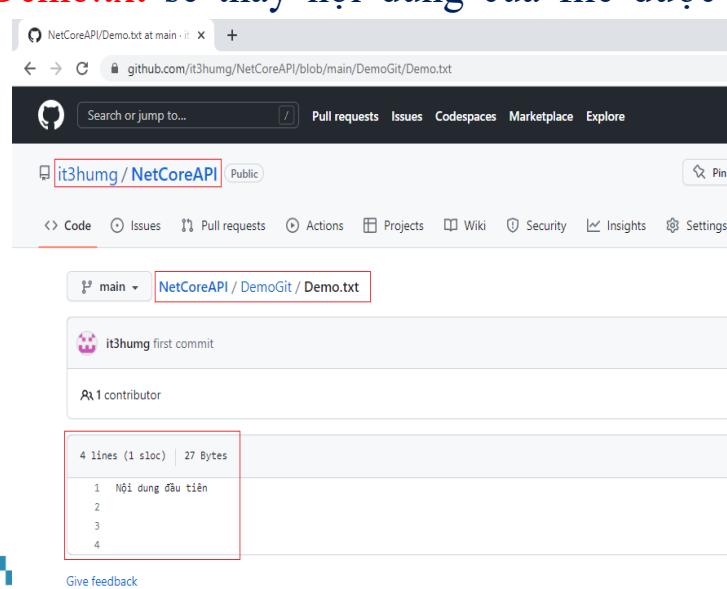
### 21. Truy cập vào github, bạn sẽ thấy:

- Thư mục DemoGit: chưa mã nguồn
- Link Remote: Sử dụng để quản lý mã nguồn (Đồng bộ mã nguồn mới, tải về mã nguồn...)



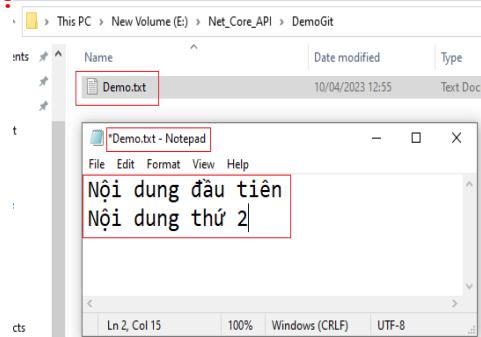
## VI. Đăng ký và cấu hình tài khoản github

### 22. Mở file DemoGit/Demo.txt sẽ thấy nội dung của file được đẩy lên



## VI. Đăng ký và cấu hình tài khoản github

22.Trên máy tính của bạn, Mở file DemoGit/Demo.txt bổ sung thêm nội dung “**Nội dung thứ 2**”



## VI. Đăng ký và cấu hình tài khoản github

22. Mở **CMD/Terminal** trong thư mục **Net\_Core\_API** và chạy 3 câu lệnh:

- git add .
- git commit -m Update\_Code
- git push

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19044.2728]
(c) Microsoft Corporation. All rights reserved.

E:\Net_Core_API>git add .
E:\Net_Core_API>git commit -m Update_Code
[main 2c17c14] Update_Code
 1 file changed, 1 insertion(+), 3 deletions(-)

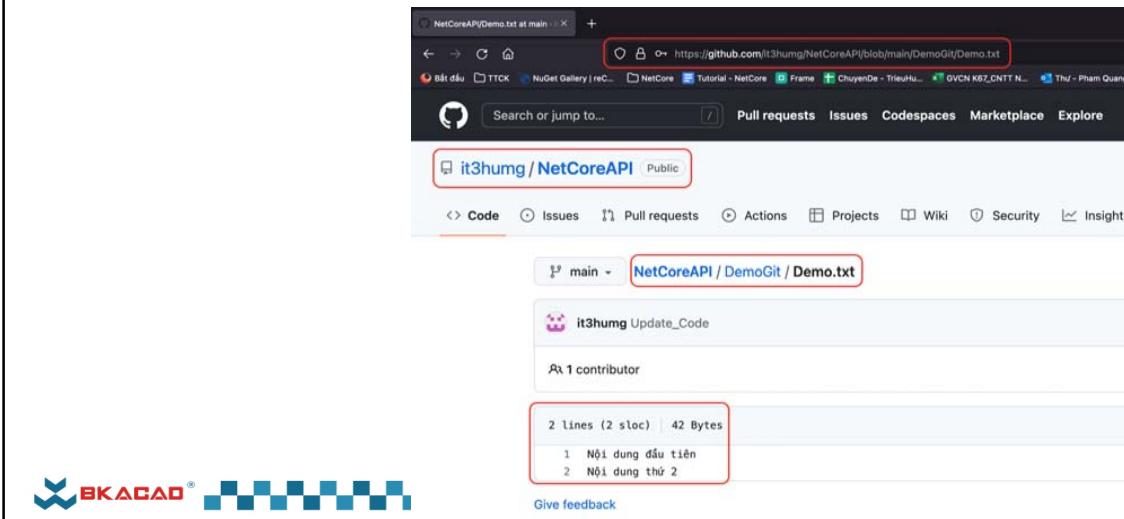
E:\Net_Core_API>git push
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Writing objects: 100% (4/4), 319 bytes | 319.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/it3humg/NetCoreAPI.git
 51313e6..2c17c14  main -> main

E:\Net_Core_API>
```



## VII. Đăng ký và cấu hình tài khoản github

23. Truy cập vào Github, mở **file DemoGit/Demo.txt** sẽ thấy nội dung của file được đẩy lên

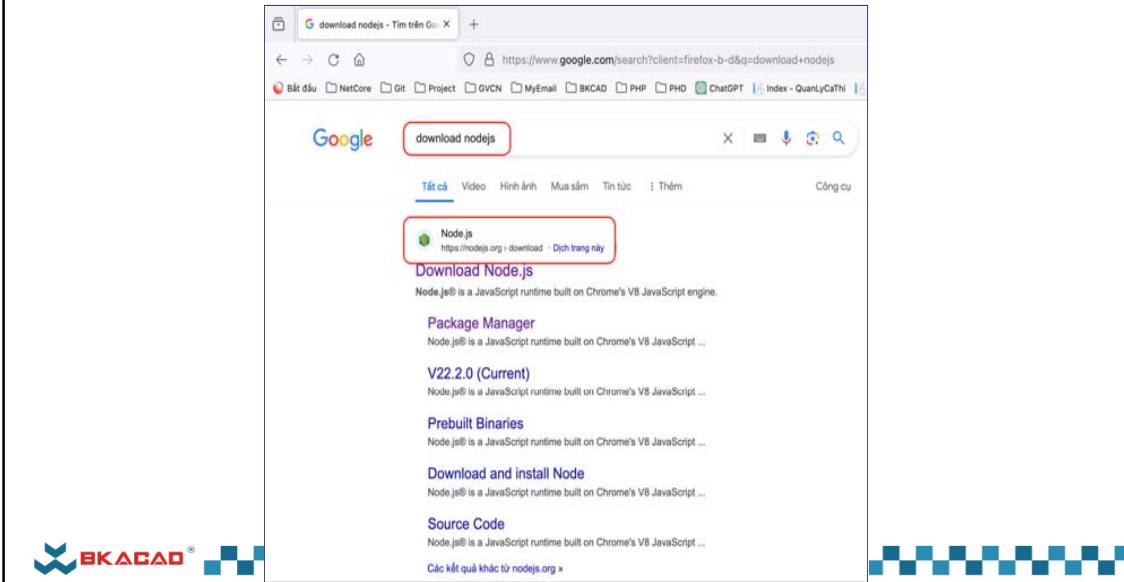


## VII. Cài đặt Nodejs

23. Truy cập vào Github, mở **file DemoGit/Demo.txt** sẽ thấy nội dung của file được đẩy lên

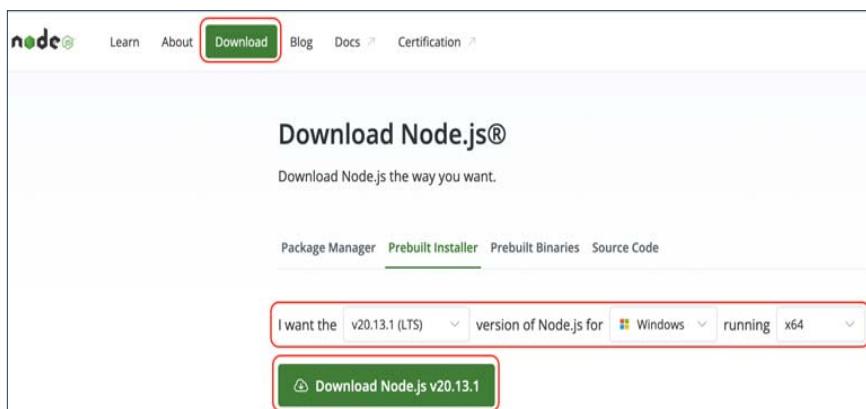
## VII. Cài đặt Nodejs

### 1. Truy cập google và tìm kiếm với từ khoá “download nodejs”



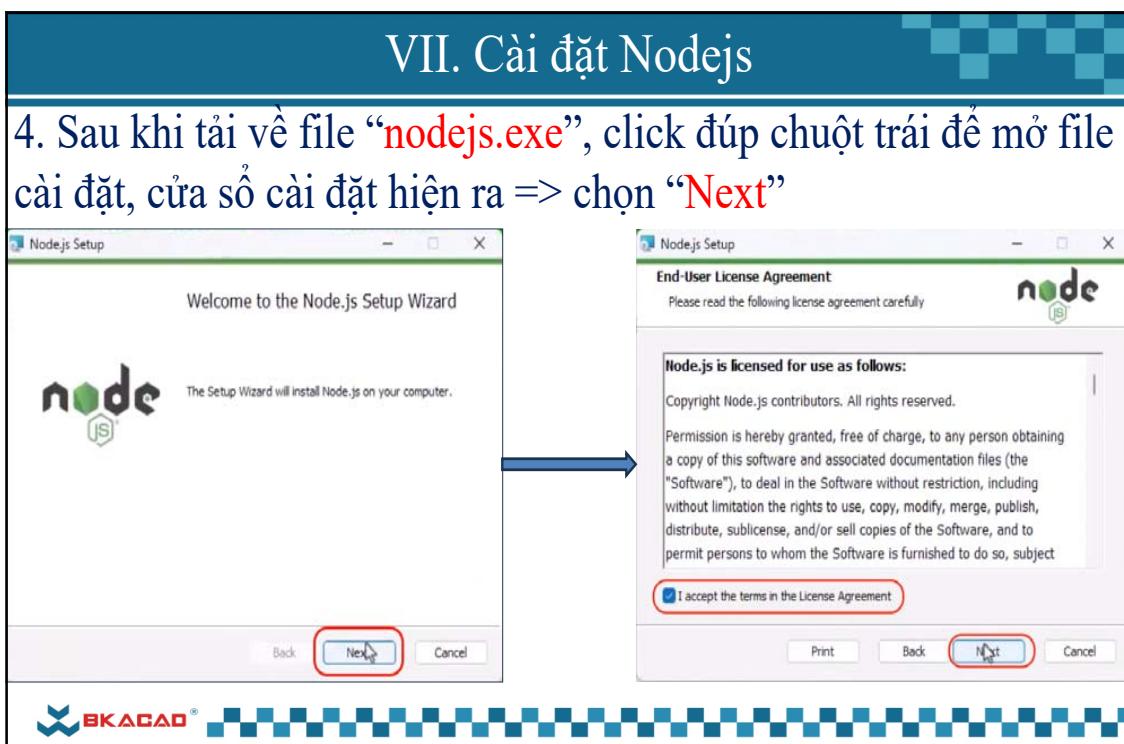
## VII. Cài đặt Nodejs

2. Truy cập vào link đầu tiên (<https://nodejs.org/en/>)
3. Chọn phiên bản phù hợp với hệ điều hành, để tải về.



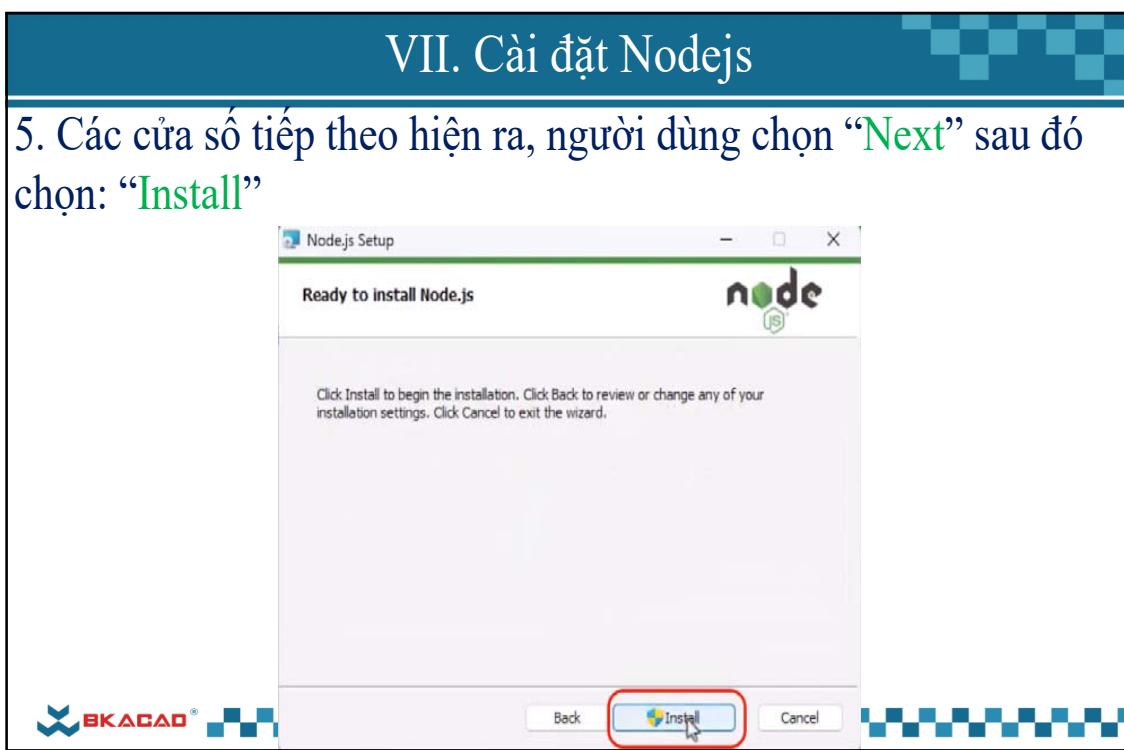
## VII. Cài đặt Nodejs

4. Sau khi tải về file “nodejs.exe”, click đúp chuột trái để mở file cài đặt, cửa sổ cài đặt hiện ra => chọn “Next”



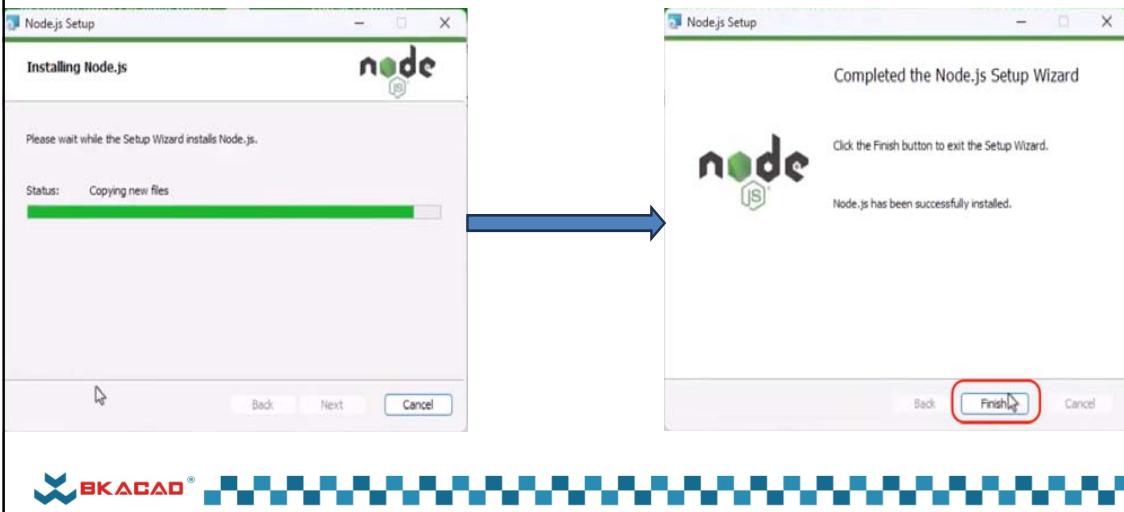
## VII. Cài đặt Nodejs

5. Các cửa sổ tiếp theo hiện ra, người dùng chọn “Next” sau đó chọn: “Install”



## VII. Cài đặt Nodejs

6. Quá trình cài đặt diện ra, sau đó chọn “Finish” để hoàn tất quá trình cài đặt



## VII. Cài đặt Nodejs

7. Mở CMD/Terminal và nhập lệnh “`node -v`” để kiểm tra xem cài đặt nodejs thành công chưa, nếu cài đặt thành công sẽ hiển thị phiên bản đã cài đặt.

The image shows a terminal window with several tabs: TERMINAL, PORTS, DEBUG CONSOLE, PROBLEMS (with a count of 3), GITLENS, and OUTPUT. The TERMINAL tab is active. The output in the terminal shows:

```
The default interactive shell is now zsh.  
To update your account to use zsh, please run `chsh -s /bin/zsh`.  
For more details, please visit https://support.apple.com/kb/HT208050.  
● (base) Phams-MacBook-Pro-2:Java phamquanghien$ node -v  
v20.13.1  
○ (base) Phams-MacBook-Pro-2:Java phamquanghien$ █
```

A red circle highlights the version number "v20.13.1" in the terminal output.

## BẢO MẬT VÀ TỐI ƯU ỨNG DỤNG WEB .NET



## HƯỚNG DẪN TẠO MỚI PROJECT TRONG .NET



## CÁC NỘI DUNG CHÍNH

1. Tạo project web MVC
2. Tạo project web API
3. Tạo project ReactJS
4. Đẩy code lên github



## 1. TẠO PROJECT WEB MVC

- B1: mở cmd/terminal sau đó cd tới thư mục muốn lưu project

```
phamquanghien ~ bash - 101x23
Last login: Sat May 18 13:44:45 on ttys001
The default interactive shell is now zsh.
(base) Phams-MacBook-Pro-2:~ phamquanghien$ cd /Volumes/Data/DataForwork/Project/Teach/BKaCad
```

- B2: chạy lệnh “dotnet new mvc --o VicemMVC” để tạo project

```
phamquanghien ~ bash - 86x18
Last login: Sat May 18 14:25:59 on ttys001
The default interactive shell is now zsh.
(base) Phams-MacBook-Pro-2:BKaCad phamquanghien$ dotnet new mvc --o VicemMVC
The template "ASP.NET Core Web App (Model-View-Controller)" was created successfully.
This template contains technologies from parties other than Microsoft, see https://aka.ms/aspnetcore/8.0-third-party-notices for details.

Processing post-creation actions...
Restoring /Volumes/Data/DataForwork/Project/Teach/BKaCad/VicemMVC/VicemMVC.csproj:
  Determining projects to restore...
  Restored /Volumes/Data/DataForwork/Project/Teach/BKaCad/VicemMVC/VicemMVC.csproj (in 779 ms).
Restore succeeded.

(base) Phams-MacBook-Pro-2:BKaCad phamquanghien$
```



## 2. TẠO PROJECT WEB API

- B1: mở cmd/terminal sau đó cd tới thư mục muốn lưu project

```
phamquanghien -- bash -- 101x23
Last login: Sat May 18 13:44:45 on ttys001
The default interactive shell is now zsh.
(base) Phams-MacBook-Pro-2:~ phamquanghien$ cd /Volumes/Data/DataForwork/Project/Teach/BKaCad
```

- B2: chạy lệnh “dotnet new webapi --use-controllers -o VicemAPI” để tạo project

```
(base) Phams-MacBook-Pro-2:BKaCad phamquanghien$ dotnet new webapi --use-controllers -o VicemAPI
The template "ASP.NET Core Web API" was created successfully.

Processing post-creation actions...
Restoring /Volumes/Data/DataForwork/Project/Teach/BKaCad/VicemAPI/VicemAPI.csproj:
  Determining projects to restore...
  Restored /Volumes/Data/DataForwork/Project/Teach/BKaCad/VicemAPI/VicemAPI.csproj (in 387 ms).
Restore succeeded.
```

(base) Phams-MacBook-Pro-2:BKaCad phamquanghien\$

## MỞ PROJECT TRÊN VISUAL STUDIO CODE

- Trên cmd/terminal chạy lệnh: “Code .” để tiến hành mở thư mục chứa mã nguồn của 2 project vừa tạo:

```
BKaCad -- bash -- 61x11
(base) Phams-MacBook-Pro-2:BKaCad phamquanghien$ code .
(base) Phams-MacBook-Pro-2:BKaCad phamquanghien$
```

The Explorer sidebar shows the following structure:

- VICEMAPI
  - bin
  - Controllers
  - obj
  - Properties
    - appsettings.Develop...
    - appsettings.json
  - Program.cs
  - VicemAPI.csproj
  - VicemAPI.http
  - WeatherForecast.cs
- VICEMMVC
  - bin
  - Controllers
  - Models
  - obj
  - Properties
  - Views
  - wwwroot
    - appsettings.Develop...
    - appsettings.json
  - Program.cs
  - VicemMVC.csproj

### 3. TẠO PROJECT REACTJS

- B1: mở cmd/terminal sau đó cd tới thư mục muốn lưu project

```
phamquanghien -- bash -- 101x23
Last login: Sat May 18 13:44:45 on ttys001
The default interactive shell is now zsh.
(base) Phams-MacBook-Pro-2:~ phamquanghien$ cd /Volumes/Data/DataForwork/Project/Teach/BKaCad
```

- B2: chạy lệnh “npx create-react-app vicem-reactjs” để tạo project

```
BKacad -- bash -- 90x24
(base) Phams-MacBook-Pro-2:BKacad phamquanghien$ npx create-react-app vicem-reactjs
Creating a new React app in /Volumes/Data/DataForwork/Project/Learn/BKacad/vicem-reactjs.

We suggest that you begin by typing:
  cd vicem-reactjs
  npm start

Happy hacking!
```



### MỞ PROJECT TRÊN VISUAL STUDIO CODE

- Trên cmd/terminal chạy lệnh: “Code .” để tiến hành mở thư mục chứa mã nguồn của 2 project vừa tạo:

```
BKacad -- bash -- 61x11
(base) Phams-MacBook-Pro-2:BKacad phamquanghien$ code .
(base) Phams-MacBook-Pro-2:BKacad phamquanghien$
```



### 3. TẠO PROJECT REACTJS

- B1: mở cmd/terminal sau đó cd tới thư mục muốn lưu project

```
phamquanghien -- bash -- 101x23
Last login: Sat May 18 13:44:45 on ttys001
The default interactive shell is now zsh.
(base) Phams-MacBook-Pro-2:~ phamquanghien$ cd /Volumes/Data/DataForwork/Project/Teach/BKaCad
```

- B2: chạy lệnh “npx create-react-app vicem-reactjs” để tạo project

```
BKaCad -- bash -- 90x24
(base) Phams-MacBook-Pro-2:BKaCad phamquanghien$ npx create-react-app vicem-reactjs
Creating a new React app in /Volumes/Data/DataForwork/Project/Teach/BKaCad/vicem-reactjs.

BKaCad -- bash -- 51x6
cd vicem-reactjs
npm start
Happy hacking!
(base) Phams-MacBook-Pro-2:BKaCad phamquanghien$
```

### 4. ĐẨY CODE LÊN GITHUB

- B1: Tạo repository VicemMVC trên github:

Dashboard

New

Top Repositories

New repository...

New repository

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Required fields are marked with an asterisk (\*).

Owner \* phamquanghien - Repository name \* VicemMVC

VicemMVC is available.

Great repository names are short and memorable. Need inspiration? How about turbo-octo-fiesta ?

Description (optional)

Public Anyone on the internet can see this repository. You choose who can commit.

Private You choose who can see and commit to this repository.

Initialize this repository with:

Add a README file This is where you can write a long description for your project. Learn more about READMEs.

Add .gitignore gitignore template: None Choose which files not to track from a list of templates. Learn more about ignoring files.

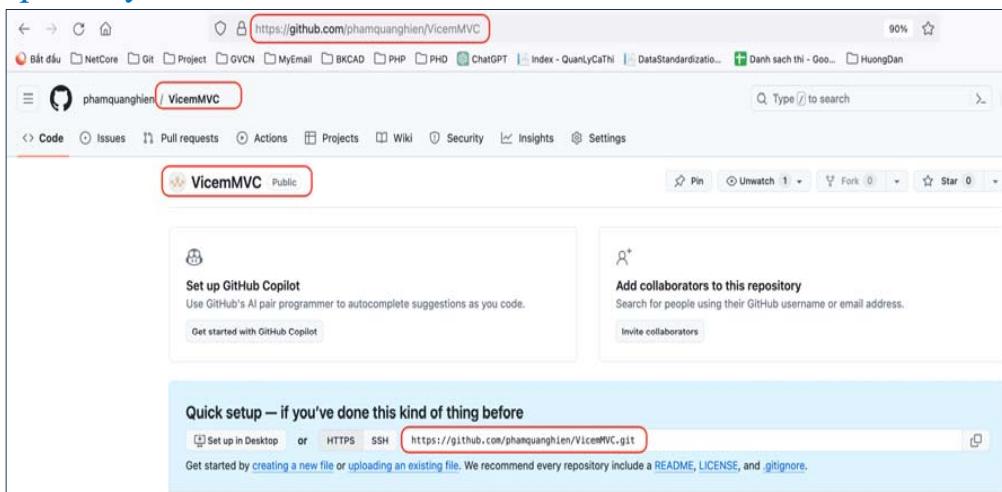
Choose a license License: None A license tells others what they can and can't do with your code. Learn more about licenses.

You are creating a public repository in your personal account.

Create repository

## 4. ĐẨY CODE LÊN GITHUB

- Repository VicemMVC được tạo ra:



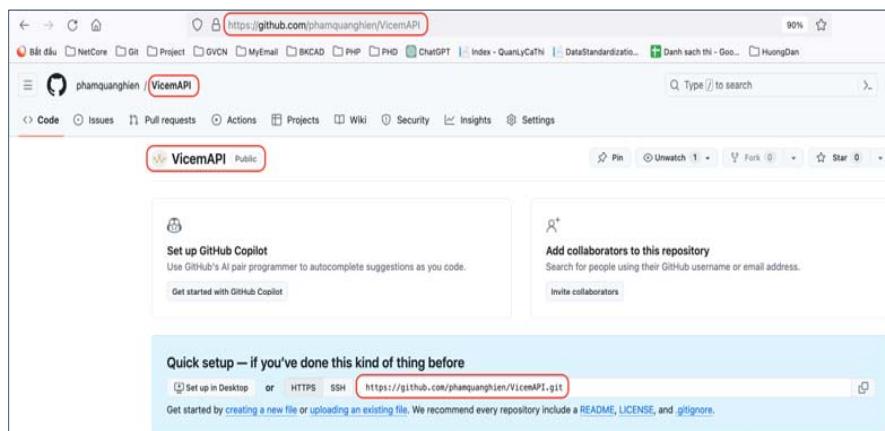
## 4. ĐẨY CODE LÊN GITHUB

- B1: Tạo repository VicemAPI trên github:

The image consists of two side-by-side screenshots of the GitHub interface. The left screenshot shows the main GitHub dashboard with a 'New' button highlighted by a red box. A large blue arrow points from this button to the right screenshot. The right screenshot shows the 'Create a new repository' form. In this form, the 'Repository name' field is filled with 'VicemAPI' and is also highlighted by a red box. Other fields in the form include 'Owner' (set to 'phamquanghien'), 'Public' (selected), and various initialization options like '.gitignore' and 'Choose a license'. The 'Create repository' button is located at the bottom right of the form.

## 4. ĐẨY CODE LÊN GITHUB

- Repository VicemAPI được tạo ra:



## 4. ĐẨY CODE LÊN GITHUB

Lưu ý: tuỳ theo cách quản lý mã nguồn có thể thực hiện theo một trong 1 hướng sau:

- ✓ Tạo 1 repository để quản lý mã nguồn của 2 dự án MVC và API
- ✓ Tạo 2 repository riêng biệt để quản lý mã nguồn của 2 dự án MVC và API

## 4. ĐÂY CODE LÊN GITHUB

- B2: Chạy các lệnh sau:

- ✓ git init
- ✓ git add .
- ✓ git commit -m "Create\_Project\_VicemMVC"

```
(base) Phams-MacBook-Pro-2:VicemMVC phamquanghien$ git init
Initialized empty Git repository in /Volumes/Data/DataForwork/Project/Teach/BKaCad/VicemMVC/.git/
(base) Phams-MacBook-Pro-2:VicemMVC phamquanghien$ git add .
(base) Phams-MacBook-Pro-2:VicemMVC phamquanghien$ git commit -m "Create_Project_VicemMVC"
[master (root-commit) 2f375dc] Create_Project_VicemMVC
 75 files changed, 74765 insertions(+)
```



## 4. ĐÂY CODE LÊN GITHUB

- B2: Chạy các lệnh sau:

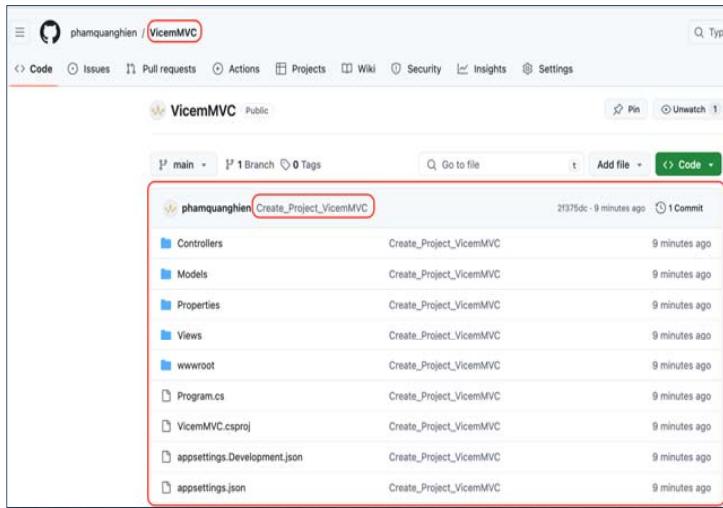
- ✓ git branch -M main
- ✓ git remote add origin <https://github.com/phamquanghien/VicemMVC.git>
- ✓ git push -u origin main

```
(base) Phams-MacBook-Pro-2:VicemMVC phamquanghien$ git branch -M main
(base) Phams-MacBook-Pro-2:VicemMVC phamquanghien$ git remote add origin https://github.com/phamquanghien/VicemMVC.git
(base) Phams-MacBook-Pro-2:VicemMVC phamquanghien$ git push -u origin main
Counting objects: 96, done.
Delta compression using up to 12 threads.
Compressing objects: 100% (98/98), done.
Writing objects: 100% (96/96), 910.36 KiB | 3.56 MiB/s, done.
Total 96 (delta 32), reused 0 (delta 0)
remote: Resolving deltas: 100% (32/32), done.
To https://github.com/phamquanghien/VicemMVC.git
 * [new branch]      main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.
(base) Phams-MacBook-Pro-2:VicemMVC phamquanghien$
```



## 4. ĐẨY CODE LÊN GITHUB

- B3: Truy cập vào github để xem mã nguồn đã đẩy lên thành công



## 4. ĐẨY CODE LÊN GITHUB

- Làm tương tự để đẩy mã nguồn dự án [VicemAPI](#) và [vicem-reactjs](#) lên github.
- Lưu ý từ lần đẩy code thứ 2 trở đi, người dùng chỉ cần thực hiện 3 lệnh sau (không cần thực hiện 6 lệnh như bước 1):
  - ✓ git add .
  - ✓ git commit -m “commit note”
  - ✓ git push





## BẢO MẬT VÀ TỐI ƯU ỨNG DỤNG WEB .NET



## XÁC THỰC VÀ PHÂN QUYỀN NGƯỜI DÙNG VỚI IDENTITY



## MỘT SỐ LỆNH CƠ BẢN

- Sinh mã Identity:

```
dotnet aspnet-codegenerator identity -dc YourNameSpace.YourContext -f
```

- Sinh mã Controller và View:

```
dotnet aspnet-codegenerator controller -name EmployeeController -m Employee -dc YourNameSpace.YourContext --relativeFolderPath Controllers  
--useDefaultLayout --referenceScriptLibraries --databaseProvider sqlite
```

- Tạo migrations : dotnet ef migrations add "Your\_Note"

- Cập nhật thay đổi vào CSDL: dotnet ef database update



## MỘT SỐ LỆNH CƠ BẢN

- Sinh mã API:

```
dotnet aspnet-codegenerator controller -name EmployeeController -async -api  
-m Employee -dc YourNameSpace.YourContext -outDir Controllers
```

- Add package:

```
dotnet tool uninstall --global dotnet-aspnet-codegenerator
```

```
dotnet tool install --global dotnet-aspnet-codegenerator
```

```
dotnet tool uninstall --global dotnet-ef
```

```
dotnet tool install --global dotnet-ef
```



## MỘT SỐ LỆNH CƠ BẢN

- Add package:

```
dotnet add package Microsoft.EntityFrameworkCore.Design
```

```
dotnet add package Microsoft.EntityFrameworkCore.SQLite
```

```
dotnet add package Microsoft.VisualStudio.Web.CodeGeneration.Design
```

```
dotnet add package Microsoft.EntityFrameworkCore.SqlServer
```

```
dotnet add package Microsoft.EntityFrameworkCore.Tools
```

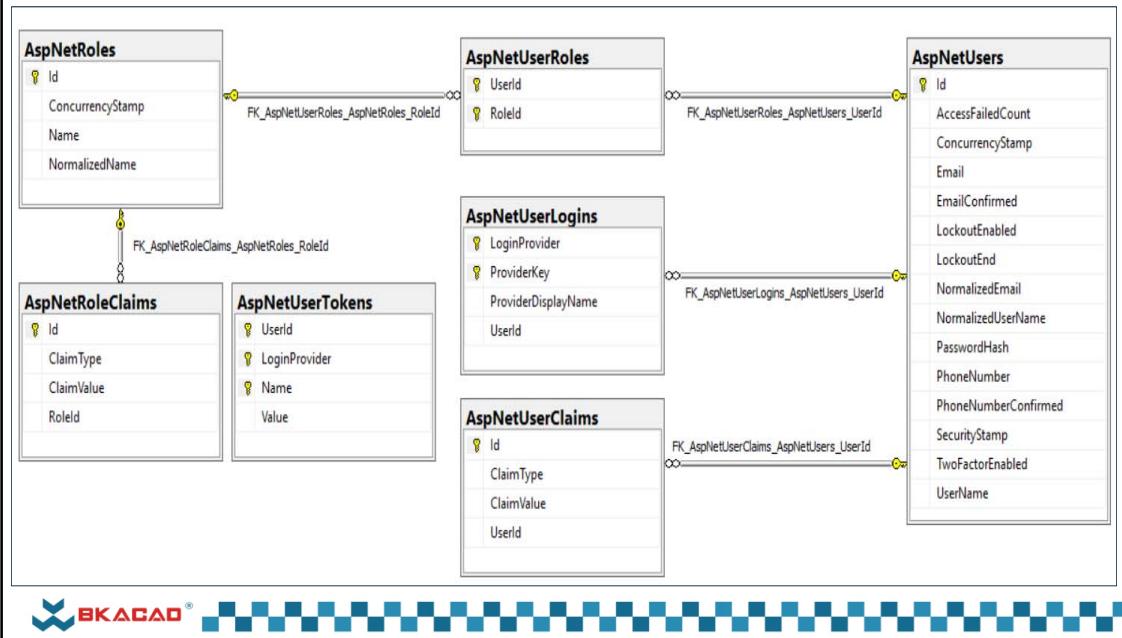


## CÁC NỘI DUNG CHÍNH

1. Giới thiệu về Identity
2. Xác thực người dùng
3. Phân quyền người dùng
4. Thực hành



# 1. GIỚI THIỆU VỀ IDENTITY



- # 1. GIỚI THIỆU VỀ IDENTITY
- **AspNetUsers**: Represents the user
  - **AspNetRoles**: Represents a role.
  - **AspNetUserClaims**: Represents a claim that a user possesses.
  - **AspNetUserTokens**: Represents an authentication token for a user.

## 1. GIỚI THIỆU VỀ IDENTITY

- **AspNetUserLogins:** Associates a user with a login.
- **AspNetRoleClaims:** Represents a claim that's granted to all users within a role.
- **AspNetUserRoles:** A join entity that associates users and roles.



## 2. XÁC THỰC NGƯỜI DÙNG

- Tích hợp Identity vào dự án
- Sinh mã cho các chức năng Login, Logout, Register
- Tuỳ chỉnh Identity
- Một số cấu hình trong Identity
- Xác nhận tài khoản và đặt lại mật khẩu trong Identity



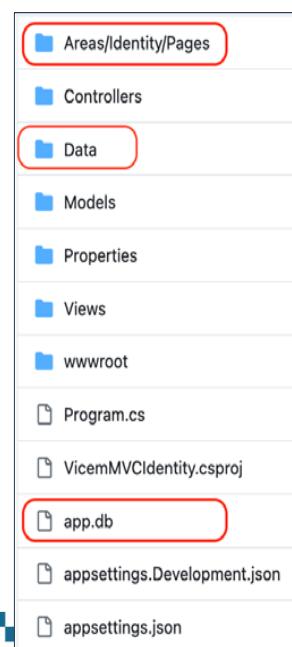
## TÍCH HỢP IDENTITY VÀO DỰ ÁN MVC

- **Cách 1:** Tạo project .Net MVC có tích hợp sẵn Identity.
- **Cách 2:** Tích hợp Identity vào project hiện tại.

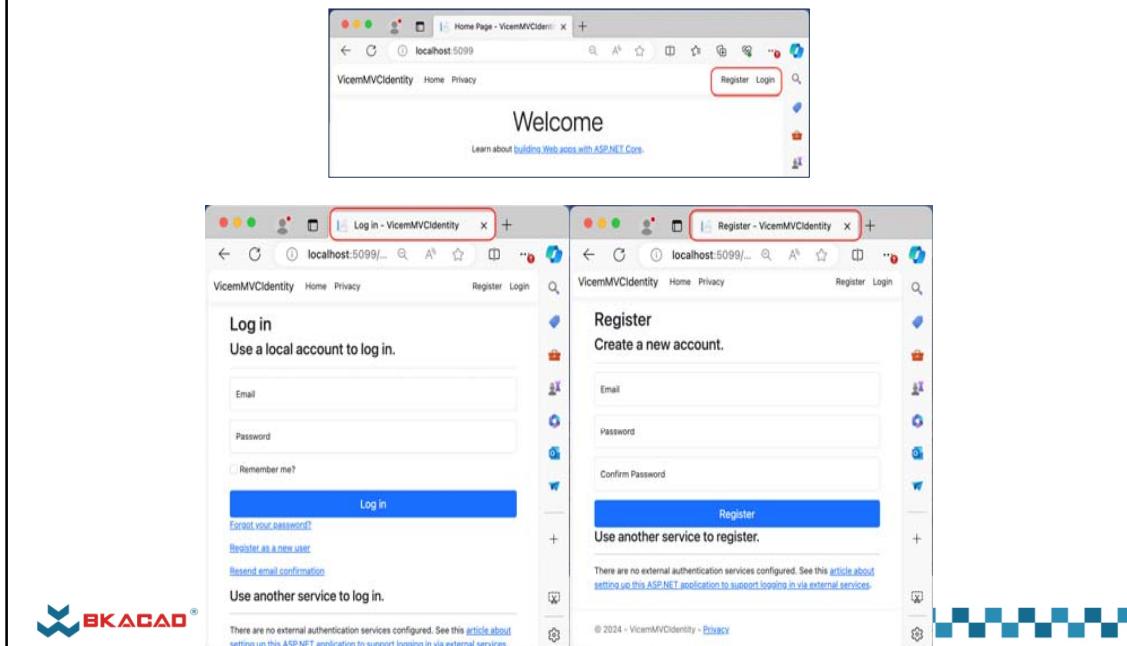


### Tạo Project MVC có tích hợp sẵn Identity

- Để tạo project .Net MVC có tích hợp sẵn Identity cần thực thi câu lệnh sau : “dotnet new mvc --auth Individual -o VicemMVCIdentity”



## Tạo Project MVC có tích hợp sẵn Identity



## Tích hợp Identity vào dự án hiện tại

- Để tích hợp Identity vào project hiện có cần chạy lệnh để add các package:  
`dotnet add package Microsoft.AspNetCore.Identity.EntityFrameworkCore`  
`dotnet add package Microsoft.AspNetCore.Identity.UI`



## Add các package cần thiết vào dự án

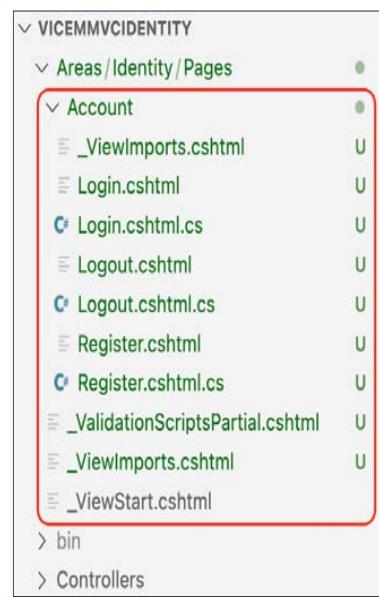
Add các package cần thiết cho project:

- dotnet add package Microsoft.VisualStudio.Web.CodeGeneration.Design
- dotnet add package Microsoft.EntityFrameworkCore.Design



## SINH MÃ CHO CÁC CHỨC NĂNG Login, Logout, Register

```
dotnet aspnet-codegenerator identity -dc  
VicemMVCIdentity.Data.ApplicationDbContext --files  
"Account.Register;Account.Login;Account.Logout"
```



## TUỲ CHỈNH IDENTITY

- Mục đích: thêm mới thuộc tính **FullName** vào class **AspNetUsers**
- Trong thư mục **Models/Entities** tạo class  **ApplicationUser** như sau:

```
EXPLORER ... C:\VicemMVCIdentity\ 1, U X
...
Models > Entities > C:\ ApplicationUser.cs ...
1  using Microsoft.AspNetCore.Identity;
2  ...
3  namespace VicemMVCIdentity.Models
4  {
5      0 references
6      public class ApplicationUser : IdentityUser
7      {
8          [PersonalData]
9          0 references
10         public string FullName { get; set; }
11     }
12 }
13 
```



## TUỲ CHỈNH IDENTITY

- Trong file **Program.cs** sửa lại code như sau:

```
14 builder.Services.AddDefaultIdentity<ApplicationUser>(options => options.SignIn.RequireConfirmedAccount = true)
15     .AddEntityFrameworkStores<ApplicationContext>();
16 builder.Services.AddControllersWithViews();
```

- Trong file **Data/ApplicationDbContext.cs** sửa lại code như sau:

```
7  public class ApplicationContext : IdentityDbContext<ApplicationUser>
8  {
9      0 references
10     public ApplicationContext(DbContextOptions<ApplicationContext> options)
11     {
12     }
13 }
```



## TUỲ CHỈNH IDENTITY

- Mở CMD/Terminal và chạy lệnh: dotnet ef migrations add "Add\_Column\_AspNetUsers\_FullName"

```
TERMINAL PORTS DEBUG CONSOLE PROBLEMS 4 GITLENS OUTPUT
● (base) Phams-MacBook-Pro-2:VicemMVCIdentity phamquanghien$ dotnet ef migrations add "Add_Column_AspNetUsers_FullName"
Build started...
Build succeeded.
The Entity Framework tools version '8.0.0' is older than that of the runtime '8.0.5'. Update the tools for the latest
nd bug fixes. See https://aka.ms/AAc1fbw for more information.
Done. To undo this action, use 'ef migrations remove'
```

- Sau đó chạy lệnh “dotnet ef database update”

```
TERMINAL PORTS DEBUG CONSOLE PROBLEMS 4 GITLENS OUTPUT
● (base) Phams-MacBook-Pro-2:VicemMVCIdentity phamquanghien$ dotnet ef database update
Build started...
Build succeeded.
```



## TUỲ CHỈNH IDENTITY

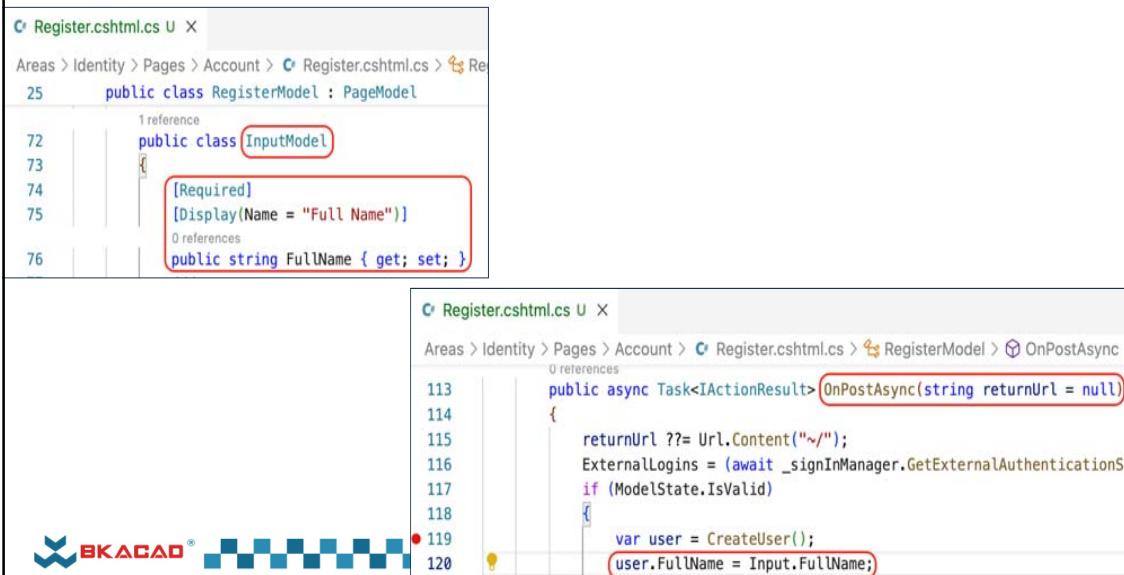
- Sinh lại mã nguồn cho các chức năng Login, Logout, Register (thêm -f ở cuối câu lệnh sinh mã).
- Trong file Views/Shared/\_LoginPartial.cshtml sửa lại code như sau:

```
_LoginPartial.cshtml M ×
Views > Shared > _LoginPartial.cshtml
You, 1 second ago | 1 author (You)
1 @using Microsoft.AspNetCore.Identity
2 @using VicemMVCIdentity.Models
3 @inject SignInManager<ApplicationUser> SignInManager
4 @inject UserManager<ApplicationUser> UserManager
```



## TUỲ CHỈNH IDENTITY

- Trong file `Areas/Identity/Pages/Account/Register.cshtml.cs` sửa lại code như sau:

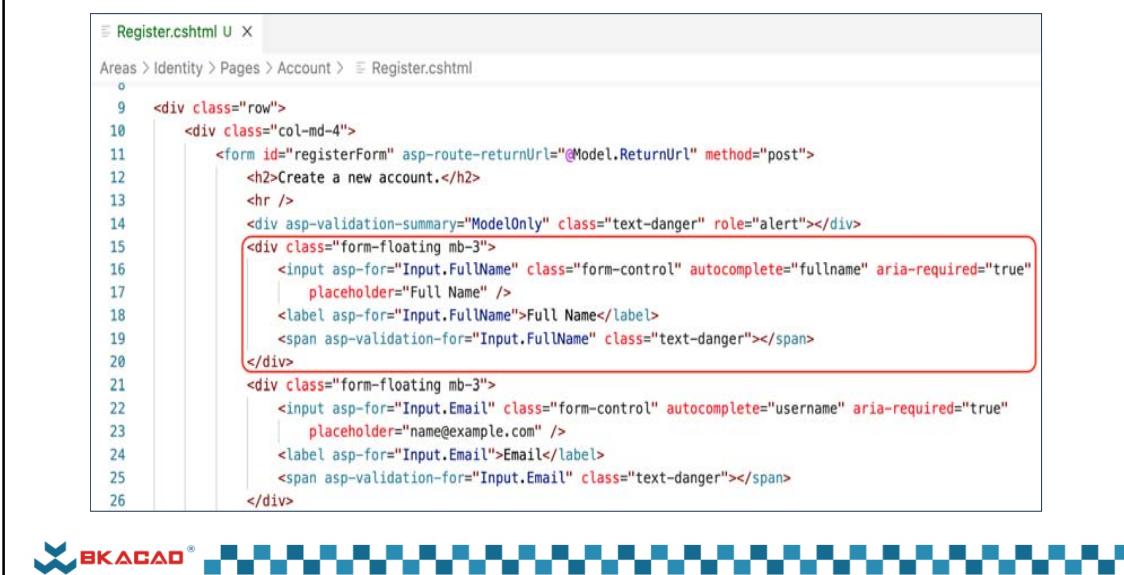


```
Register.cshtml.cs
...
25     public class RegisterModel : PageModel
26     {
27         public class InputModel
28         {
29             [Required]
30             [Display(Name = "Full Name")]
31             public string FullName { get; set; }
32         }
33     }
34 }

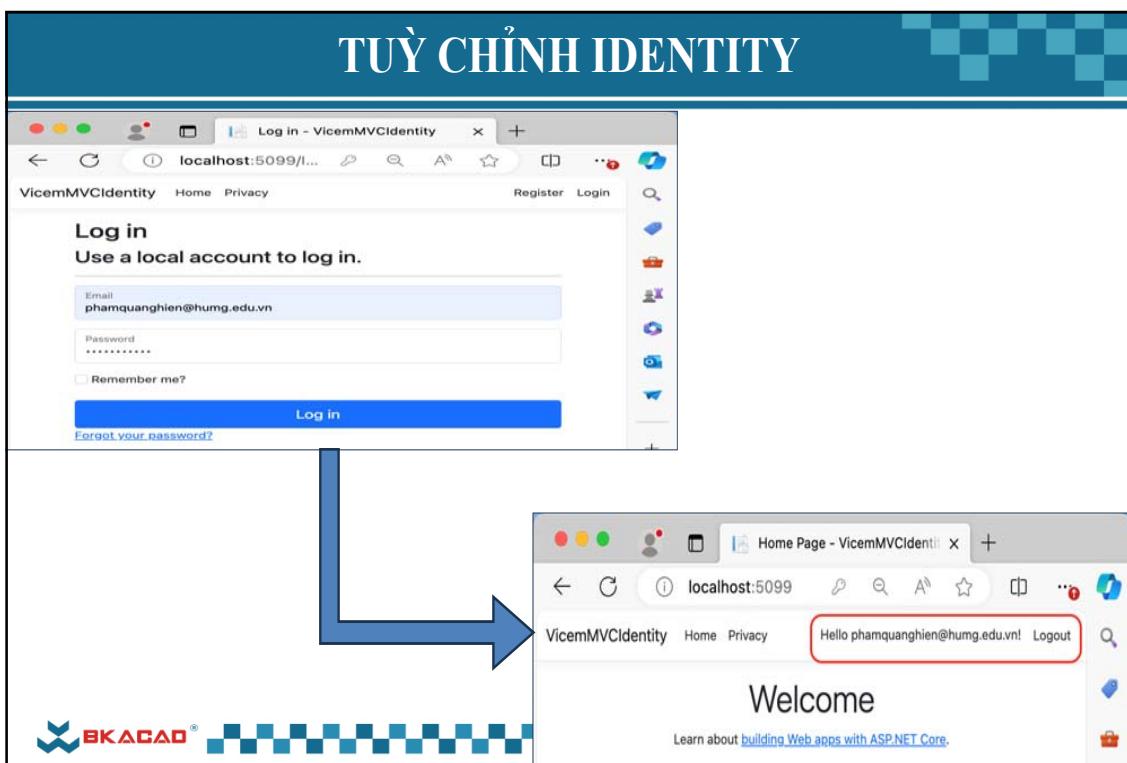
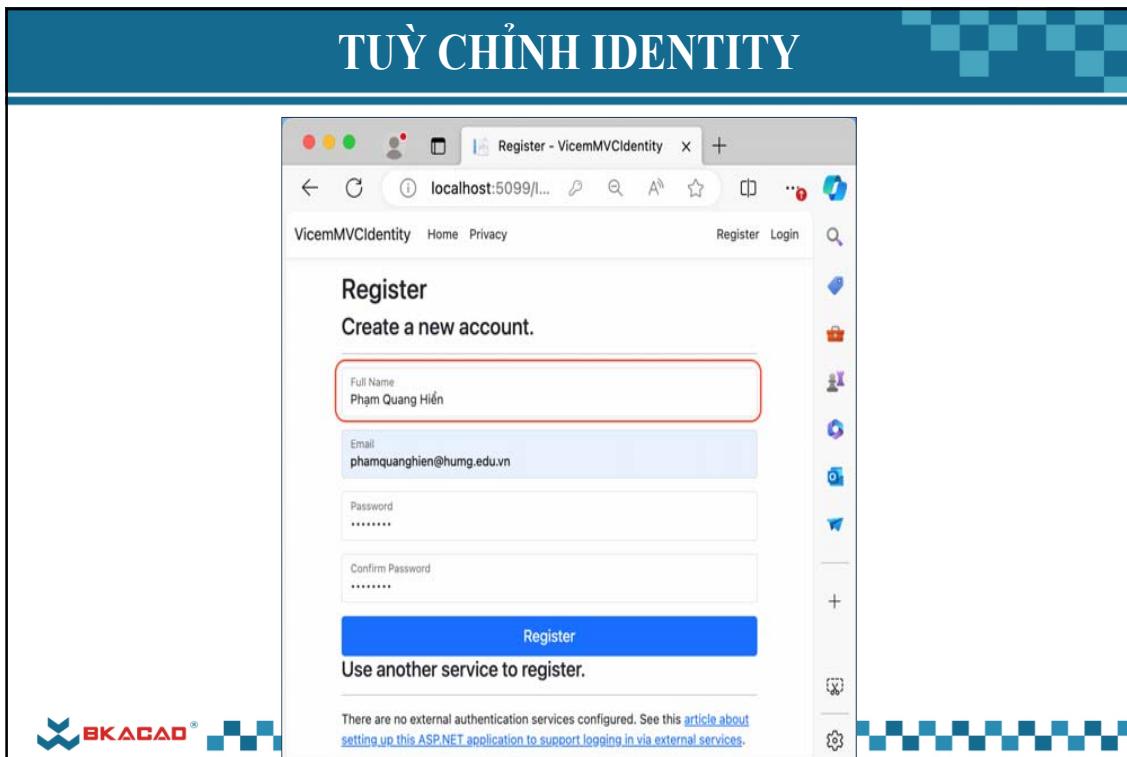
Register.cshtml.cs
...
113     public async Task<IActionResult> OnPostAsync(string returnUrl = null)
114     {
115         returnUrl ??= Url.Content("~/");
116         ExternalLogins = await _signInManager.GetExternalAuthenticationS
117         if (ModelState.IsValid)
118         {
119             var user = CreateUser();
120             user.FullName = Input.FullName;
121         }
122     }
123 }
```

## TUỲ CHỈNH IDENTITY

- Trong file `Areas/Identity/Pages/Account/Register.cshtml` sửa lại code như sau:



```
Register.cshtml
...
9     <div class="row">
10        <div class="col-md-4">
11            <form id="registerForm" asp-route-returnUrl="@Model.ReturnUrl" method="post">
12                <h2>Create a new account.</h2>
13                <hr />
14                <div asp-validation-summary="ModelOnly" class="text-danger" role="alert"></div>
15                <div class="form-floating mb-3">
16                    <input asp-for="Input.FullName" class="form-control" autocomplete="fullname" aria-required="true"
17                           placeholder="Full Name" />
18                    <label asp-for="Input.FullName">Full Name</label>
19                    <span asp-validation-for="Input.FullName" class="text-danger"></span>
20                </div>
21                <div class="form-floating mb-3">
22                    <input asp-for="Input.Email" class="form-control" autocomplete="username" aria-required="true"
23                           placeholder="name@example.com" />
24                    <label asp-for="Input.Email">Email</label>
25                    <span asp-validation-for="Input.Email" class="text-danger"></span>
26                </div>
27            </form>
28        </div>
29    </div>
```



## TUỲ CHỈNH IDENTITY

### Lưu ý:

- Người dùng có thể tùy chỉnh thêm các thuộc tính vào các Entity có sẵn của Identity như trên để phù hợp với dự án của mình.
- Người dùng có thể đổi tên các Entity trong Identity như sau (Code trong file ApplicationDbContext):

```
protected override void OnModelCreating(ModelBuilder builder)
{
    base.OnModelCreating(builder);
    builder.Entity<ApplicationUser>().ToTable("Users");
    builder.Entity<IdentityRole>().ToTable("Roles");
    builder.Entity<IdentityUserRole<string>>().ToTable("UserRoles");
    builder.Entity<IdentityUserClaim<string>>().ToTable("UserClaims");
    builder.Entity<IdentityUserLogin<string>>().ToTable("UserLogins");
    builder.Entity<IdentityUserToken<string>>().ToTable("UserTokens");
    builder.Entity<IdentityRoleClaim<string>>().ToTable("RoleClaims");
    builder.Entity<IdentityUserClaim<string>>().ToTable("UserClaims");
}
```



## MỘT SỐ CẤU HÌNH TRONG IDENTITY

Cấu hình khoá tài khoản khi đăng nhập sai nhiều lần:

- File Program.cs:

```
builder.Services.AddControllersWithViews();
builder.Services.Configure<IdentityOptions>(options =>
{
    // Default Lockout settings.
    options.Lockout.DefaultLockoutTimeSpan = TimeSpan.FromMinutes(5);
    options.Lockout.MaxFailedAccessAttempts = 5;
    options.Lockout.AllowedForNewUsers = true;
});
```

- File Areas/Identity/Pages/Account/Login.cshtml: đặt giá trị lockoutOnFailure: true

```
ExternalLogins = (await _signInManager.GetExternalAuthenticationSchemesAsync()).ToList();

if (ModelState.IsValid)
{
    // This doesn't count login failures towards account lockout
    // To enable password failures to trigger account lockout, set lockoutOnFailure: true
    var result = await _signInManager.PasswordSignInAsync(Input.Email, Input.Password, Input.RememberMe, lockoutOnFailure: true);
```



## MỘT SỐ CẤU HÌNH TRONG IDENTITY

- Cấu hình mật khẩu của tài khoản (File Program.cs):

```
17 builder.Services.Configure<IdentityOptions>(options =>
18 {
19     // Default Lockout settings.
20     options.Lockout.DefaultLockoutTimeSpan = TimeSpan.FromMinutes(5);
21     options.Lockout.MaxFailedAccessAttempts = 5;
22     options.Lockout.AllowedForNewUsers = true;
23     //Config Password
24     options.Password.RequireDigit = true;
25     options.Password.RequiredLength = 8;
26     options.Password.RequireNonAlphanumeric = false;
27     options.Password.RequireUppercase = true;
28     options.Password.RequireLowercase = false;
29});
```



## MỘT SỐ CẤU HÌNH TRONG IDENTITY

- Cấu hình xác thực tài khoản (File Program.cs):

```
//Config Login
options.SignIn.RequireConfirmedEmail = false;
options.SignIn.RequireConfirmedPhoneNumber = false;
```

- Cấu hình tài khoản người dùng (File Program.cs):

```
//Config User
options.User.RequireUniqueEmail = true;
```



## MỘT SỐ CÂU HÌNH TRONG IDENTITY

- Cấu hình Cookie (File Program.cs):

```
38 builder.Services.ConfigureApplicationCookie(options =>
39 {
40     options.Cookie.HttpOnly = true;
41     options.ExpireTimeSpan = TimeSpan.FromMinutes(60);
42     options.LoginPath = "/Account/Login";
43     options.AccessDeniedPath = "/Account/AccessDenied";
44     options SlidingExpiration = true;
45});
```



## MỘT SỐ CÂU HÌNH TRONG IDENTITY

- Cấu hình Cookie (File Program.cs):

```
38 builder.Services.ConfigureApplicationCookie(options =>
39 {
40     options.Cookie.HttpOnly = true;
41     options.ExpireTimeSpan = TimeSpan.FromMinutes(60);
42     options.LoginPath = "/Account/Login";
43     options.AccessDeniedPath = "/Account/AccessDenied";
44     options SlidingExpiration = true;
45});
```



## MỘT SỐ CẤU HÌNH TRONG IDENTITY

- Cấu hình Cookie (File Program.cs):

```
39 builder.Services.ConfigureApplicationCookie(options =>
40 {
41     options.Cookie.HttpOnly = true;
42     //chi gui Cooke qua HTTPS
43     options.Cookie.SecurePolicy = CookieSecurePolicy.Always;
44     //Giam thieu rui ro CSRF
45     options.Cookie.SameSite = SameSiteMode.Lax;
46     options.ExpireTimeSpan = TimeSpan.FromMinutes(60);
47     options.LoginPath = "/Account/Login";
48     options.AccessDeniedPath = "/Account/AccessDenied";
49     options SlidingExpiration = true;
50 });
```



## MỘT SỐ CẤU HÌNH TRONG IDENTITY

- Cấu hình bảo vệ dữ liệu (File Program.cs):

```
51 builder.Services.AddDataProtection()
52     //chi dinh thu muc luu tru khoa bao ve du lieu
53     .PersistKeysToFileSystem(new DirectoryInfo("./keys"))
54     //xac dinh ten ung dung su dung dich vu bao ve du lieu
55     .SetApplicationName("YourAppName")
56     //dat thoi gian so cho khoa bao mat du lieu
57     .SetDefaultKeyLifetime(TimeSpan.FromDays(14));
```



## XÁC NHẬN TÀI KHOẢN VÀ ĐẶT LẠI MẬT KHẨU

- Sinh bô sung mã nguồn cho các chức năng
- Cài đặt và cấu hình bên Gmail
- Thêm các gói cần thiết cho dự án
- Cài đặt và cấu hình trên dự án MVC
- Đăng ký và xác nhận tài khoản
- Đặt lại mật khẩu



## Sinh bô sung mã nguồn cho các chức năng

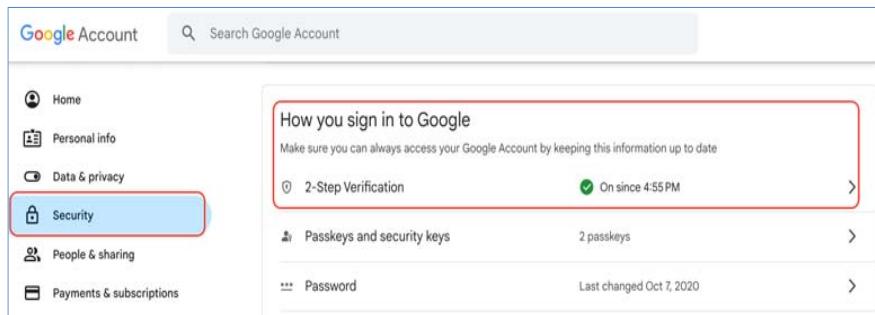
```
dotnet aspnet-codegenerator identity -dc  
VicemMVCIdentity.Data.ApplicationDbContext -f
```

VICEMMVCIDENTITY	
	•
Areas/Identity/Pages	•
Account	•
Manage	•
_StatusMessage.cshtml	U
_ViewImports.cshtml	
AccessDenied.cshtml	U
AccessDenied.cshtml.cs	U
ConfirmEmail.cshtml	U
ConfirmEmail.cshtml.cs	U
ConfirmEmailChange.cshtml	U
ConfirmEmailChange.cshtml.cs	U
ExternalLogin.cshtml	U
ExternalLogin.cshtml.cs	U
ForgotPassword.cshtml	U
ForgotPassword.cshtml.cs	U
ForgotPasswordConfirmation.cs...	U
ForgotPasswordConfirmation.cs...	U
Lockout.cshtml	U
Lockout.cshtml.cs	U
Login.cshtml	



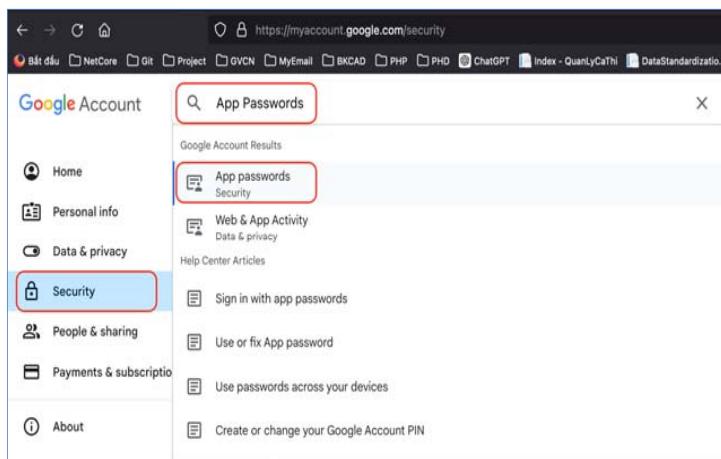
## Cài đặt và cấu hình trên Gmail

- Đăng nhập vào tài khoản Email, sau đó truy cập vào đường dẫn <https://myaccount.google.com/security>, sau đó mở chức năng xác thực 2 bước:



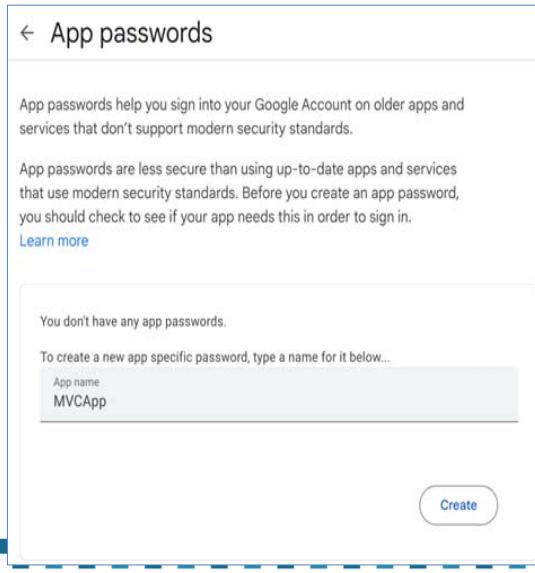
## Cài đặt và cấu hình trên Gmail

- Tìm kiếm với từ khoá App Passwords sau đó mở App Passwords để thực hiện cấu hình



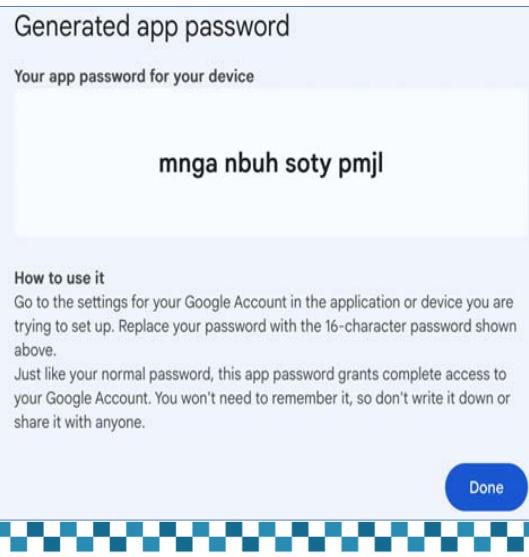
## Cài đặt và cấu hình trên Gmail

- Đặt một tên bất kỳ (ví dụ **MVCApp**) và chọn **Create**:



## Cài đặt và cấu hình trên Gmail

- app password được khởi tạo thành công, sử dụng app password để cấu hình trong **appsettings.json**:



## Thêm các gói cần thiết cho dự án

- ✓ dotnet add package Microsoft.AspNetCore.Identity.UI
- ✓ dotnet add package Microsoft.AspNetCore.Identity.EntityFrameworkCore
- ✓ dotnet add package MailKit



## Cài đặt và cấu hình trên dự án MVC

- Tạo class `Models/Process/MailSettings.cs` và viết code
- Tạo class `Models/Process/SendMailService.cs` và viết code như sau: [Tham khảo mã nguồn ở đây](#)
- Sửa mã nguồn file `appsettings.json`
- Sửa mã nguồn file `Program.cs`



## Models/Process/MailSettings.cs

```
1  namespace VicemMVCIdentity.Models.Process
2  {
3      public class MailSettings
4      {
5          public string Mail { get; set; }
6          public string DisplayName { get; set; }
7          public string Password { get; set; }
8          public string Host { get; set; }
9          public int Port { get; set; }
10     }
11 }
```



## Models/Process/SendMailService.cs

```
1  using MailKit.Net.Smtp;
2  using MimeKit;
3  using Microsoft.AspNetCore.Identity.UI.Services;
4  using Microsoft.Extensions.Options;
5  using MailKit.Security;
6  namespace VicemMVCIdentity.Models.Process
7  {
8      public class SendMailService : IEmailSender
9      {
10         private readonly MailSettings mailSettings;
11         private readonly ILogger<SendMailService> logger;
12         public SendMailService(IOptions<MailSettings> _mailSettings, ILogger<SendMailService> _logger)
13         {
14             mailSettings = _mailSettings.Value;
15             logger = _logger;
16             logger.LogInformation("Create SendMailService");
17         }
18         public async Task SendEmailAsync(string email, string subject, string htmlMessage) ->
19     }
48 }
49 }
```



## Models/Process/SendMailService.cs

```
public async Task SendEmailAsync(string email, string subject, string htmlMessage)
{
    var message = new MimeMessage();
    message.Sender = new MailboxAddress(mailSettings.DisplayName, mailSettings.Mail);
    message.From.Add(new MailboxAddress(mailSettings.DisplayName, mailSettings.Mail));
    message.To.Add(MailboxAddress.Parse(email));
    message.Subject = subject;
    var builder = new BodyBuilder();
    builder.HtmlBody = htmlMessage;
    message.Body = builder.ToMessageBody();
    using var smtp = new MailKit.Net.Smtp.SmtpClient();
    try
    {
        smtp.Connect(mailSettings.Host, mailSettings.Port, SecureSocketOptions.StartTls);
        smtp.Authenticate(mailSettings.Mail, mailSettings.Password);
        await smtp.SendAsync(message);
    }
    catch (Exception ex)
    {
        // Gửi mail thất bại, nội dung email sẽ lưu vào thư mục mailsSave
        System.IO.Directory.CreateDirectory("mailsSave");
        var emailSaveFile = string.Format("mailsSave/{0}.eml", Guid.NewGuid());
        await message.WriteToAsync(emailSaveFile);

        logger.LogInformation("Lỗi gửi mail, lưu tại - " + emailSaveFile);
        logger.LogError(ex.Message);
    }
    smtp.Disconnect(true);
    logger.LogInformation("send mail to: " + email);
}
```



## appsettings.json

```
{} appsettings.json M X
{} appsettings.json > ...
11   "AllowedHosts": "*",
12   "MailSettings": {
13     "Mail": "Your Email",
14     "DisplayName": "Administrator",
15     "Password": "Your App Password",
16     "Host": "smtp.gmail.com",
17     "Port": 587
18   }
19 }
```



## Program.cs

```
7 var builder = WebApplication.CreateBuilder(args);
8 builder.Services.AddOptions();
9 var mailSettings = builder.Configuration.GetSection("MailSettings");
10 builder.Services.Configure<MailSettings>(mailSettings);
11 builder.Services.AddTransient<IEmailSender, SendMailService>();
12
```



## Đăng ký và xác nhận tài khoản

- Nhập thông tin vào form để đăng ký tài khoản với hệ thống.

### Register

Create a new account.

Full Name  
Pham Quang Hien

Email  
pqh.cntt@gmail.com

Password  
\*\*\*\*\*

Confirm Password  
\*\*\*\*\*

Register

VicemMVIdentity Home Privacy

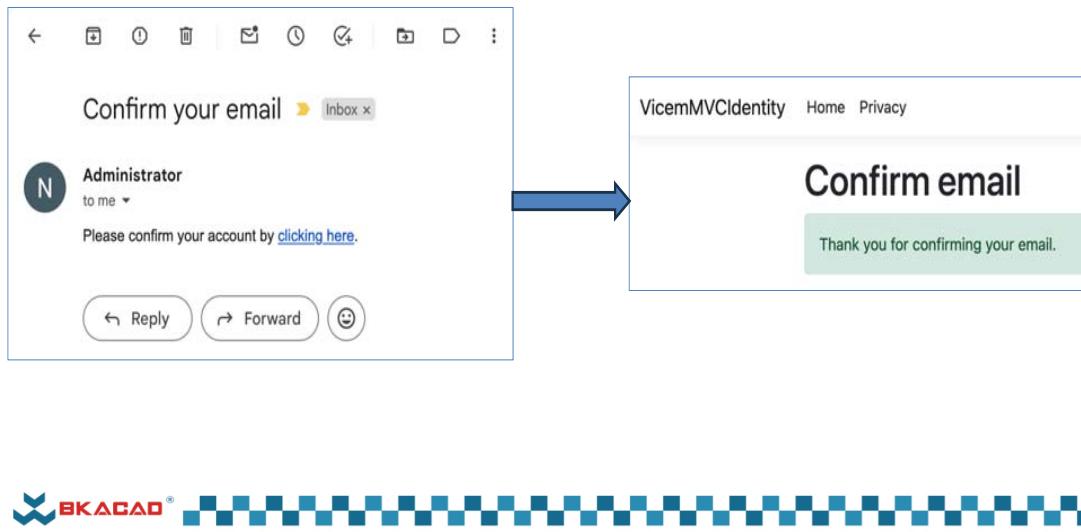
### Register confirmation

Please check your email to confirm your account.



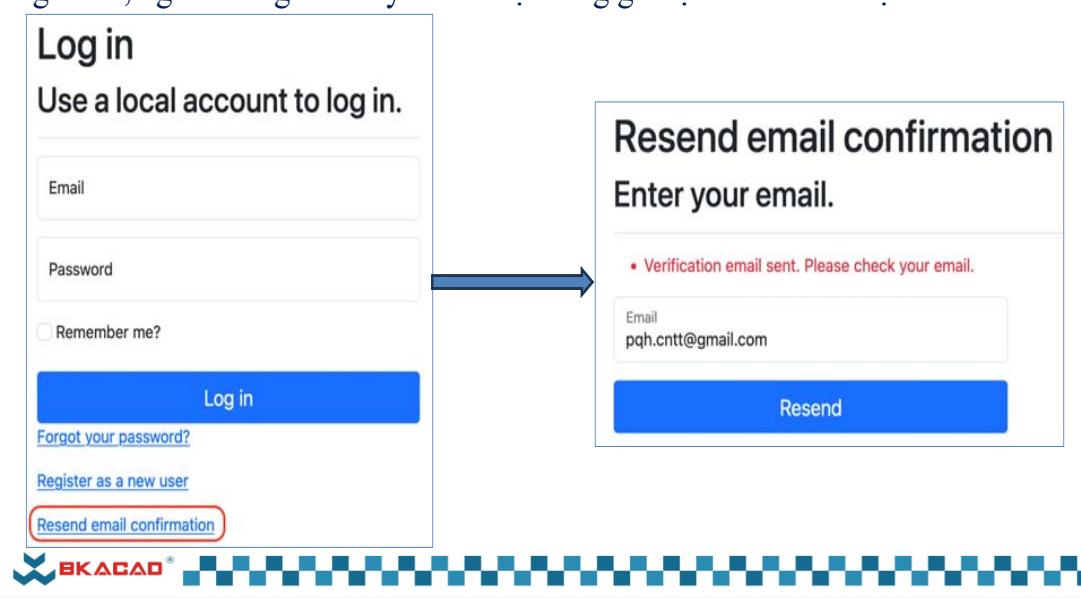
## Đăng ký và xác nhận tài khoản

- Bấm vào link trong email nhận được để xác nhận tài khoản



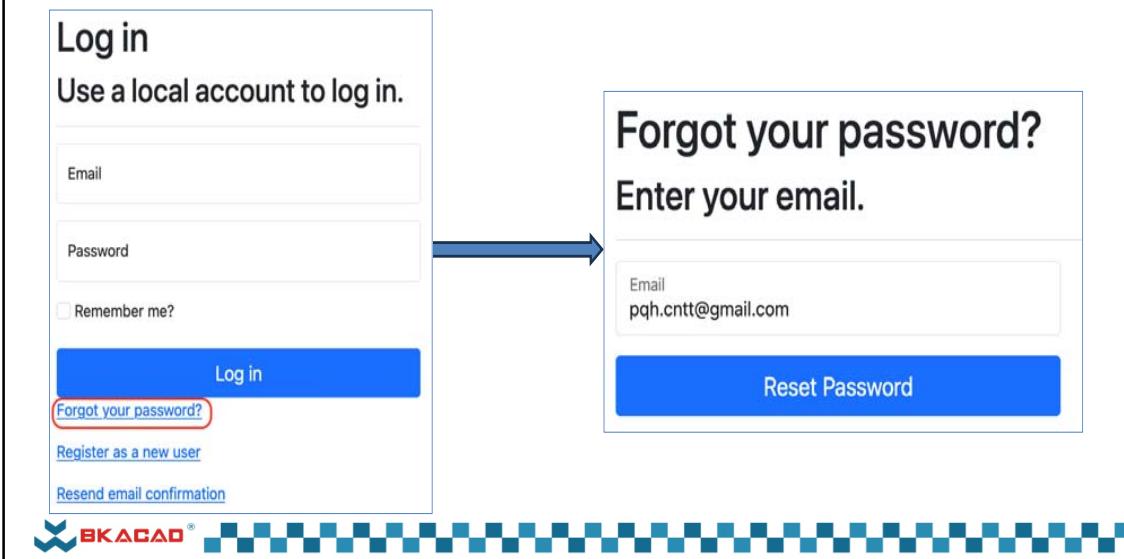
## Đăng ký và xác nhận tài khoản

- Ngoài ra, người dùng có thể yêu cầu hệ thống gửi lại email xác nhận tài khoản



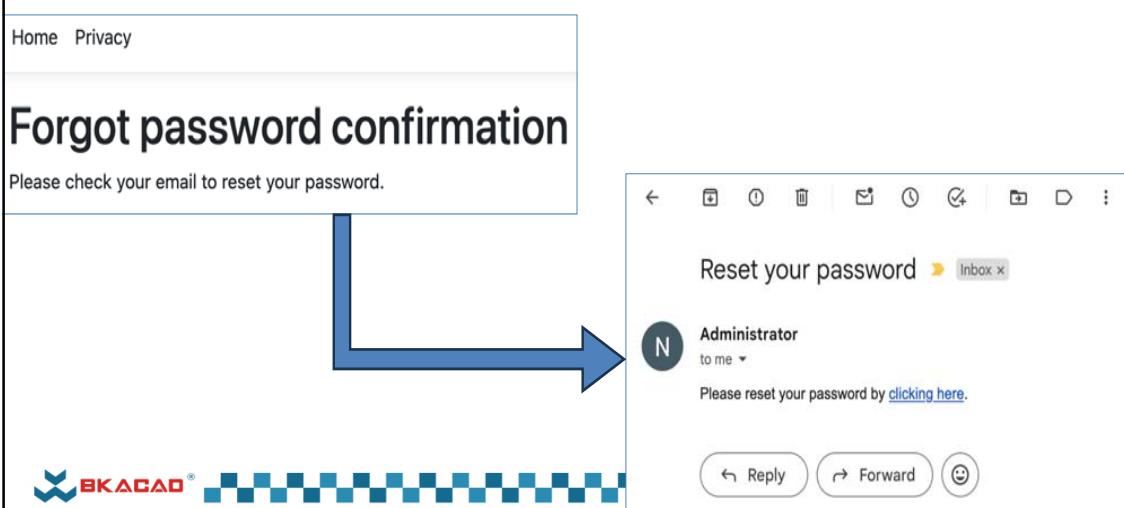
## Đặt lại mật khẩu

- Người dùng có thể yêu cầu hệ thống đặt lại mật khẩu khi quên mật khẩu



## Đặt lại mật khẩu

- Sau khi hệ thống thông báo, người dùng kiểm tra email sẽ nhận được email chứa liên kết đặt lại mật khẩu



## Đặt lại mật khẩu

- Người dùng nhập thông tin để đặt lại mật khẩu

click here to log in.' with links for Home and Privacy at the top."/>

**Reset password**  
Reset your password.

Email: pqh.cntt@gmail.com

Password: .....  
Confirm password: .....

Reset

Home Privacy

**Reset password confirmation**

Your password has been reset. Please [click here to log in.](#)



## 3. PHÂN QUYỀN NGƯỜI DÙNG

- Tạo class Employee, MemberUnit và tạo dữ liệu ngẫu nhiên
- Phân quyền đơn giản trên .Net
- Phân quyền dựa trên vai trò (Role)
- Phân quyền dựa trên yêu cầu (Claim)
- Phân quyền dựa trên chính sách (Policy)
- Tuỳ chỉnh phân quyền hệ thống



## Tạo class Employee và tạo dữ liệu ngẫu nhiên

- Tạo class Entities/Employee
- Sử dụng Code First để ánh xạ vào cơ sở dữ liệu

```
public class Employee
{
    [Key]
    0 references
    public int EmployeeId { get; set; }
    [Required]
    0 references
    public string FirstName { get; set; }
    [Required]
    0 references
    public string LastName { get; set; }
    0 references
    public string Address { get; set; }
    [Required]
    [DataType(DataType.Date)]
    0 references
    public DateTime DateOfBirth { get; set; }
    [Required]
    0 references
    public string Position { get; set; }
    [Required]
    [EmailAddress]
    0 references
    public string Email { get; set; }
    [DataType(DataType.Date)]
    0 references
    public DateTime HireDate { get; set; }
}
```



## Tạo class MemberUnit

- Tạo class Entities/MemberUnit
- Sử dụng Code First để ánh xạ vào CSDL
- Sinh mã CRUD với MemberUnit

```
C MemberUnit.cs 4, U X
Models > Entities > C MemberUnit.cs > ...
1  using System.ComponentModel.DataAnnotations;
2
3  namespace ViciemMVCIdentity.Models.Entities
4  {
    18 references
    5      public class MemberUnit
    6      {
    7          [Key]
    8          11 references
    9          public int MemberUnitId { get; set; }
    10         [Required]
    11         12 references
    12         public string Name { get; set; }
    13         [Required]
    12 references
    13         public string Address { get; set; }
    12 references
    14         public string PhoneNumber { get; set; }
    12 references
    15         public string WebsiteUrl { get; set; }
    16     }
}
```



# Tạo class Employee và tạo dữ liệu ngẫu nhiên

- Xây dựng chức năng **CRUD** với class **Employee**
- Thêm package Bogus vào dự án: `dotnet add package Bogus`
- Tạo class **Models/Process/EmployeeSeeder.cs**



Tạo class	nhiên
<ul style="list-style-type: none"><li>- Xây dựng chức năng <b>CRUD</b> với class <b>Employee</b></li><li>- Thêm package Bogus vào dự án: <code>dotnet add package Bogus</code></li><li>- Tạo class <b>Models/Process/EmployeeSeeder.cs</b></li></ul> <pre>EmployeeSeeder.cs U X Models &gt; Process &gt; EmployeeSeeder.cs &gt; ... 5  namespace VicemMVCIdentity.Models.Process 6  { 7      3 references 8      public class EmployeeSeeder 9      { 10         3 references 11         private readonly ApplicationDbContext _context; 12         0 references 13         public EmployeeSeeder(ApplicationDbContext context) 14         { 15             _context = context; 16         } 17         1 reference 18         public void SeedEmployees(int n) 19         { 20             var employees = GenerateEmployees(n); 21 22             _context.Employee.AddRange(employees); 23             _context.SaveChanges(); 24         } 25         1 reference 26         private List&lt;Employee&gt; GenerateEmployees(int n) 27         { 28             var faker = new Faker&lt;Employee&gt;() 29                 .RuleFor(e =&gt; e.FirstName, f =&gt; f.Name.FirstName()) 30                 .RuleFor(e =&gt; e.LastName, f =&gt; f.Name.LastName()) 31                 .RuleFor(e =&gt; e.Address, f =&gt; f.Address.FullAddress()) 32                 .RuleFor(e =&gt; e.DateOfBirth, f =&gt; f.Date.Past(30, DateTime.Now.AddYears(-20))) 33                 .RuleFor(e =&gt; e.Position, f =&gt; f.Name.JobTitle()) 34                 .RuleFor(e =&gt; e.Email, (f, e) =&gt; f.Internet.Email(e.FirstName, e.LastName)) 35                 .RuleFor(e =&gt; e.HireDate, f =&gt; f.Date.Past(10)); 36             return faker.Generate(n); 37         } 38     } 39 }</pre>	

## Tạo class Employee và tạo dữ liệu ngẫu nhiên

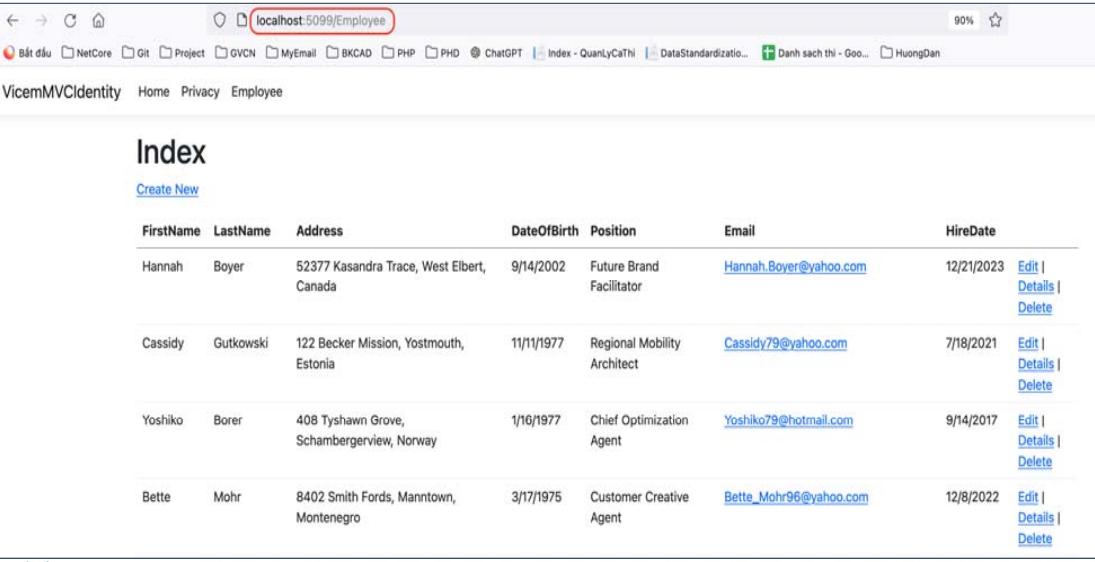
- Xây dựng chức năng **CRUD** với class **Employee**
- Thêm package Bogus vào dự án: **dotnet add package Bogus**
- Tạo class **Models/Process/EmployeeSeeder.cs**
- Cấu hình trong **Program.cs**



```
Program.cs M X
C Program.cs
24 | builder.Services.AddTransient<EmployeeSeeder>();
25 |
26 | var app = builder.Build();
27 |
28 | using (var scope = app.Services.CreateScope())
29 | {
30 |     var services = scope.ServiceProvider;
31 |     var seeder = services.GetRequiredService<EmployeeSeeder>();
32 |     seeder.SeedEmployees(1000);
33 | }
```

## Tạo class Employee và tạo dữ liệu ngẫu nhiên

- Chạy chương trình và truy cập vào đường dẫn **/Employee/Index**



FirstName	LastName	Address	DateOfBirth	Position	Email	HireDate	
Hannah	Boyer	52377 Kasandra Trace, West Elbert, Canada	9/14/2002	Future Brand Facilitator	<a href="#">Hannah.Boyer@yahoo.com</a>	12/21/2023	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>
Cassidy	Gutkowski	122 Becker Mission, Yostmouth, Estonia	11/11/1977	Regional Mobility Architect	<a href="#">Cassidy79@yahoo.com</a>	7/18/2021	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>
Yoshiko	Borer	408 Tyshawn Grove, Schambergerview, Norway	1/16/1977	Chief Optimization Agent	<a href="#">Yoshiko79@hotmail.com</a>	9/14/2017	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>
Bette	Mohr	8402 Smith Fords, Manntown, Montenegro	3/17/1975	Customer Creative Agent	<a href="#">Bette_Mohr96@yahoo.com</a>	12/8/2022	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>

## PHÂN QUYỀN ĐƠN GIẢN TRÊN .NET

- Sửa mã nguồn của EmployeeController.cs như sau:

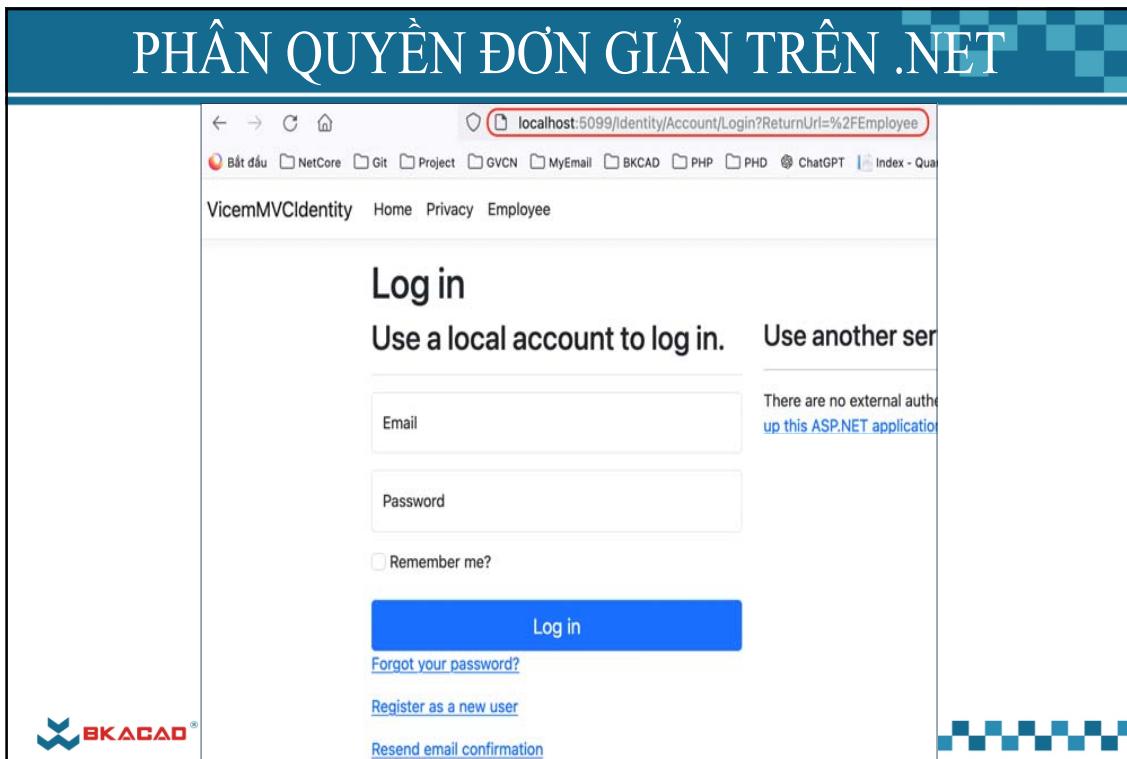
```
C# EmployeeController.cs U X  
Controllers > C# EmployeeController.cs > EmployeeController > .ctor  
1  using Microsoft.AspNetCore.Authorization;  
2  using Microsoft.AspNetCore.Mvc;  
3  using Microsoft.EntityFrameworkCore;  
4  using VicemMVCIdentity.Data;  
5  using VicemMVCIdentity.Models.Entities;  
6  
7  namespace VicemMVCIdentity.Controllers  
8  {  
9  |  [Authorize]  
10 |  public class EmployeeController : Controller  
11 |  {  
12 |  |  private readonly ApplicationDbContext _context;
```



## PHÂN QUYỀN ĐƠN GIẢN TRÊN .NET

- Chạy chương trình và truy cập vào đường dẫn /Employee/Index hoặc /Employee/Create hoặc /Employee/Edit/1 hoặc /Employee/Delete/1  
=> hệ thống sẽ yêu cầu người dùng đăng nhập (nếu không đăng nhập sẽ không có quyền truy cập vào tài nguyên)





The screenshot shows a code editor with the file `EmployeeController.cs` open. The code defines a `EmployeeController` class that inherits from `Controller`. It includes an `Index` action method. Two annotations are highlighted with red boxes: `[Authorize]` at line 9 and `[AllowAnonymous]` at line 19. The code editor interface shows line numbers and some basic navigation controls.

```
Controllers > EmployeeController.cs > EmployeeController.cs > EmployeeController.cs
9 [Authorize]
10 public class EmployeeController : Controller
11 {
12     private readonly ApplicationDbContext _context;
13 
14     public EmployeeController(ApplicationDbContext context)
15     {
16         _context = context;
17     }
18 
19 [AllowAnonymous]
20 public async Task<IActionResult> Index()
21 {
22     return View(await _context.Employee.ToListAsync());
23 }
```

## PHÂN QUYỀN ĐƠN GIẢN TRÊN .NET

- Thuộc tính [Authorize] và [AllowAnonymous] đều có thể sử dụng cho các Action riêng lẻ:

The screenshot shows the code for EmployeeController.cs. It highlights three sections with red boxes:

- Line 19: `[AllowAnonymous]` (3 references)
- Line 24: `[Authorize]` (0 references)
- Line 41: `[Authorize]` (0 references)

The code is as follows:

```
C# EmployeeController.cs U X
Controllers > C# EmployeeController.cs > EmployeeController > Create
10     public class EmployeeController : Controller
11
12
13
14
15
16
17
18
19     [AllowAnonymous]
19     3 references
20     >     public async Task<IActionResult> Index() ...
24     [Authorize]
24     0 references
25     >     public async Task<IActionResult> Details(int? id) ...
41     [Authorize]
41     0 references
42     >     public IActionResult Create() ...
46
```



## PHÂN QUYỀN DỰA TRÊN VAI TRÒ

- Xây dựng chức năng hiển thị danh sách tài khoản.
- Xây dựng chức năng để quản lý vai trò người dùng (Role)
- Xây dựng chức năng thêm vai trò (Role) cho người dùng
- Phân quyền người dùng dựa trên vai trò



## Xây dựng chức năng hiển thị danh sách tài khoản

- Tạo AccountController:

```
File: AccountController.cs (1 file)
```

Controllers > AccountController.cs > ...

```
1  using Microsoft.AspNetCore.Identity;
2  using Microsoft.AspNetCore.Mvc;
3  using Microsoft.EntityFrameworkCore;
4  using VicemMVCIdentity.Models;
5
6  namespace VicemMVCIdentity.Controllers
7  {
8      public class AccountController : Controller
9      {
10         private readonly UserManager< ApplicationUser> _userManager;
11         public AccountController(UserManager< ApplicationUser> userManager)
12         {
13             _userManager = userManager;
14         }
15         public async Task< ActionResult > Index()
16         {
17             var users = await _userManager.Users.ToListAsync();
18             return View(users);
19         }
20     }
21 }
```



## Xây dựng chức năng hiển thị danh sách tài khoản

- Tạo view Account/Index:

```
File: Index.cshtml (1 file)
```

Views > Account > Index.cshtml

```
1  @model IEnumerable< VicemMVCIdentity.Models.ApplicationUser >
2  <table class="table">
3      <thead>
4          <tr>
5              <th>Full Name</th>
6              <th>Email</th>
7          </tr>
8      </thead>
9      <tbody>
10         @foreach (var item in Model)
11         {
12             <tr>
13                 <td>@item.FullName</td>
14                 <td>@item.Email</td>
15             </tr>
16         }
17     </tbody>
18 </table>
```



## Xây dựng chức năng hiển thị danh sách tài khoản

- Chạy chương trình và truy cập vào URL /Account/Index:

Full Name	Email
Pham Quang Hien	phamquanghien@humg.edu.vn
Pham Thanh Cong	pqh.cntt@gmail.com

## Xây dựng chức năng quản lý ROLE

- Tạo RoleController:

```
public class RoleController : Controller
{
    2 references
    private readonly RoleManager<IdentityRole> _roleManager;
    0 references
    public RoleController(RoleManager<IdentityRole> roleManager)
    {
        _roleManager = roleManager;
    }
    0 references
    public async Task<IActionResult> Index()
    {
        var roles = await _roleManager.Roles.ToListAsync();
        return View(roles);
    }
}
```

## Xây dựng chức năng quản lý ROLE

- Tạo view /Role/Index:



```
Views > Role > Index.cshtml U X
1 @model IEnumerable<Microsoft.AspNetCore.Identity.IdentityRole>
2 <div class="d-flex justify-content-between align-items-center">
3     <h3>Role/Index</h3>
4     <a asp-action="Create" class="btn btn-primary">Create New</a>
5 </div>
6 <hr>
7 <table class="table">
8     <thead>
9         <tr>
10            <th>RoleID</th>
11            <th>Role Name</th>
12            <th>Actions</th>
13        </tr>
14    </thead>
15    <tbody>
16        @foreach (var item in Model)
17        {
18            <tr>
19                <td>@item.Id</td>
20                <td>@item.Name</td>
21                <td>
22                    <a asp-action="Edit" asp-route-id="@item.Id">Edit</a> | 
23                    <a asp-action="Delete" asp-route-id="@item.Id">Delete</a>
24                </td>
25            </tr>
26        }
27    </tbody>
28 </table>
```

## Xây dựng chức năng quản lý ROLE

- Cấu hình trong Program.cs:

```
18 builder.Services.AddDatabaseDeveloperPageExceptionFilter();
19 builder.Services.AddRazorPages();
20 builder.Services.AddIdentity<ApplicationUser, IdentityRole>(options =>
21     .AddEntityFrameworkStores<ApplicationContext>());
22 builder.Services.AddControllersWithViews();
23 builder.Services.AddTransient<EmployeeSeeder>();
```

## Xây dựng chức năng quản lý ROLE

- Tạo action `Create` trong `RoleController`:

```
public IActionResult Create()
{
    return View();
}

[HttpPost]
0 references
public async Task<IActionResult> Create(string roleName)
{
    if (!string.IsNullOrEmpty(roleName))
    {
        var role = new IdentityRole(roleName.Trim());
        await _roleManager.CreateAsync(role);
    }
    return RedirectToAction("Index");
}
```



## Xây dựng chức năng quản lý ROLE

- Tạo view `/Role/Create`:

```
Views > Role > Create.cshtml
1  <h2>Create Role</h2>
2
3  <form asp-action="Create">
4      <div class="form-group">
5          <label for="roleName">Role Name:</label>
6          <input type="text" class="form-control" id="roleName" name="roleName">
7      </div>
8      <button type="submit" class="btn btn-primary">Create</button>
9  </form>
```



## Xây dựng chức năng quản lý ROLE

- Tạo action Edit trong RoleController:

```
public async Task<IActionResult> Edit(string id)
{
    var role = await _roleManager.FindByIdAsync(id);
    if (role == null)
    {
        return NotFound();
    }
    return View(role);
}
[HttpPost]
0 references
public async Task<IActionResult> Edit(string id, string newName)
{
    var role = await _roleManager.FindByIdAsync(id);
    if (role == null)
    {
        return NotFound();
    }
    role.Name = newName;
    await _roleManager.UpdateAsync(role);
    return RedirectToAction("Index");
}
```



## Xây dựng chức năng quản lý ROLE

- Tạo view /Role/Edit:

```
Views > Role > Edit.cshtml
Views > Role > Edit.cshtml
1 @model Microsoft.AspNetCore.Identity.IdentityRole
2
3 <h2>Edit Role</h2>
4
5 <form asp-action="Edit" asp-route-id="@Model.Id">
6     <div class="form-group">
7         <label for="roleName">Role Name:</label>
8         <input type="text" class="form-control" id="roleName" name="newName" value="@Model.Name">
9     </div>
10    <button type="submit" class="btn btn-primary">Save</button>
11 </form>
```



## Xây dựng chức năng quản lý ROLE

- Tạo action Delete trong RoleController:

```
[HttpPost]  
0 references  
public async Task<IActionResult> Delete(string id)  
{  
    var role = await _roleManager.FindByIdAsync(id);  
    if (role != null)  
    {  
        await _roleManager.DeleteAsync(role);  
    }  
    return RedirectToAction("Index");  
}
```



## Xây dựng chức năng quản lý ROLE

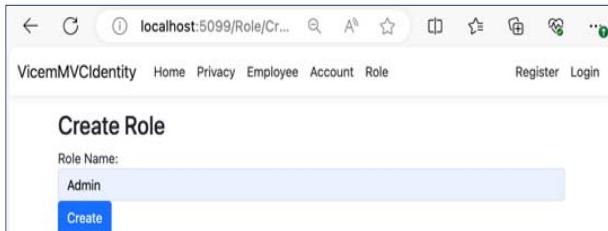
- Tạo view /Role/Delete:

```
☰ Delete.cshtml U X  
Views > Role > Delete.cshtml  
1 @model Microsoft.AspNetCore.Identity.IdentityRole  
2  
3 <h2>Delete Role</h2>  
4  
5 <p>Are you sure you want to delete role "@Model.Name"?</p>  
6  
7 <form asp-action="Delete" asp-route-id="@Model.Id" method="post">  
8     <button type="submit" class="btn btn-danger">Delete</button>  
9     <a asp-action="Index">Cancel</a>  
10    </form>
```



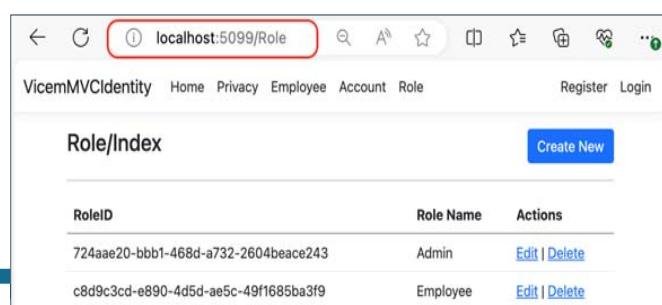
## Xây dựng chức năng quản lý ROLE

- Truy cập vào URL [/Role/Create](#) và nhập dữ liệu để tạo mới Role:



The screenshot shows a browser window with the URL `localhost:5099/Role/Cr...`. The page title is "Create Role". There is an input field labeled "Role Name:" containing the value "Admin". Below the input field is a blue "Create" button.

- Truy cập vào URL [/Role/Index](#):



The screenshot shows a browser window with the URL `localhost:5099/Role`. The page title is "Role/Index". At the top right is a blue "Create New" button. Below it is a table with three columns: "RoleID", "Role Name", and "Actions". The table contains two rows. Row 1: RoleID "724aae20-bbb1-468d-a732-2604beace243", Role Name "Admin", Actions "Edit | Delete". Row 2: RoleID "c8d9c3cd-e890-4d5d-ae5c-49f1685ba3f9", Role Name "Employee", Actions "Edit | Delete".

## Xây dựng chức năng thêm vai trò cho người dùng

- Tạo class [Models/ViewModels/UserWithRoleVM](#)
- Tạo class [Models/ViewModels/AssignRoleVM](#)
- Chính sửa code [AccountController](#)
- Chính sửa code view [/Account/Index](#)
- Tạo view [/Account/AssignRole](#)
- Chạy thử nghiệm chức năng Assign Role



## Tạo class Models/ViewModels/UserWithRoleVM

```
C# UserWithRoleVM.cs 2, U X  
Models > ViewModels > C# UserWithRoleVM.cs > ...  
1  namespace VicemMVCIdentity.Models.ViewModels  
2  {  
    5 references  
3  |  public class UserWithRoleVM  
4  |  {  
    3 references  
5  |  |  public ApplicationUser User { get; set; }  
    2 references  
6  |  |  public IList<string> Roles { get; set; }  
7  |  }  
8  }
```



## Tạo class Models/ViewModels/AssignRoleVM

```
C# AssignRoleVM.cs 4, U X  
Models > ViewModels > C# AssignRoleVM.cs > ...  
1  namespace VicemMVCIdentity.Models.ViewModels  
2  {  
    5 references  
3  |  public class AssignRoleVM  
4  |  {  
    3 references  
5  |  |  public string UserId { get; set; }  
    5 references  
6  |  |  public IList<string> SelectedRoles { get; set; }  
    2 references  
7  |  |  public IList<RoleVM>? AllRoles { get; set; }  
8  |  }  
    2 references  
9  |  public class RoleVM  
10 |  {  
    3 references  
11 |  |  public string Id { get; set; }  
    4 references  
12 |  |  public string Name { get; set; }  
13 |  }  
14 }
```



## Chỉnh sửa code AccountController

```
private readonly UserManager< ApplicationUser > _userManager;
2 references
private readonly RoleManager< IdentityRole > _roleManager;
0 references
public AccountController(UserManager< ApplicationUser > userManager, RoleManager< IdentityRole > roleManager)
{
    _userManager = userManager;
    _roleManager = roleManager;
}
0 references
public async Task< ActionResult > Index()
{
    var users = await _userManager.Users.ToListAsync();
    var usersWithRoles = new List< UserWithRoleVM >();

    foreach (var user in users)
    {
        var roles = await _userManager.GetRolesAsync(user);
        usersWithRoles.Add(new UserWithRoleVM { User = user, Roles = roles.ToList() });
    }

    return View(usersWithRoles);
}
```



## Chỉnh sửa code AccountController

```
public async Task< IActionResult > AssignRole(string userId)
{
    var user = await _userManager.FindByIdAsync(userId);
    if (user == null)
    {
        return NotFound();
    }
    var userRoles = await _userManager.GetRolesAsync(user);
    var allRoles = await _roleManager.Roles.Select(r => new RoleVM { Id = r.Id, Name = r.Name }).ToListAsync();
    var viewModel = new AssignRoleVM
    {
        UserId = userId,
        AllRoles = allRoles,
        SelectedRoles = userRoles
    };
    return View(viewModel);
}
```



## Chỉnh sửa code AccountController

```
[HttpPost]
0 references
public async Task<IActionResult> AssignRole(AssignRoleVM model)
{
    if (ModelState.IsValid)
    {
        var user = await _userManager.FindByIdAsync(model.UserId);
        if (user == null)
        {
            return NotFound();
        }
        var userRoles = await _userManager.GetRolesAsync(user);
        foreach (var role in model.SelectedRoles)
        {
            if (!userRoles.Contains(role))
            {
                await _userManager.AddToRoleAsync(user, role);
            }
        }
        foreach (var role in userRoles)
        {
            if (!model.SelectedRoles.Contains(role))
            {
                await _userManager.RemoveFromRoleAsync(user, role);
            }
        }
    }
    return RedirectToAction("Index", "Account");
}
return View(model);
```



## Chỉnh sửa code view /Account/Index

```
@using Microsoft.AspNetCore.Identity
@using VicemMVCIdentity.Models.ViewModels
@model IEnumerable<UserWithRoleVM>


| Username | Roles | Action |
|----------|-------|--------|
|----------|-------|--------|


```



## Tạo view /Account/AssignRole

```
AssignRole.cshtml 1, U X
Views > Account > AssignRole.cshtml
1 @using VicemMVCIdentity.Models.ViewModels
2 @model AssignRoleVM
3
4 <h2>Assign Role</h2>
5 <hr>
6 <form asp-action="AssignRole" method="post">
7     @foreach (var role in Model.AllRoles)
8     {
9         <div>
10            <input class="checkbox" type="checkbox" id="@role.Id" name="SelectedRoles" value="@role.Name"
11            @Model.SelectedRoles != null && Model.SelectedRoles.Contains(role.Name) ? "checked" : "">
12            <label for="@role.Id">@role.Name</label>
13        </div>
14    }
15    <input type="hidden" name="UserId" value="@Model.UserId">
16    <button type="submit" class="btn btn-primary">Assign Roles</button>
17 </form>
```



## Chạy thử nghiệm chức năng Assign Role

- Chạy ứng dụng và truy cập vào URL [/Account/Index](#), sau đó thực thi chức năng AssignRole và xem kết quả

Username	Roles	Action
phamquanghien@humg.edu.vn	Admin, Employee,	<a href="#">Assign Role</a>
pqh.cntt@gmail.com	Employee,	<a href="#">Assign Role</a>



## Phân quyền người dùng dựa trên vai trò

- Sửa lại mã nguồn của EmployeeController
- Cấu hình file Program.cs
- Chạy thử nghiệm



## Sửa lại mã nguồn của EmployeeController

```
EmployeeController.cs M X
Controllers > EmployeeController.cs > EmployeeController > ⓘ
14 | [Authorize]
15 | 1 reference
16 | public class EmployeeController : Controller
17 | {
18 |     13 references
19 |     private readonly ApplicationDbContext _context;
20 |     0 references
21 |     public EmployeeController(ApplicationDbContext context)
22 |     // GET: Employee
23 |     3 references
24 |     public async Task<IActionResult> Index() ->
25 |         [Authorize(Roles = "Employee")]
26 |         0 references
27 |     public async Task<IActionResult> Details(int? id) ->
28 |         [Authorize(Roles = "Admin")]
29 |         0 references
30 |     public IActionResult Create() ->
31 |         [HttpPost]
32 |         [ValidateAntiForgeryToken]
33 |         0 references
34 |     public async Task<IActionResult> Create([Bind("Employee")]
35 |         [Authorize(Roles = "Admin")]
36 |         0 references
37 |     public async Task<IActionResult> Edit(int? id) ->
38 |         [HttpPost]
39 |         [ValidateAntiForgeryToken]
40 |         0 references
```

BKACAD®

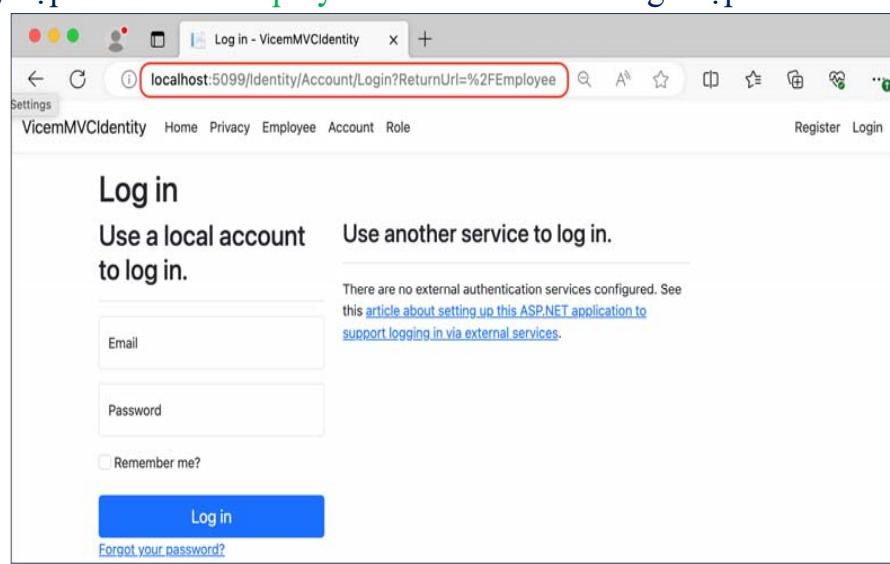
## Cấu hình file Program.cs

```
C# Program.cs M X  
C# Program.cs  
23     builder.Services.AddTransient<EmployeeSeeder>();  
24     builder.Services.ConfigureApplicationCookie(options =>  
25     {  
26         options.LoginPath = $"/Identity/Account/Login";  
27         options.LogoutPath = $"/Identity/Account/Logout";  
28         options.AccessDeniedPath = $"/Identity/Account/AccessDenied";  
29     });  
30     var app = builder.Build();
```



## Chạy thử nghiệm

- Truy cập vào URL `/Employee/Index` khi chưa đăng nhập



## Chạy thử nghiệm

- Truy cập vào URL `/Employee/Index` khi đã đăng nhập

The screenshot shows a web browser window titled "Index - VicemMVCIdentity". The address bar contains "localhost:5099/Employee". The top navigation bar includes "Settings", "VicemMVCIdentity", "Home", "Privacy", "Employee", "Account", "Role", "Hello pqh.cntt@gmail.com!", and "Logout". The main content area is titled "Index" and contains a table with two rows of employee data:

FirstName	LastName	Address	DateOfBirth	Position	Email
Alexa	Blanda	73502 Terry Avenue, Gloverbury, Paraguay	3/21/2000	National Communications Associate	<a href="mailto:Alexa.Blanda@gmail.com">Alexa.Blanda@gmail.com</a>
Hank	Spencer	8388 Daniel Cliff, Port Kendalland, Timor-Leste	12/26/2004	Central Division Producer	<a href="mailto:Hank.Spencer26@yahoo.com">Hank.Spencer26@yahoo.com</a>

## Chạy thử nghiệm

- Truy cập vào URL `/Employee/Details/1`

=> Do tài khoản có role là `Employee` nên được phép truy cập

The screenshot shows a web browser window titled "Details - VicemMVCIdentity". The address bar contains "localhost:5099/Employee/Details/1". The top navigation bar includes "Settings", "VicemMVCIdentity", "Home", "Privacy", "Employee", "Account", "Role", "Hello pqh.cntt@gmail.com!", and "Logout". The main content area is titled "Details" and contains the following information for an employee:

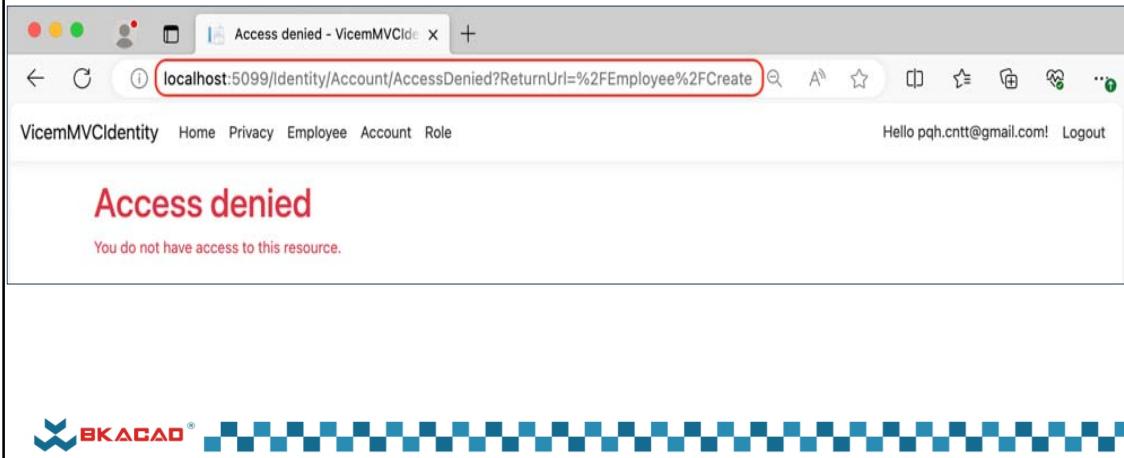
Employee

FirstName	Alexa
LastName	Blanda
Address	73502 Terry Avenue, Gloverbury, Paraguay
DateOfBirth	3/21/2000
Position	National Communications Associate
Email	<a href="mailto:Alexa.Blanda@gmail.com">Alexa.Blanda@gmail.com</a>
HireDate	1/7/2015

## Chạy thử nghiệm

- Truy cập vào URL [/Employee/Create](#)

=> Do tài khoản có role là Employee nên bị từ chối truy cập



## PHÂN QUYỀN DỰA TRÊN YÊU CẦU (CLAIM)

- Xây dựng chức năng thêm, xoá Claim cho User
- Phân quyền người dùng với Claim



## Xây dựng chức năng thêm, xoá Claim cho User

- Tạo class Views/ViewModels/UserClaimVM:

```
public class UserClaimVM
{
    3 references
    public string UserId { get; set; }
    2 references
    public string UserName { get; set; }
    2 references
    public List<Claim> UserClaims { get; set; }
    1 reference
    public UserClaimVM(string userId, string userName, List<Claim> userClaims)
    {
        UserId = userId;
        UserName = userName;
        UserClaims = userClaims;
    }
}
```



## Xây dựng chức năng thêm, xoá Claim cho User

- Tạo action AddClaim trong AccountController:

```
public async Task<IActionResult> AddClaim(string userId)
{
    var user = await _userManager.FindByIdAsync(userId);
    var userClaims = await _userManager.GetClaimsAsync(user);
    var model = new UserClaimVM(userId, user.UserName, userClaims.ToList());
    return View(model);
}
[HttpPost]
0 references
public async Task<IActionResult> AddClaim(string userId, string claimType, string claimValue)
{
    var user = await _userManager.FindByIdAsync(userId);
    var result = await _userManager.AddClaimAsync(user, new Claim(claimType, claimValue));
    if (result.Succeeded)
    {
        return RedirectToAction("AddClaim", new { userId });
    }
    return View();
}
```



## Xây dựng chức năng thêm, xoá Claim cho User

- Tạo Views/Account/AddClaim.cshtml:



```
Views > Account > AddClaim.cshtml
1  @model VicemMVCIdentity.Models.ViewModels.UserClaimVM
2
3  <h3>User: @Model.UserName</h3>
4  <form asp-controller="Account" asp-action="AddClaim" method="post">
5      <div class="row">
6          <input type="hidden" name="userId" value="@Model.UserId" />
7          <input type="text" id="claimType" name="claimType" class="form-control col" placeholder="Enter Claim Type" />
8          <input type="text" id="claimValue" name="claimValue" class="form-control col" placeholder="Enter Claim Value" />
9          <button type="submit" class="col btn btn-success">Add new Claim</button>
10     </div>
11 </form>
12 <br>
13 <h3>Current Claims:</h3>
14 <table class="table">
15     <thead>
16         <tr>
17             <th>Claim Type</th>
18             <th>Claim Value</th>
19             <th>Action</th>
20         </tr>
21     </thead>
22     <tbody>--</tbody>
23     </table>
24 </div>
25 </div>
26 </div>
27 </div>
28 </div>
29 </div>
30 </div>
31 </div>
32 </div>
33 </div>
34 </div>
35 </div>
36 </div>
37 </div>
38 </div>
39 </div>
```

## Xây dựng chức năng thêm, xoá Claim cho User

- Tạo Views/Account/AddClaim.cshtml:



```
Views > Account > AddClaim.cshtml
14 <table class="table">
15     <thead>
16     </thead>
17     <tbody>
18         @foreach (var claim in Model.UserClaims)
19         {
20             <tr>
21                 <td>@claim.Type</td>
22                 <td>@claim.Value</td>
23                 <td>
24                     <form asp-controller="Account" asp-action="DeleteClaim" method="post">
25                         <input type="hidden" name="userId" value="@Model.UserId" />
26                         <input type="hidden" name="claimType" value="@claim.Type" />
27                         <input type="hidden" name="claimValue" value="@claim.Value" />
28                         <button type="submit" class="btn btn-danger">Delete Claim</button>
29                     </form>
30                 </td>
31             </tr>
32         }
33     </tbody>
34 </table>
```

## Xây dựng chức năng thêm, xoá Claim cho User

- Cấu hình file Program.cs:

```
22     builder.Services.AddControllersWithViews();
23     builder.Services.AddAuthorization(options =>
24     {
25         options.AddPolicy("Role", policy => policy.RequireClaim("Role", "AdminOnly"));
26         options.AddPolicy("Permission", policy => policy.RequireClaim("Role", "EmployeeOnly"));
27     });
28     builder.Services.AddTransient<EmployeeSeeder>();
```

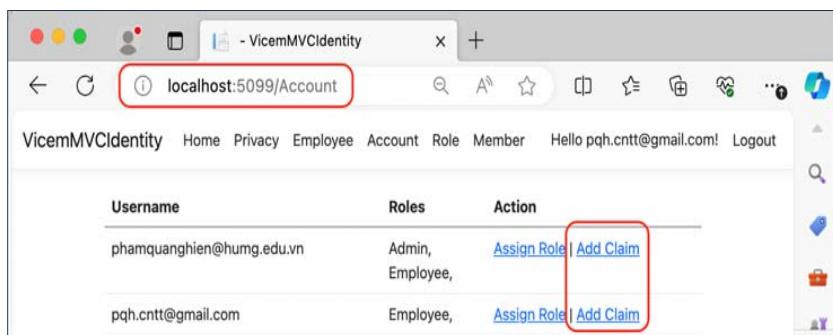


## Xây dựng chức năng thêm, xoá Claim cho User

- Chính sửa code file Views/Account/Index.cshtml:

```
<td>
    <a asp-controller="Account" asp-action="AssignRole" asp-route-userId="@userWithRoles.User.Id">Assign Role</a> | 
    <a asp-controller="Account" asp-action="AddClaim" asp-route-userId="@userWithRoles.User.Id">Add Claim</a>
</td>
```

- Chạy chương trình và truy cập vào URL /Account/Index:



## Xây dựng chức năng thêm, xoá Claim cho User

- Thêm Claim cho tài khoản:

User: phamquanghien@humg.edu.vn

Enter Claim Type Enter Claim Value Add new Claim

### Current Claims:

Claim Type	Claim Value	Action
Role	EmployeeOnly	Delete Claim

User: pqh.cntt@gmail.com

Enter Claim Type Enter Claim Value Add new Claim

### Current Claims:

Claim Type	Claim Value	Action
Role	AdminOnly	Delete Claim
Role	EmployeeOnly	Delete Claim



## Phân quyền người dùng với Claim

- Chính sửa code: MemberUnitController:

```
[Authorize(Policy = "Permission")]
1 reference
public class MemberUnitController : Controller
{
    15 references
    private readonly ApplicationDbContext _context;

    0 references
    public MemberUnitController(ApplicationDbContext context) ...

    // GET: MemberUnit
    3 references
    public async Task<IActionResult> Index() ...

    // GET: MemberUnit/Details/5
    0 references
    public async Task<IActionResult> Details(int? id) ...

    [Authorize(Policy = "Role")]
    0 references
    public IActionResult Create()
```



## Phân quyền người dùng với Claim

- Chạy chương trình và truy cập vào URL: /MemberUnit/Index:

Name	Address	PhoneNumber	WebsiteUrl	
Vicem Hạ Long	Thống Nhất, huyện Hoành Bồ, tỉnh Quảng Ninh	0203 3699240	www.ximanghalong.vn	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>
Vicem Hoàng Thạch	Thị trấn Minh Tân - Huyện Kinh Môn - Tỉnh Hải Dương	0220 3821092	www.ximangoanthach.com	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>



## Phân quyền người dùng với Claim

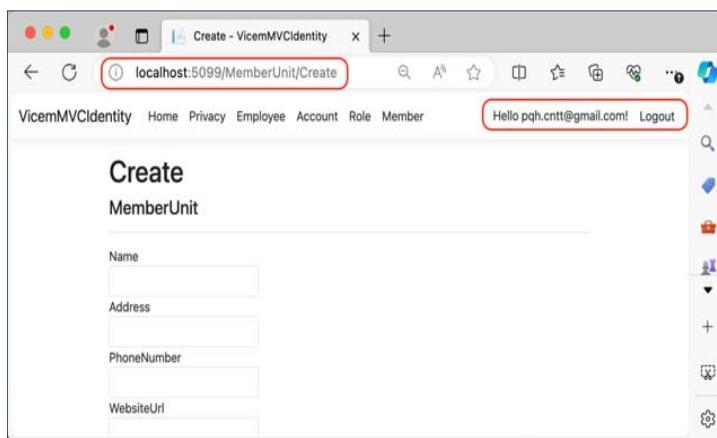
- Chạy chương trình và truy cập vào URL: /MemberUnit/Index:

Name	Address	PhoneNumber	WebsiteUrl	
Vicem Hạ Long	Thống Nhất, huyện Hoành Bồ, tỉnh Quảng Ninh	0203 3699240	www.ximanghalong.vn	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>
Vicem Hoàng Thạch	Thị trấn Minh Tân - Huyện Kinh Môn - Tỉnh Hải Dương	0220 3821092	www.ximangoanthach.com	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>



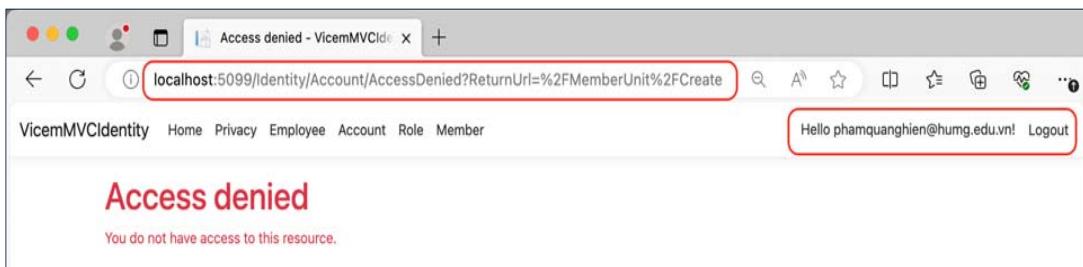
## Phân quyền người dùng với Claim

- Chạy chương trình và truy cập vào URL: /MemberUnit/Create:



## Phân quyền người dùng với Claim

- Chạy chương trình và truy cập vào URL: /MemberUnit/Create:



## PHÂN QUYỀN DỰA TRÊN CHÍNH SÁCH (POLICY)

- Policy Require Role
- Policy Requirements



### Policy Require Role

- Add Role cho tài khoản người dùng

A screenshot of a web browser window titled 'VicemMVCIdentity'. The URL bar shows 'localhost:5099/Account'. The page content is a table titled 'Policy Require Role' with columns 'Username', 'Roles', and 'Action'. There are two rows:

Username	Roles	Action
phamquanghien@humg.edu.vn	Admin, Employee,	<a href="#">Assign Role</a>   <a href="#">Add Claim</a>
pqh.cntt@gmail.com	Employee,	<a href="#">Assign Role</a>   <a href="#">Add Claim</a>

The 'Action' column contains two hyperlinks: 'Assign Role' and 'Add Claim'. The entire screenshot is framed by a decorative blue and white checkered border at the bottom.

## Policy Require Role

- Cấu hình file Program.cs:

```
builder.Services.AddAuthorization(options =>
{
    options.AddPolicy("Role", policy => policy.RequireClaim("Role", "AdminOnly"));
    options.AddPolicy("Permission", policy => policy.RequireClaim("Role", "EmployeeOnly"));
    options.AddPolicy("PolicyAdmin", policy => policy.RequireRole("Admin"));
    options.AddPolicy("PolicyEmployee", policy => policy.RequireRole("Employee"));
});
```



## Policy Require Role

- Cấu hình file MemberUnitController:

```
[Authorize(Policy = "PolicyEmployee")]
1 reference
public class MemberUnitController : Controller
{
    15 references
    private readonly ApplicationDbContext _context;

    0 references
    public MemberUnitController(ApplicationDbContext context) ...

    // GET: MemberUnit
    3 references
    public async Task<IActionResult> Index() ...

    // GET: MemberUnit/Details/5
    0 references
    public async Task<IActionResult> Details(int? id) ...

    [Authorize(Policy = "PolicyAdmin")]
    0 references
    public IActionResult Create() ...
```



## Policy Require Role

- Chạy chương trình và truy cập vào URL: /MemberUnit/Index:

Name	Address	PhoneNumber	WebsiteUrl	
Vicem	Thống Nhất, Huyện Hoành Bồ, tỉnh Quảng Ninh	0203 3699240	www.ximanghalong.vn	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>
Hạ Long				



## Policy Require Role

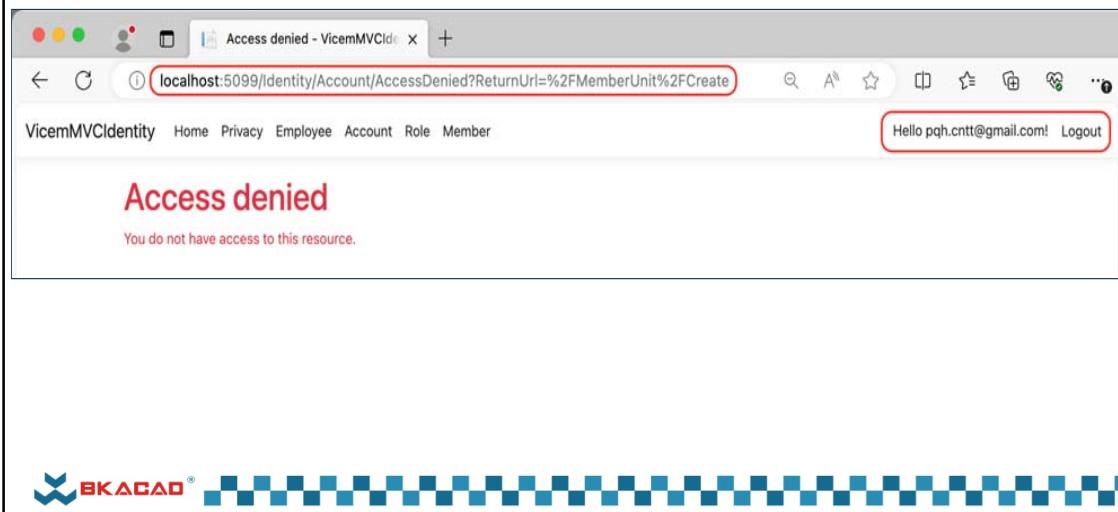
- Chạy chương trình và truy cập vào URL: /MemberUnit/Index:

Name	Address	PhoneNumber	WebsiteUrl	
Vicem Hạ Long	Thống Nhất, huyện Hoành Bồ, tỉnh Quảng Ninh	0203 3699240	www.ximanghalong.vn	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>
Vicem Hoàng Thạch	Thị trấn Minh Tân - Huyện Kinh Môn - Tỉnh Hải Dương	0220 3821092	www.ximanghoangthach.com	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>



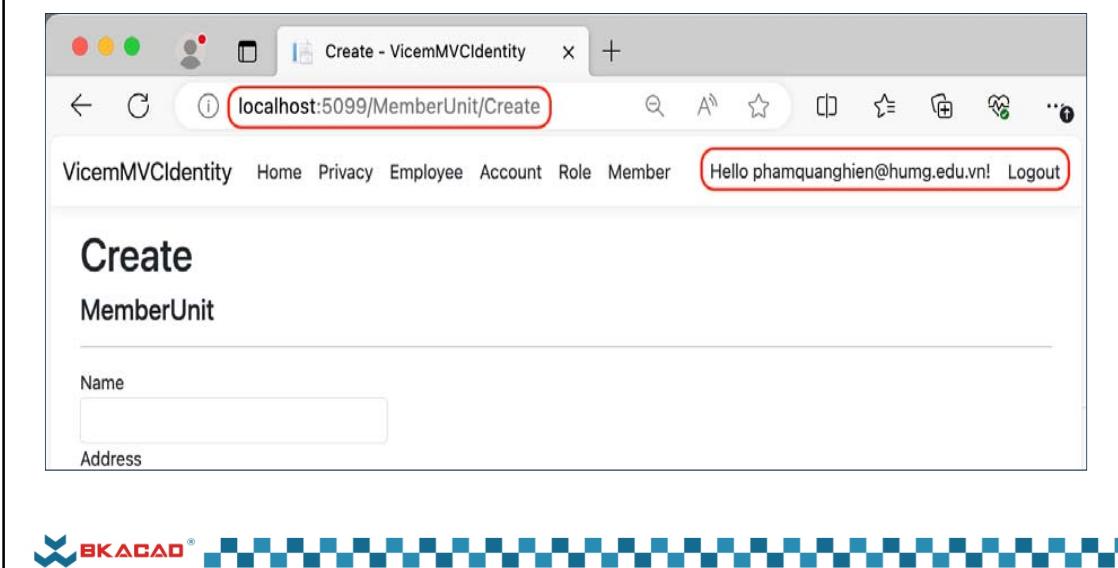
## Policy Require Role

- Chạy chương trình và truy cập vào URL: /MemberUnit/Create:



## Policy Require Role

- Chạy chương trình và truy cập vào URL: /MemberUnit/Create:



## Policy Requirements

- Tạo class Models/Process/PolicyByPhoneNumberHandle.cs:

```
public class PolicyByPhoneNumberRequirement : IAuthorizationRequirement { }
2 references
public class PolicyByPhoneNumberHandler : AuthorizationHandler<PolicyByPhoneNumberRequirement>
{
    2 references
    private readonly IServiceProvider _serviceProvider;
    0 references
    public ..._PolicyByPhoneNumberHandler(IServiceProvider serviceProvider)
    {
        _serviceProvider = serviceProvider;
    }
    0 references
    protected override async Task HandleRequirementAsync(AuthorizationHandlerContext context, PolicyByPhoneNumberRequirement requirement)
    {
        var httpContext = _serviceProvider.GetRequiredService<IHttpContextAccessor>().HttpContext;
        if (httpContext == null)
        {
            return;
        }

        var userManager = httpContext.RequestServices.GetRequiredService<UserManager<ApplicationUser>>();
        var user = await userManager.GetUserAsync(context.User);
        if (user != null && !string.IsNullOrWhiteSpace(user.PhoneNumber))
        {
            context.Succeed(requirement);
        }
    }
}
```



## Policy Requirements

- Cấu hình file Program.cs:

```
builder.Services.AddAuthorization(options =>
{
    options.AddPolicy("Role", policy => policy.RequireClaim("Role", "AdminOnly"));
    options.AddPolicy("Permission", policy => policy.RequireClaim("Role", "EmployeeOnly"));
    options.AddPolicy("PolicyAdmin", policy => policy.RequireRole("Admin"));
    options.AddPolicy("PolicyEmployee", policy => policy.RequireRole("Employee"));
    options.AddPolicy("PolicyByPhoneNumber", policy => policy.Requirements.Add(new PolicyByPhoneNumberRequirement()));
});

builder.Services.AddSingleton<IAuthorizationHandler, PolicyByPhoneNumberHandler>();
builder.Services.AddTransient<EmployeeSeeder>();
```

- Cấu hình AccountController:

```
[Authorize(Policy = "PolicyByPhoneNumber")]
1 reference
public class AccountController : Controller
{
    16 references
    private readonly UserManager<ApplicationUser> _userManager;
    2 references
    private readonly RoleManager<IdentityRole> _roleManager;
```



## Policy Requirements

- Chạy chương trình và truy cập vào URL /Account/Index:

Username	PhoneNumber	Roles	Action
phamquanghien@humg.edu.vn	0987654321	Admin, Employee,	<a href="#">Assign Role   Add Claim</a>
pqh.cntt@gmail.com		Employee,	<a href="#">Assign Role   Add Claim</a>

Access denied

You do not have access to this resource.



## TUỲ CHỈNH PHÂN QUYỀN

- Tạo danh sách các chức năng của hệ thống.
- Xây dựng chức năng AssignClaim
- Cấu hình file Program.cs
- Sử dụng thuộc tính [Authorize] trên controller
- Chạy thử nghiệm



## Tạo danh sách các chức năng của hệ thống

- Tạo class Models/Process/SystemPermissions:

```
C# SystemPermissions.cs U X  
Models > Process > C# SystemPermissions.cs > ...  
1  namespace VicemMVCIentity.Models.Process  
2  {  
3      32 references  
4      public enum SystemPermissions  
5      {  
6          1 reference | 2 references | 2 references | 2 references  
7          EmployeeView, EmployeeCreate, EmployeeEdit, EmployeeDelete,  
8          1 reference | 2 references | 2 references | 2 references  
9          MemberUnitView, MemberUnitCreate, MemberUnitEdit, MemberUnitDelete,  
10         1 reference | 2 references | 0 references | 3 references | 2 references  
11         RoleView, RoleCreate, RoleEdit, RoleDelete, AssignClaim,  
12         1 reference | 2 references | 2 references | 1 reference  
13         AccountView, AssignRole, AddClaim, DeleteClaim  
14     }  
15 }  
16 }
```



## Xây dựng chức năng AssignClaim

- Tạo class Models/ViewModels/RoleClaimVM:

```
namespace VicemMVCIentity.Models.ViewModels  
{  
    5 references  
    public class RoleClaimVM  
    {  
        3 references  
        public string RoleId { get; set; }  
        3 references  
        public string RoleName { get; set; }  
        7 references  
        public List<RoleClaim> Claims { get; set; }  
    }  
    2 references  
    public class RoleClaim  
    {  
        3 references  
        public string Type { get; set; }  
        4 references  
        public string Value { get; set; }  
        3 references  
        public bool Selected { get; set; }  
    }  
}
```



## Xây dựng chức năng AssignClaim

- Tạo action `AssignClaim` trong `RoleController`:

```
public async Task<IActionResult> AssignClaim(string id)
{
    var role = await _roleManager.FindByIdAsync(id);
    if (role == null)
    { return BadRequest(); }
    var allPermissions = Enum.GetValues(typeof(SystemPermissions)).Cast<SystemPermissions>().Select(p => p.ToString()).ToList();
    var roleClaims = await _roleManager.GetClaimsAsync(role);
    if (roleClaims == null)
    {
        roleClaims = new List<Claim>();
    }
    var model = new RoleClaimVM
    {
        RoleId = role.Id,
        RoleName = role.Name,
        Claims = allPermissions.Select(p => new RoleClaim
        {
            Type = "Permission",
            Value = p,
            Selected = roleClaims.Any(c => c.Type == "Permission" && c.Value == p)
        }).ToList()
    };
    return View(model);
}
```



## Xây dựng chức năng AssignClaim

```
[HttpPost]
[ValidateAntiForgeryToken]
0 references
public async Task<IActionResult> AssignClaim(RoleClaimVM model)
{
    if (!ModelState.IsValid)
    { return View(model); }
    var role = await _roleManager.FindByIdAsync(model.RoleId);
    if (role == null)
    { return BadRequest(); }
    var claims = await _roleManager.GetClaimsAsync(role);
    if (claims == null)
    {
        claims = new List<Claim>();
    }
    foreach (var claim in claims.Where(c => c.Type == "Permission"))
    {
        await _roleManager.RemoveClaimAsync(role, claim);
    }
    foreach (var claim in model.Cclaims.Where(c => c.Selected))
    {
        await _roleManager.AddClaimAsync(role, new Claim(claim.Type, claim.Value));
    }
    return RedirectToAction(nameof(Index));
}
```



## Xây dựng chức năng AssignClaim

- Tạo view Role/AssignClaim:

```
@model VicemMVCIdentity.Models.ViewModels.RoleClaimVM
<h3>Assign Claims to @Model.RoleName</h3>
<form asp-action="AssignClaim" method="post">
<input type="hidden" asp-for="RoleId" />
<input type="hidden" asp-for="RoleName" />
<table class="table">
<thead>
<tr>
<th>Permission</th>
<th>Assigned</th>
</tr>
</thead>
<tbody>
@for (int i = 0; i < Model.Claims.Count; i++)
{
<tr>
<td>@Model.Claims[i].Value</td>
<td>
<input type="checkbox" asp-for="Claims[i].Selected" />
<input type="hidden" asp-for="Claims[i].Type" />
<input type="hidden" asp-for="Claims[i].Value" />
</td>
</tr>
}
</tbody>
</table>
<button type="submit" class="btn btn-primary">Save</button>
</form>
```



## Xây dựng chức năng AssignClaim

- Chạy chương trình và truy cập vào URL /Role/Index:

RoleID	Role Name	Actions
724aae20-bbb1-468d-a732-2604beace243	Admin	<a href="#">Edit</a>   <a href="#">Delete</a> <a href="#">Assign Claim</a>
c8d9c3cd-e890-4d5d-ae5c-49f1685ba3f9	Employee	<a href="#">Edit</a>   <a href="#">Delete</a> <a href="#">Assign Claim</a>



## Xây dựng chức năng AssignClaim

- Thực hiện assign cho role Admin:

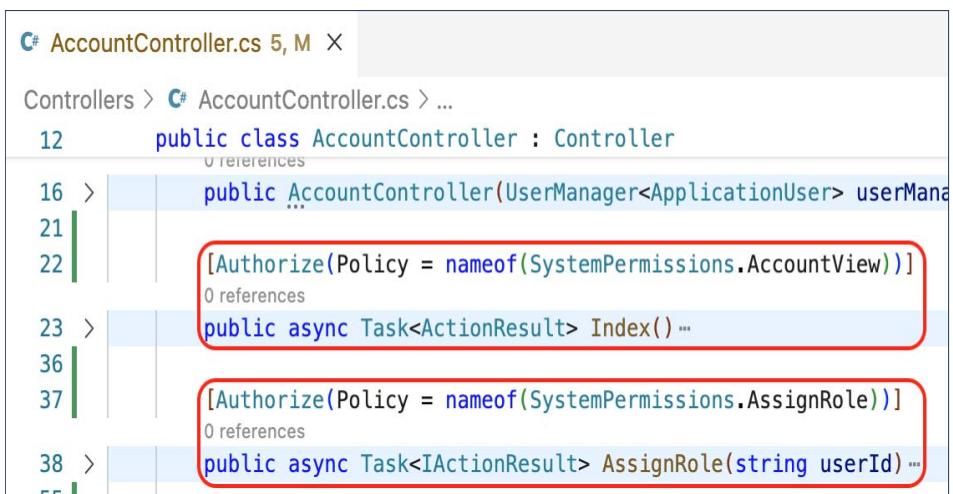
The screenshot shows a web browser window titled "VicemMVCIdentity" with the URL "localhost:5099/Role/AssignClaim?2...". The page has a header with links: Home, Privacy, Employee, Account, Role, Member, Hello pqh.cntt@gmail.com!, and Logout. Below the header is a table titled "Assign Claims to Admin". The table has two columns: "Permission" and "Assigned". The "Assigned" column contains checkboxes. A red box highlights the "Assigned" column for the "AssignClaim" permission, which is checked. Other permissions listed include ViewEmployee, CreateEmployee, EditEmployee, DeleteEmployee, ViewMemberUnit, CreateMemberUnit, EditMemberUnit, DeleteMemberUnit, ViewRole, CreateRole, EditRole, DeleteRole, ViewAccount, and AssignRole.

## Cấu hình trong file Program.cs

```
builder.Services.AddAuthorization(options =>
{
    foreach (var permission in Enum.GetValues(typeof(SystemPermissions)).Cast<SystemPermissions>())
    {
        options.AddPolicy(permission.ToString(), policy =>
            policy.RequireClaim("Permission", permission.ToString()));
    }
    // options.AddPolicy("ViewEmployee", policy => policy.RequireClaim("Employee", "Index"));
    // options.AddPolicy("CreateEmployee", policy => policy.RequireClaim("Employee", "Create"));
    // options.AddPolicy("Role", policy => policy.RequireClaim("Role", "AdminOnly"));
    // options.AddPolicy("Permission", policy => policy.RequireClaim("Role", "EmployeeOnly"));
    // options.AddPolicy("PolicyAdmin", policy => policy.RequireRole("Admin"));
    // options.AddPolicy("PolicyEmployee", policy => policy.RequireRole("Employee"));
    // options.AddPolicy("PolicyByPhoneNumber", policy => policy.Requirements.Add(new PolicyByPhoneNumberRequirement()));
});
```

## Sử dụng thuộc tính [Authorize] trên controller

- Thực hiện **Authorize** trên controller:



```
C# AccountController.cs 5, M ×  
Controllers > C# AccountController.cs > ...  
12     public class AccountController : Controller  
16     >     public AccountController(UserManager<ApplicationUser> userManager, IActionFilter...  
21  
22     >     [Authorize(Policy = nameof(SystemPermissions.AccountView))]  
23     >     [0 references]  
23     >     public async Task<ActionResult> Index() ...  
36  
37     >     [Authorize(Policy = nameof(SystemPermissions.AssignRole))]  
38     >     [0 references]  
38     >     public async Task<IActionResult> AssignRole(string userId) ...  
55
```



## Chạy thử nghiệm

- Tạo các Role
- Assign các claim bất kỳ cho các Role
- Gán Role cho tài khoản người dùng
- Đăng nhập hệ thống và truy cập vào tài nguyên để kiểm tra



## 4. THỰC HÀNH

Áp dụng nội dung về **Authentication** và **Authorization** với **WebAPI** theo các bước sau:

1. Tạo project **API**
2. Tích hợp **Identity** vào dự án
3. Tạo class **Employee** và tạo dữ liệu ngẫu nhiên
4. Sinh mã cho **API EmployeeController**
5. Xây dựng **AccountController** (**Register**, **Login**, **Logout**)



## 4. THỰC HÀNH

- Các nội dung 1 => 4 làm tương tự như project VicemMVC
- Không sinh mã cho các chức năng Register, Login, Logout.
- Xây dựng AccountController riêng



## 5. Xây dựng AccountController

- Xây dựng phương thức Register (Tương tự như VicemMVC: UserName, Password, ConfirmPassword, FullName)
- Xây dựng phương thức Login (Tương tự như VicemMVC: UserName, Password, RememberMe)
- Xây dựng phương thức Logout



### Xây dựng phương thức Register

Tạo class Models/ViewModels/RegisterVM.cs:

```
C:\ RegisterVM.cs U X
Models > ViewModels > C:\ RegisterVM.cs > ...
1  using System.ComponentModel.DataAnnotations;
2  namespace VicemAPI.Models.ViewModels
3  {
4      public class RegisterVM
5      {
6          [Required]
7          [DataType(DataType.EmailAddress)]
8          public string Email { get; set; }
9          [Required]
10         [DataType(DataType.Password)]
11         public string Password { get; set; }
12         [Required]
13         [DataType(DataType.Password)]
14         [Compare("Password", ErrorMessage = "The password and confirmation password do not match.")]
15         public string ConfirmPassword { get; set; }
16         [Required]
17         public string FullName { get; set; }
18     }
19 }
```



## Xây dựng phương thức Register

Tạo class Controllers/AccountController.cs:

```
[Route("api/[controller]")]
[ApiController]
1 reference
public class AccountController : ControllerBase
{
    2 references
    private readonly UserManager<ApplicationUser> _userManager;
    2 references
    private readonly SignInManager<ApplicationUser> _signInManager;
    0 references
    public AccountController(UserManager<ApplicationUser> userManager, SignInManager<ApplicationUser> signInManager)
    {
        _userManager = userManager;
        _signInManager = signInManager;
    }
}
```



## Xây dựng phương thức Register

Tạo phương thức Register:

```
[HttpPost("register")]
0 references
public async Task<IActionResult> Register([FromBody] RegisterVM model)
{
    if (ModelState.IsValid)
    {
        var user = new ApplicationUser { UserName = model.Email, Email = model.Email, FullName = model.FullName };
        var result = await _userManager.CreateAsync(user, model.Password);

        if (result.Succeeded)
        {
            await _signInManager.SignInAsync(user, isPersistent: false);
            return Ok(new { Message = "User registered successfully" });
        }

        return BadRequest(result.Errors);
    }

    return BadRequest(ModelState);
}
```



## Xây dựng phương thức Register

Sử dụng [Authorize] với EmployeeController:

```
You, 32 seconds ago | 1 author (You)
[Route("api/[controller]")]
[ApiController]
[Authorize]
1 reference
public class EmployeeController : ControllerBase
{}
```



## Xây dựng phương thức Register

Chạy chương trình và truy cập vào API Get Employee:

The screenshot shows a REST client interface with the following details:

- Method:** GET /api/Employee
- Parameters:** No parameters
- Responses:** Curl command:

```
curl -X 'GET' \
'http://localhost:5075/api/Employee' \
-H 'accept: text/plain'
```

Request URL:

```
http://localhost:5075/api/Employee
```

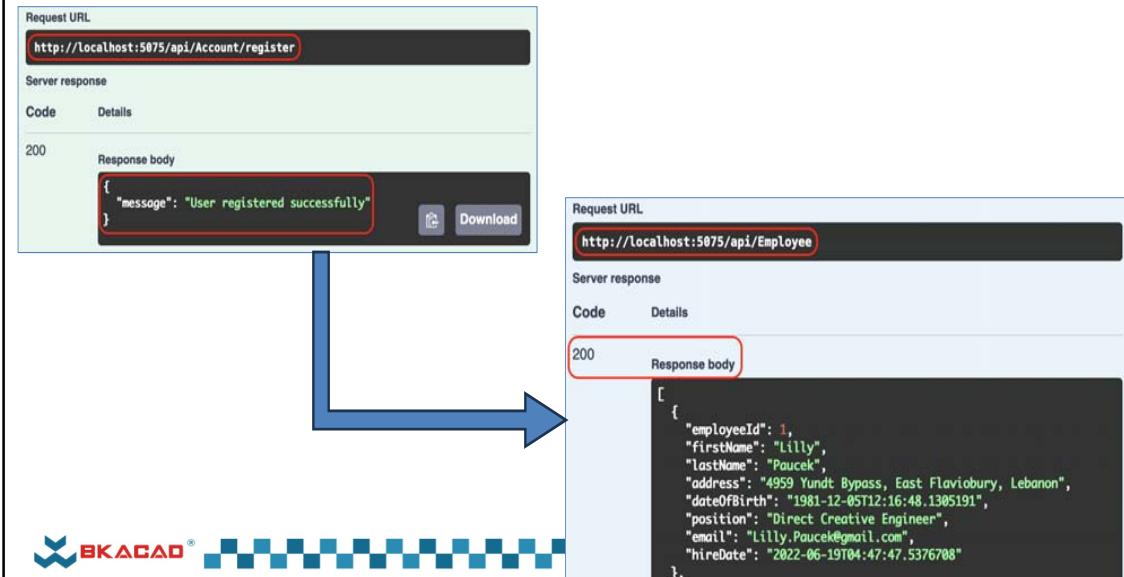
Server response:

Code	Details
404	Undocumented Error: Not Found
Response headers	
content-length: 0	
date: Tue, 18 Jun 2024 06:36:41 GMT	
server: Kestrel	



## Xây dựng phương thức Register

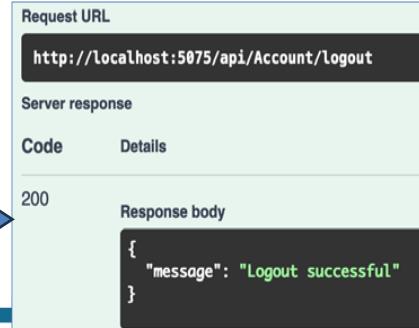
Đăng ký tài khoản và truy cập lại vào API Get Employee:



## Xây dựng phương thức Logout

- Xây dựng phương thức Logout:

```
[HttpPost("logout")]
0 references
public async Task<IActionResult> Logout()
{
    await _signInManager.SignOutAsync();
    return Ok(new { Message = "Logout successful" });
}
```



## Xây dựng phương thức Login

- Tạo class Models/ViewModels/LoginVM.cs:

```
public class LoginVM
{
    [Required]
    [EmailAddress]
    1 reference
    public string Email { get; set; }
    [Required]
    [DataType(DataType.Password)]
    1 reference
    public string Password { get; set; }
    1 reference
    public bool RememberMe { get; set; }
}
```



## Xây dựng phương thức Login

- Xây dựng phương thức LoginVM:

```
[HttpPost("login")]
0 references
public async Task<IActionResult> Login([FromBody] LoginVM model)
{
    if (ModelState.IsValid)
    {
        var result = await _signInManager.PasswordSignInAsync(model.Email,
            model.Password, model.RememberMe, lockoutOnFailure: false);

        if (result.Succeeded)
        {
            return Ok(new { Message = "Login successful" });
        }

        return Unauthorized();
    }
    return BadRequest(ModelState);
}
```



## Xây dựng phương thức Login

- Chạy chương trình và truy cập vào API Login:

Request URL  
`http://localhost:5075/api/Account/login`

Server response

Code	Details
401	<i>Undocumented</i> Error: Unauthorized

- Thay giá trị của RequireConfirmedAccount = `False` và thử lại:

```
builder.Services.AddIdentity<ApplicationUser, IdentityRole>(options  
    => options.SignIn(RequireConfirmedAccount = true)  
        .AddEntityFrameworkStores<ApplicationContext>();
```





## BẢO MẬT VÀ TỐI ƯU ỨNG DỤNG WEB .NET

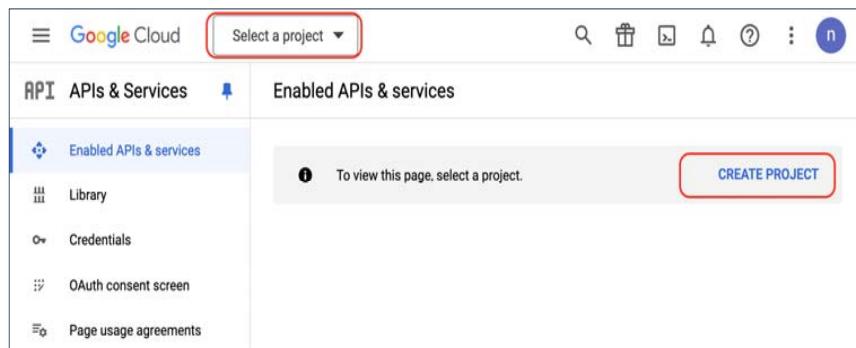


## ĐĂNG NHẬP VỚI TÀI KHOẢN GOOGLE



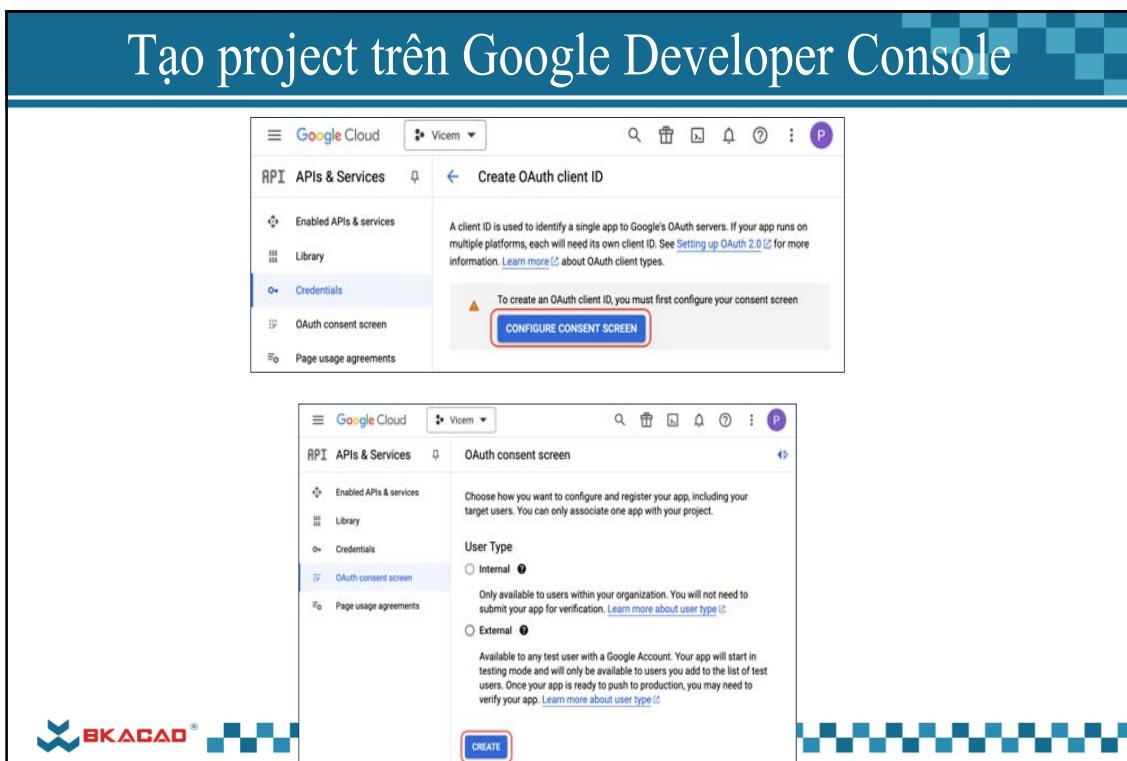
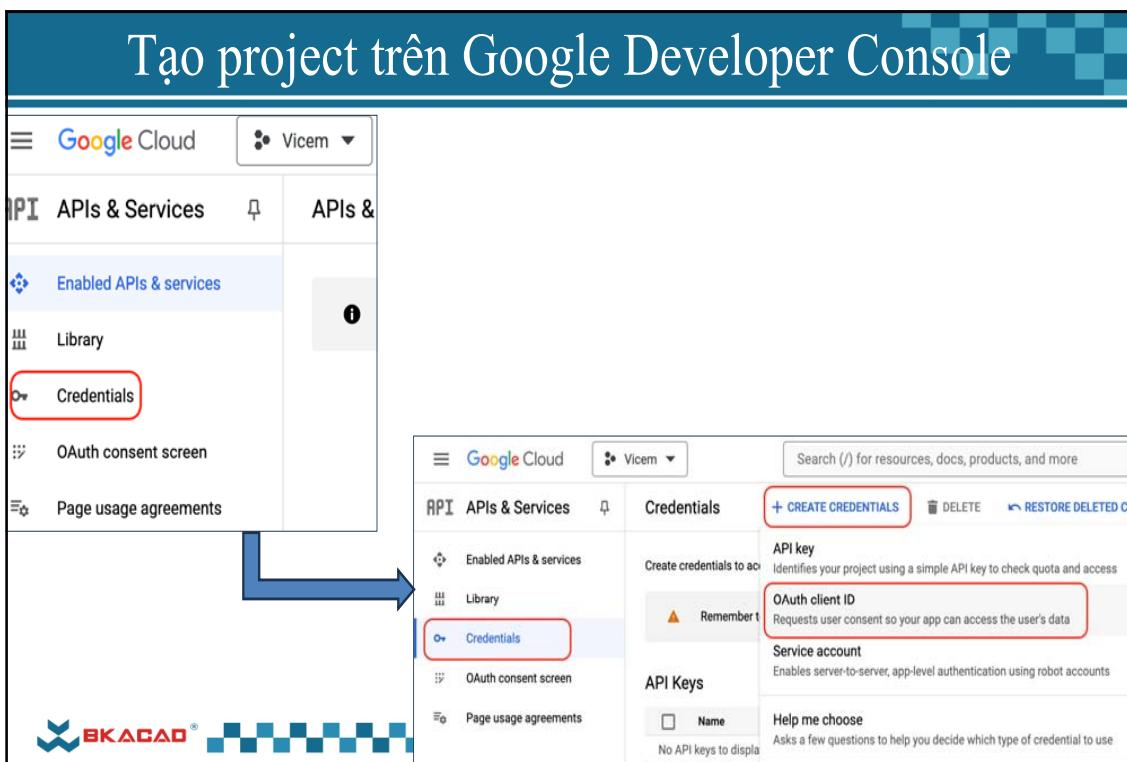
## Tạo project trên Google Developer Console

Đăng nhập vào tài khoản Google của bạn, sau đó truy cập vào đường dẫn: [Google Developer Console](#)

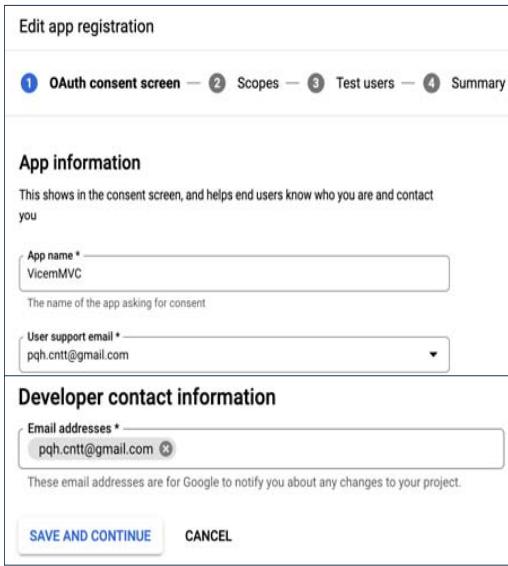


## Tạo project trên Google Developer Console

A screenshot of the "New Project" creation form in the Google Developer Console. The form has a header "New Project" and a warning message: "⚠ You have 12 projects remaining in your quota. Request an increase or delete projects. [Learn more](#)". Below the message is a "MANAGE QUOTAS" link. The "Project name \*" field is filled with "VicemMVC" and has a red border. The "Location \*" field shows "No organization" and has a "BROWSE" button. Below the location field, it says "Parent organization or folder". At the bottom, there are two buttons: a red-bordered "CREATE" button and a "CANCEL" button.

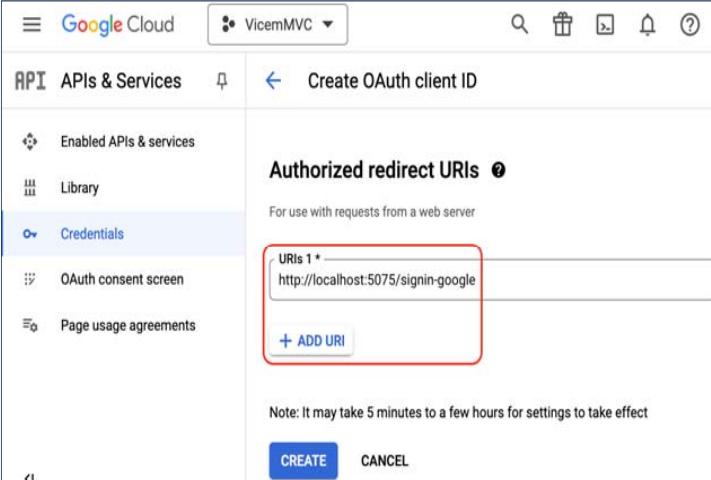


## Tạo project trên Google Developer Console



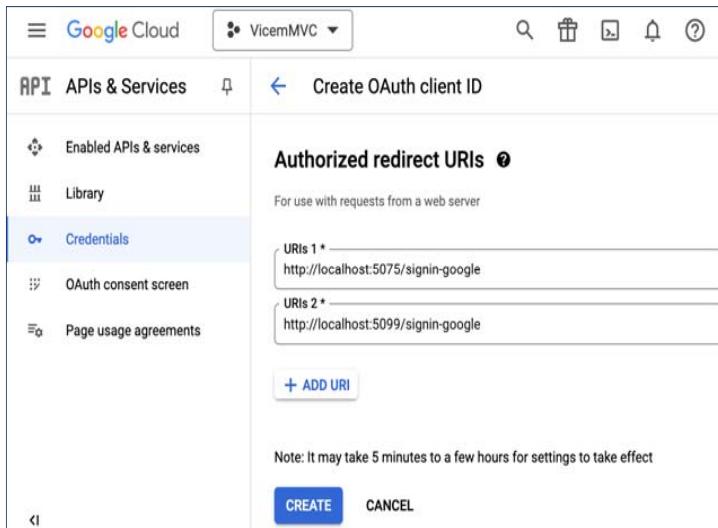
The screenshot shows the 'Edit app registration' page in the Google Developer Console. It's step 1 of 4: OAuth consent screen. The 'App information' section contains fields for 'App name' (VicemMVC) and 'User support email' (pgh.cntt@gmail.com). The 'Developer contact information' section has an 'Email addresses' field with pgh.cntt@gmail.com. Buttons at the bottom are 'SAVE AND CONTINUE' and 'CANCEL'.

## Tạo project trên Google Developer Console



The screenshot shows the 'Create OAuth client ID' screen in the Google Cloud Platform API & Services section. Under 'Authorized redirect URIs', the URL http://localhost:5075/signin-google is listed. A note at the bottom says 'Note: It may take 5 minutes to a few hours for settings to take effect'. Buttons at the bottom are 'CREATE' and 'CANCEL'.

## Tạo project trên Google Developer Console



## Cấu hình trên project VicemMVC

Chạy 2 lệnh sau để cài đặt Google ClientID và Google Client Secret trên project:

- dotnet user-secrets set "Authentication:Google:ClientId" "<client-id>"
- dotnet user-secrets set "Authentication:Google:ClientSecret" "<client-secret>"

Add package Google Authentication cho project: dotnet add package Microsoft.AspNetCore.Authentication.Google

## Cấu hình trên project VicemMVC

Cập nhật mã nguồn của file Program.cs:

```
Program.cs 2, M X
C:\Program.cs > Program > Main
10 internal class Program
11 {
12     private static void Main(string[] args)
13     {
14         var builder = WebApplication.CreateBuilder(args);
15         var configuration = builder.Configuration;
16         builder.Services.AddOptions();
17         var mailSettings = builder.Configuration.GetSection("MailSettings");
18         builder.Services.Configure<MailSettings>(mailSettings);
19         builder.Services.AddTransient<IEmailSender, SendMailService>();

        // Add services to the container.
21         var connectionString = builder.Configuration.GetConnectionString("DefaultConnection");
22         builder.Services.AddDbContext<ApplicationDbContext>(options =>
23             options.UseSqlite(connectionString));
24             options.UseSqlite(connectionString));
25         builder.Services.AddDatabaseDeveloperPageExceptionFilter();
26         builder.Services.AddRazorPages();
27         builder.Services.AddIdentity< ApplicationUser, IdentityRole>(options => options.SignIn.
28             .AddEntityFrameworkStores<ApplicationDbContext>());
29         builder.Services.AddControllersWithViews();
30     }

    builder.Services.AddAuthentication().AddGoogle(googleOptions =>
31     {
32         googleOptions.ClientId = configuration["Authentication:Google:ClientId"];
33         googleOptions.ClientSecret = configuration["Authentication:Google:ClientSecret"];
34     });
35 }
```



## Cấu hình trên project VicemMVC

VicemMVCIdentity Home Privacy Employee Account Role Member

### Log in

Use a local account to log in. Use another service to log in.

Email  Google

Password

Remember me?

[Forgot your password?](#)

[Register as a new user](#)

[Resend email confirmation](#)



## Cấu hình trên project VicemMVC

The screenshot shows the Google sign-in interface. At the top left is a 'Sign in with Google' button. In the center, there's a 'Sign in' button with the text 'to continue to VicemMVC'. To the right is an input field for 'Email or phone' containing 'it3.cntt.humg@gmail.com', with a 'Forgot email?' link below it. A note at the bottom states: 'To continue, Google will share your name, email address, language preference, and profile picture with VicemMVC.' Below the input field are 'Create account' and 'Next' buttons. The 'Next' button is highlighted with a red border.

Below this, another Google sign-in window is shown. It has a 'Sign in with Google' button at the top left. In the center, it says 'Sign in to VicemMVC' and displays the email 'it3.cntt.humg@gmail.com'. To the right, it says 'By continuing, Google will share your name, email address, language preference, and profile picture with VicemMVC. See VicemMVC's Privacy Policy and Terms of Service.' and 'You can manage Sign in with Google in your [Google Account](#)'. At the bottom are 'Cancel' and 'Continue' buttons, with 'Continue' also highlighted with a red border.

At the bottom left of the entire screenshot area is the BKACAO logo.

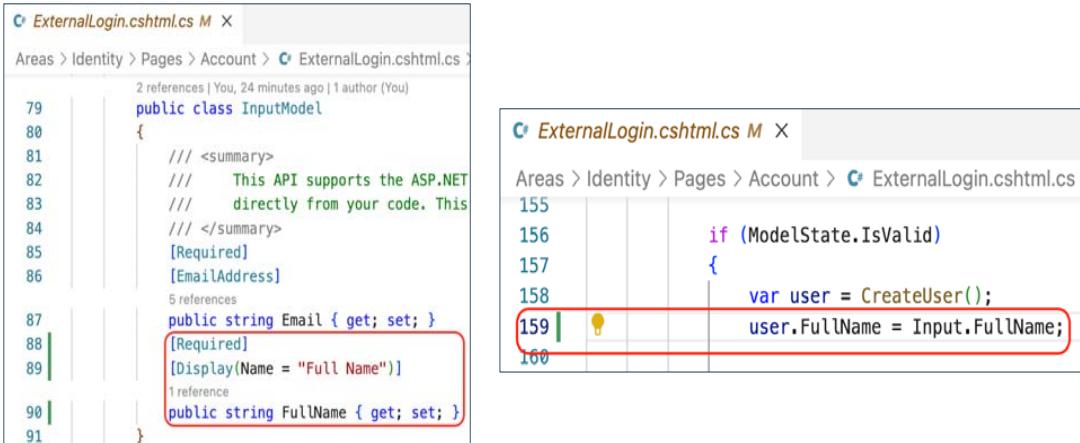
## Cấu hình trên project VicemMVC

The screenshot shows a registration page for the VicemMVC identity system. At the top, there's a navigation bar with links for 'VicemMVCIdentity', 'Home', 'Privacy', 'Employee', 'Account', 'Role', 'Member', 'Register', and 'Login'. The main content area has a title 'Register' and a subtitle 'Associate your Google account.' Below this is a message: 'You've successfully authenticated with Google. Please enter an email address for this site below and click the Register button to finish logging in.' An input field contains the email 'it3.cntt.humg@gmail.com'. At the bottom is a large blue 'Register' button.

At the bottom left of the entire screenshot area is the BKACAO logo.

## Cấu hình trên project VicemMVC

Cập nhật mã nguồn của file ExternalLogin.cshtml.cs:



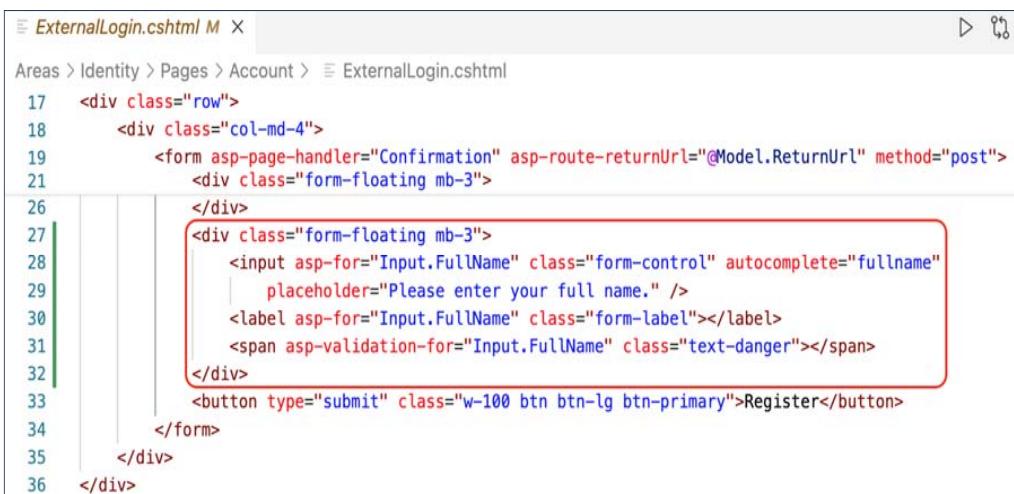
```
ExternalLogin.cshtml.cs M X
Areas > Identity > Pages > Account > ExternalLogin.cshtml.cs
2 references | You, 24 minutes ago | 1 author (You)
public class InputModel
{
    /// <summary>
    ///     This API supports the ASP.NET
    ///     directly from your code. This
    /// </summary>
    [Required]
    [EmailAddress]
    public string Email { get; set; }
    [Required]
    [Display(Name = "Full Name")]
    public string FullName { get; set; }
}

if (ModelState.IsValid)
{
    var user = CreateUser();
    user.FullName = Input.FullName;
}
```



## Cấu hình trên project VicemMVC

Cập nhật mã nguồn của file ExternalLogin.cshtml:



```
ExternalLogin.cshtml M X
Areas > Identity > Pages > Account > ExternalLogin.cshtml
<div class="row">
    <div class="col-md-4">
        <form asp-page-handler="Confirmation" asp-route-returnUrl="@Model.ReturnUrl" method="post">
            <div class="form-floating mb-3">
                </div>
                <div class="form-floating mb-3">
                    <input asp-for="Input.FullName" class="form-control" autocomplete="fullname" placeholder="Please enter your full name." />
                    <label asp-for="Input.FullName" class="form-label"></label>
                    <span asp-validation-for="Input.FullName" class="text-danger"></span>
                </div>
                <button type="submit" class="w-100 btn btn-lg btn-primary">Register</button>
            </form>
        </div>
    </div>
</div>
```



# Cấu hình trên project VicemMVC

The screenshot shows a registration page for the VicemMVC Identity system. At the top, there is a navigation bar with links for Home, Privacy, Employee, Account, Role, Member, Register, and Login. Below the navigation bar, the word "Register" is prominently displayed, followed by the instruction "Associate your Google account." A message indicates successful authentication with Google, prompting the user to enter an email address for the site and click the Register button. Two input fields are shown: one for "Email" containing "it3.cntt.hung@gmail.com" and another for "Full Name" containing "Pham Thanh Cong". A blue "Register" button is at the bottom of the form. The page is framed by a decorative blue and white checkered border.