

HƯỚNG DẪN CÀI ĐẶT VÀ SỬ DỤNG MÃ NGUỒN

A. Thông tin tài khoản sử dụng trong hệ thống

Tài khoản để tạo lập và đăng nhập cơ sở dữ liệu tương ứng với connection string:

Tài khoản: sa

Mật khẩu: Admin123!

Tài khoản mặc định của ứng dụng với phân quyền admin trong hệ thống:

Tài khoản: admin

Mật khẩu: admin123

B. Hướng dẫn cài đặt và sử dụng mã nguồn

1. Cơ sở dữ liệu và API

Cơ sở dữ liệu và API của hệ thống có thể được cài đặt và sử dụng trên các hệ điều hành khác nhau nhưng sẽ có sự khác biệt nhất định trong các bước thực hiện. Đối với hệ điều hành thuộc họ **Windows**, khuyến khích sử dụng **Cách 1: Cài đặt API thông qua Visual Studio**. Đối với hệ điều hành thuộc họ **Linux** và **MacOS**, chỉ có thể sử dụng **Cách 2: Cài đặt API thông qua Docker**. (Hệ điều hành thuộc họ **Windows** có thể sử dụng **Cách 2** để tiết kiệm tài nguyên cho máy có cấu hình thấp).

1.1 Phần mềm cần thiết

- ASP.NET Core Runtime 7.0 (ASP.NET Core Module v2)
- DotNet Core SDK 7.0
- Azure Data Studio

Phần mềm cần thiết cho cách 1:

- SQL Server (Khuyến khích sử dụng phiên bản 2019)
- Visual Studio (Khuyến khích sử dụng phiên bản 2022)

Phần mềm cần thiết cho cách 2:

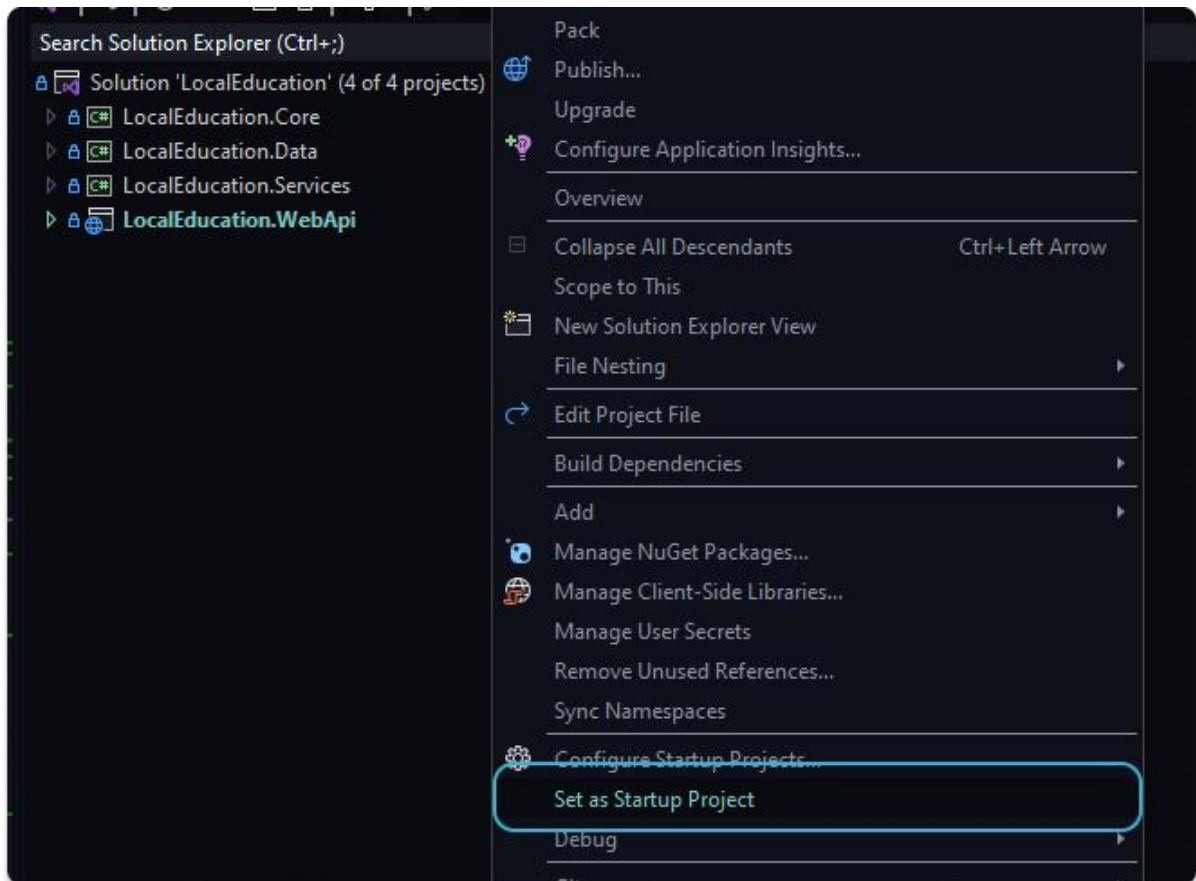
- Docker
- Entity Framework Core .NET Command-line Tools 7.0.0

1.2 Hướng dẫn cài đặt

Cách 1: Cài đặt API thông qua Visual Studio

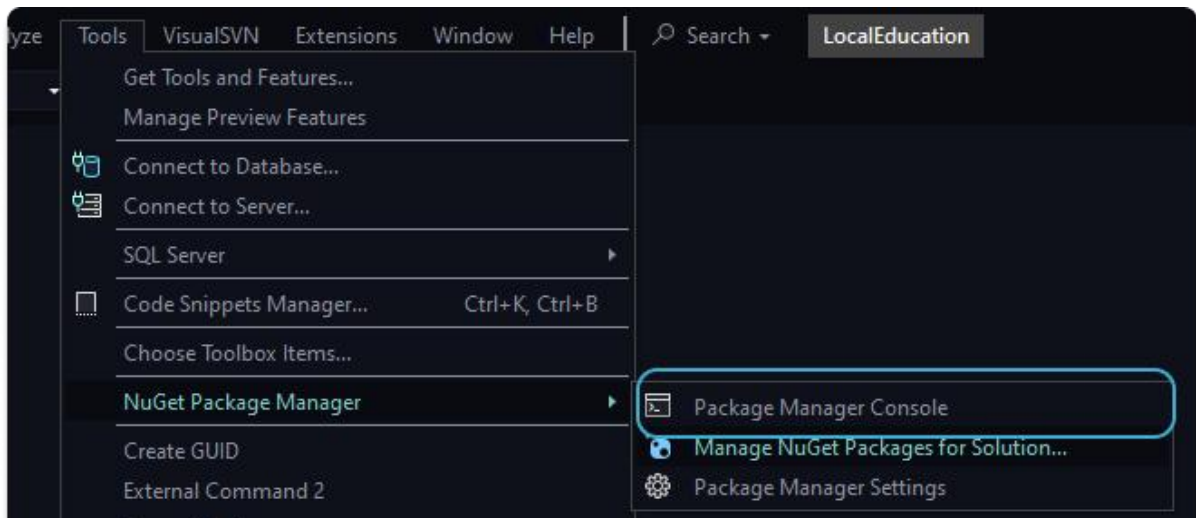
Bước 1: Tạo cơ sở dữ liệu cho hệ thống

Mở dự án API theo đường dẫn `/local-education-api` bằng phần mềm **Visual Studio**. Tại cửa sổ **Solution Explorer** của dự án, click chuột phải vào dự án con **LocalEducation.WebApi** và chọn **Set as Startup Project**:



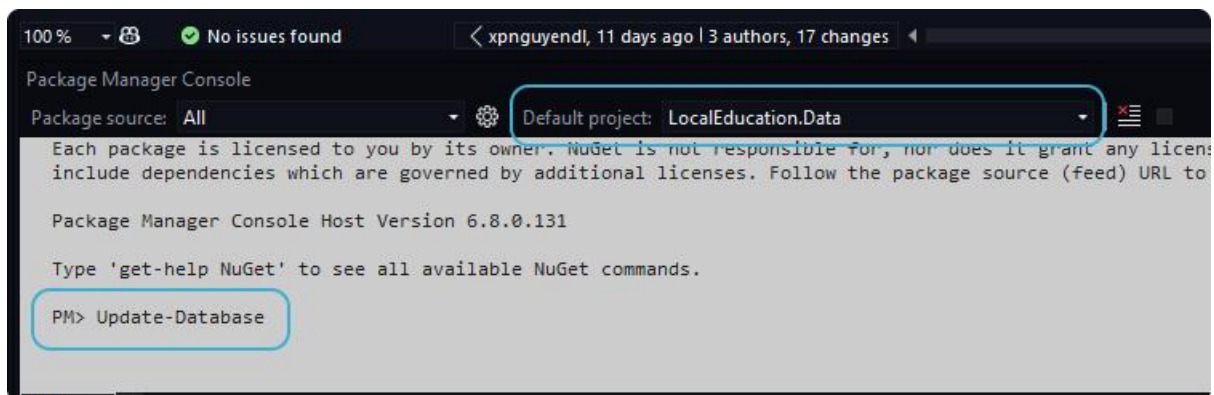
Hình 1.1: Cài đặt dự án khởi động mặc định

Mở cửa sổ **Package Manager Console** bằng cách nhấn tổ hợp phím **Ctrl + `** hoặc nhấn vào bảng chọn **Tools > NuGet Package Manager > Package Manager Console**:



Hình 1.2: Mở cửa sổ Package Manager Console

Tại cửa sổ **Package Manager Console**, chọn **Default project** là **LocalEducation.Data**. Sau đó chạy lệnh **Update-Database** để khởi tạo cơ sở dữ liệu:



Hình 1.3: Khởi tạo cơ sở dữ liệu

Bước 2: Chạy thủ tục cho cơ sở dữ liệu

Sau khi khởi tạo cơ sở dữ liệu, đăng nhập vào cơ sở dữ liệu bằng **Azure Data Studio** với thông tin:

Tài khoản: **sa**

Mật khẩu: **Admin123!**

Database: **LocalEducation**

Trust server certificate: **True**

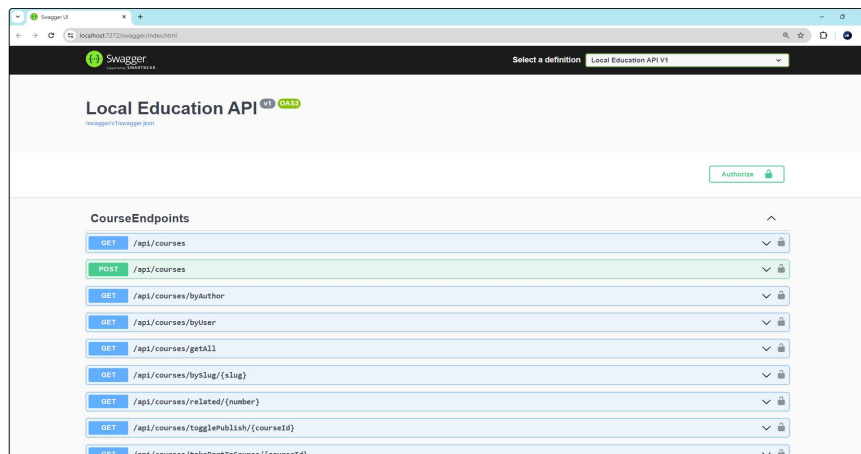
Hình 1.4: Đăng nhập vào cơ sở dữ liệu với thông tin phù hợp

Sau khi đăng nhập thành công, mở tập tin **StoredProcedures.sql** theo đường dẫn **/local-education-database/Stored Procedures/** bằng phần mềm **Azure Data Studio** và nhấn chọn nút **Run** trên thanh công cụ hoặc nhấn phím **F5** để chạy:

Hình 1.5: Chạy thử tục cho cơ sở dữ liệu

Bước 3: Chạy dự án API

Quay lại dự án trong **Visual Studio**, nhấn tổ hợp phím **Ctrl + F5** để chạy chương trình. Sau khi đã tải xong giao diện **Swagger** lên trên trình duyệt là đã có thể sử dụng được API:



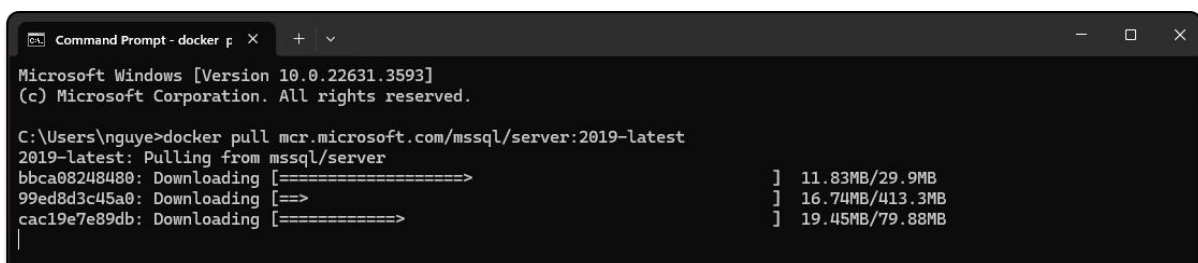
Hình 1.6: API được hiển thị thông qua Swagger

Cài đặt API thông qua Docker:

Bước 1: Tải hệ cơ sở dữ liệu SQL Server trên Docker

Mở cửa sổ dòng lệnh (**cmd** hoặc **terminal**) và chạy lệnh sau để tải image của hệ cơ sở dữ liệu SQL Server 2019 từ máy chủ:

```
docker pull mcr.microsoft.com/mssql/server:2019-latest
```

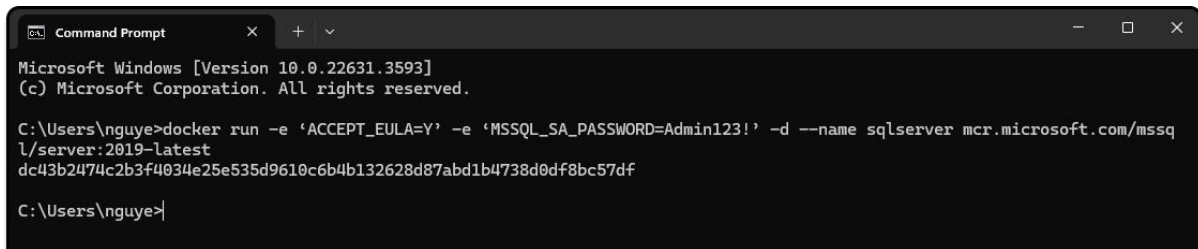


Hình 1.7: Tải thành công image SQL Server 2019 từ máy chủ

Bước 2: Khởi tạo và chạy hệ cơ sở dữ liệu với thông tin tài khoản **sa** phù hợp với Connection String của dự án API

Sau khi tải xong image, tiếp tục chạy lệnh sau để khởi tạo hệ cơ sở dữ liệu phù hợp với dự án:

```
| docker run -e 'ACCEPT_EULA=Y' -e 'MSSQL_SA_PASSWORD=Admin123!'
1433:1433 -d --name sqlserver mcr.microsoft.com/mssql/server:2019-latest
```



```
Command Prompt
Microsoft Windows [Version 10.0.22631.3593]
(c) Microsoft Corporation. All rights reserved.

C:\Users\nguye>docker run -e 'ACCEPT_EULA=Y' -e 'MSSQL_SA_PASSWORD=Admin123!' -d --name sqlserver mcr.microsoft.com/mssql
l/server:2019-latest
dc43b2474c2b3f4034e25e535d9610c6b4b132628d87abd1b4738d0df8bc57df

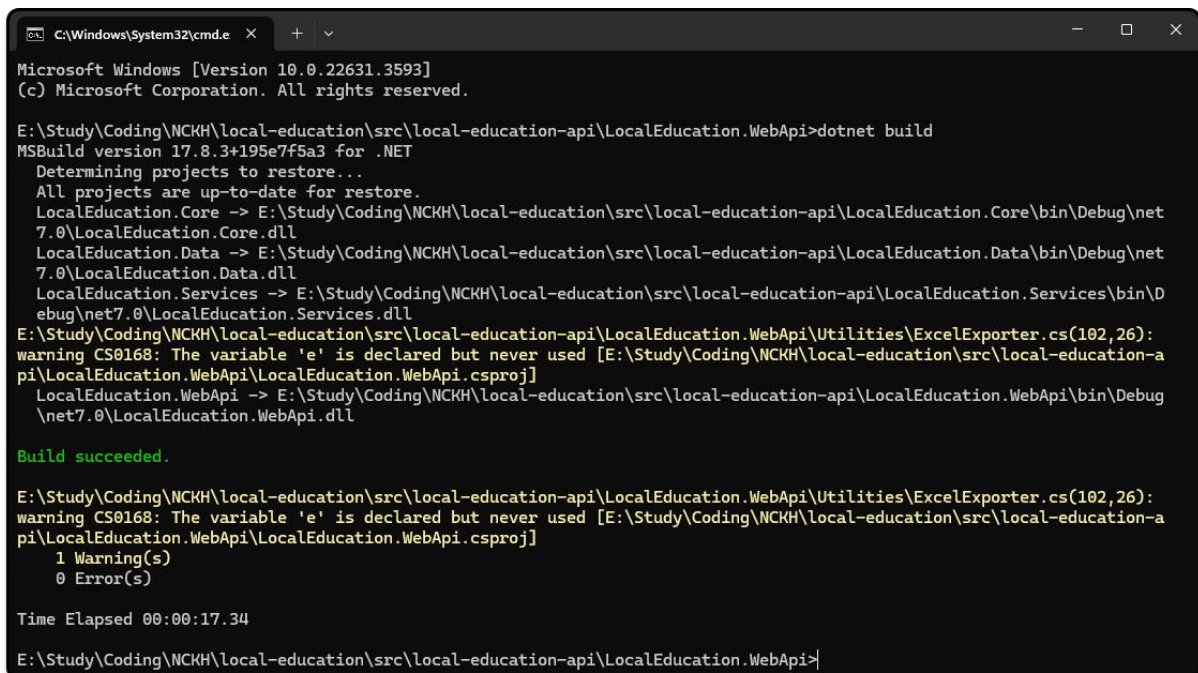
C:\Users\nguye>
```

Hình 1.8: Khởi tạo và chạy thành công hệ cơ sở dữ liệu cho hệ thống

Bước 3: Tải các gói thư viện và cài đặt dự án API

Sau khi khởi tạo hệ cơ sở dữ liệu, mở cửa sổ dòng lệnh (**cmd** hoặc **terminal**) theo đường dẫn chứa mã nguồn của dự án API (**/local-education-api/LocalEducation.WebApi**) và chạy lệnh sau để tải các gói thư viện cần thiết và cài đặt dự án:

```
| dotnet build
```



```
C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.22631.3593]
(c) Microsoft Corporation. All rights reserved.

E:\Study\Coding\NCKH\local-education\src\local-education-api\LocalEducation.WebApi>dotnet build
MSBuild version 17.8.3+195e7f5a3 for .NET
Determining projects to restore...
All projects are up-to-date for restore.
LocalEducation.Core -> E:\Study\Coding\NCKH\local-education\src\local-education-api\LocalEducation.Core\bin\Debug\net
7.0\LocalEducation.Core.dll
LocalEducation.Data -> E:\Study\Coding\NCKH\local-education\src\local-education-api\LocalEducation.Data\bin\Debug\net
7.0\LocalEducation.Data.dll
LocalEducation.Services -> E:\Study\Coding\NCKH\local-education\src\local-education-api\LocalEducation.Services\bin\D
ebug\net7.0\LocalEducation.Services.dll
E:\Study\Coding\NCKH\local-education\src\local-education-api\LocalEducation.WebApi\Utilities\ExcelExporter.cs(102,26):
warning CS0168: The variable 'e' is declared but never used [E:\Study\Coding\NCKH\local-education\src\local-education-a
pi\LocalEducation.WebApi\LocalEducation.WebApi.csproj]
LocalEducation.WebApi -> E:\Study\Coding\NCKH\local-education\src\local-education-api\LocalEducation.WebApi\bin\Debug
\net7.0\LocalEducation.WebApi.dll

Build succeeded.

E:\Study\Coding\NCKH\local-education\src\local-education-api\LocalEducation.WebApi\Utilities\ExcelExporter.cs(102,26):
warning CS0168: The variable 'e' is declared but never used [E:\Study\Coding\NCKH\local-education\src\local-education-a
pi\LocalEducation.WebApi\LocalEducation.WebApi.csproj]
1 Warning(s)
0 Error(s)

Time Elapsed 00:00:17.34

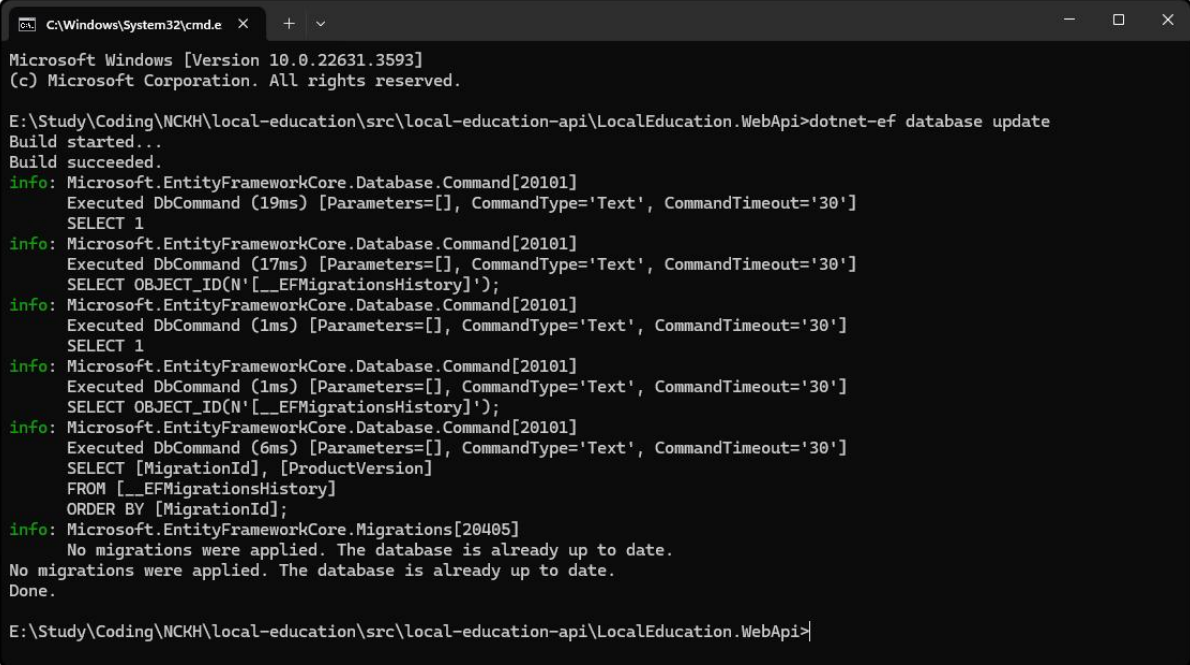
E:\Study\Coding\NCKH\local-education\src\local-education-api\LocalEducation.WebApi>
```

Hình 1.9: Cài đặt thành công dự án API

Bước 4: Tạo cơ sở dữ liệu cho hệ thống

Sau khi cài đặt xong dự án API, tiếp tục chạy lệnh sau để tạo cơ sở dữ liệu:

```
dotnet-ef database update
```



```
C:\Windows\System32\cmd.e  X  +  v
Microsoft Windows [Version 10.0.22631.3593]
(c) Microsoft Corporation. All rights reserved.

E:\Study\Coding\NCKH\local-education\src\local-education-api\LocalEducation.WebApi>dotnet-ef database update
Build started...
Build succeeded.
info: Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (19ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      SELECT 1
info: Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (17ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      SELECT OBJECT_ID(N'[__EFMigrationsHistory]');
info: Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (1ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      SELECT 1
info: Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (1ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      SELECT OBJECT_ID(N'[__EFMigrationsHistory]');
info: Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (6ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      SELECT [MigrationId], [ProductVersion]
      FROM [__EFMigrationsHistory]
      ORDER BY [MigrationId];
info: Microsoft.EntityFrameworkCore.Migrations[20405]
      No migrations were applied. The database is already up to date.
No migrations were applied. The database is already up to date.
Done.

E:\Study\Coding\NCKH\local-education\src\local-education-api\LocalEducation.WebApi>
```

Hình 1.10: Tạo thành công cơ sở dữ liệu

Bước 5: Chạy thử tục cho cơ sở dữ liệu

Sau khi tạo thành công cơ sở dữ liệu, thực hiện tương tự **Bước 2** ở phần **Cài đặt API thông qua Visual Studio** để chạy các thủ tục cần thiết cho cơ sở dữ liệu của hệ thống.

Bước 6: Chạy dự án API

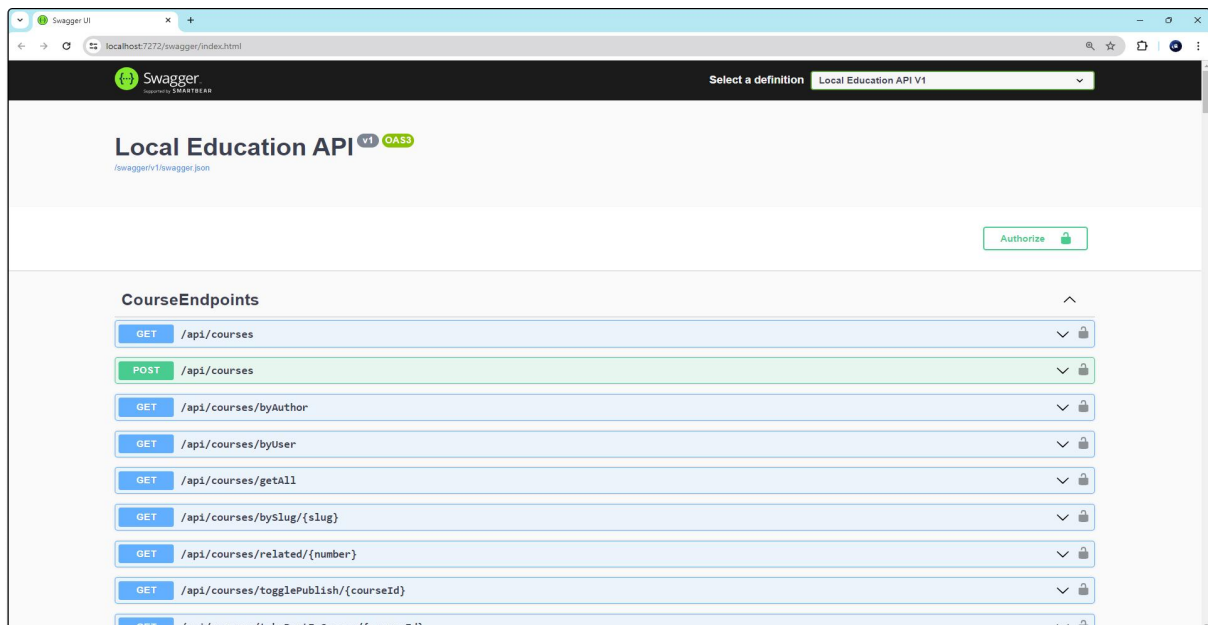
Sau khi chạy xong các thủ tục, tiếp tục chạy lệnh sau để chạy dự án:

```
dotnet run --urls=https://localhost:7272
```

```
C:\Windows\System32\cmd.exe
E:\Study\Coding\NCKH\local-education\src\local-education-api\LocalEducation.WebApi>dotnet run --urls=https://localhost:7272
Building...
info: Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (17ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      SELECT 1
info: Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (48ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      IF EXISTS
      (
        (SELECT *
         FROM [sys].[objects] o
         WHERE [o].[type] = 'U'
         AND [o].[is_ms_shipped] = 0
         AND NOT EXISTS (SELECT *
                        FROM [sys].[extended_properties] AS [ep]
                        WHERE [ep].[major_id] = [o].[object_id]
                        AND [ep].[minor_id] = 0
                        AND [ep].[class] = 1
                        AND [ep].[name] = N'microsoft_database_tools_support'
                        )
        )
      )
      SELECT 1 ELSE SELECT 0
info: Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (4ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      SELECT CASE
      WHEN EXISTS (
        SELECT 1
        FROM [Users] AS [u]
      ) THEN CAST(1 AS bit)
      ELSE CAST(0 AS bit)
```

Hình 1.11: Chạy thành công dự án API

Sau khi cửa sổ dòng lệnh hiển thị thông tin tương tự **hình 1.11** là dự án API đã chạy thành công, có thể truy cập vào đường dẫn <https://localhost:7272> để kiểm tra.



Hình 1.12: API hiển thị thông qua Swagger

2. Ứng dụng web

2.1 Phần mềm cần thiết

- Node **v20** trở lên (khuyến khích sử dụng phiên bản **20.11.0 LTS**)
- Yarn Package Manager (khuyến khích sử dụng phiên bản **1.22.21**)

2.2 Hướng dẫn cài đặt

Bước 1: Cài đặt các gói thư viện cho ứng dụng web

Mở cửa sổ dòng lệnh (**cmd** hoặc **terminal**) tại thư mục chứa mã nguồn của ứng dụng web (**/local-education-web**) và chạy một trong các lệnh:

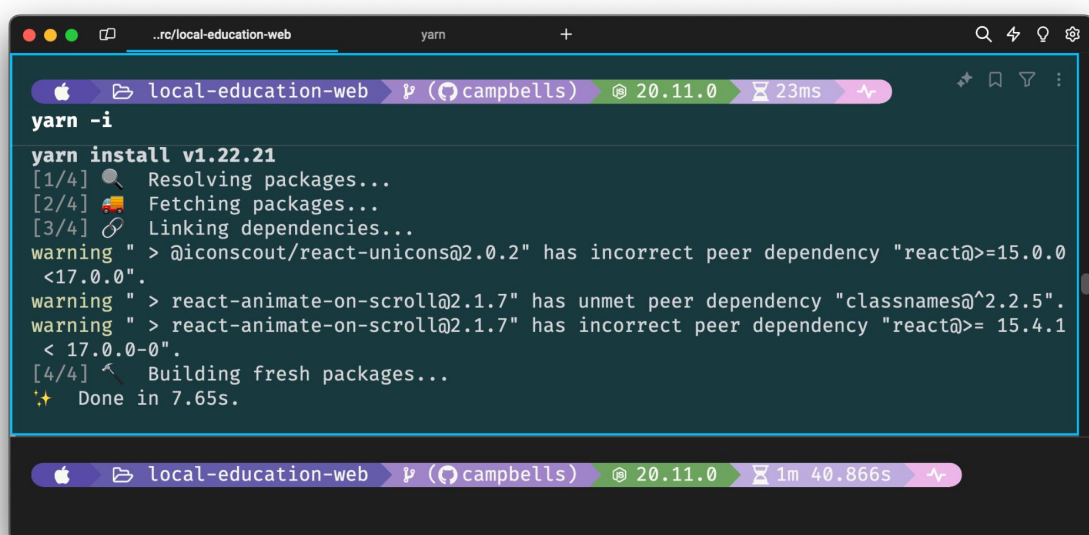
```
-----  
| yarn |  
-----
```

hoặc:

```
-----  
| yarn -i |  
-----
```

hoặc:

```
-----  
| yarn install |  
-----
```

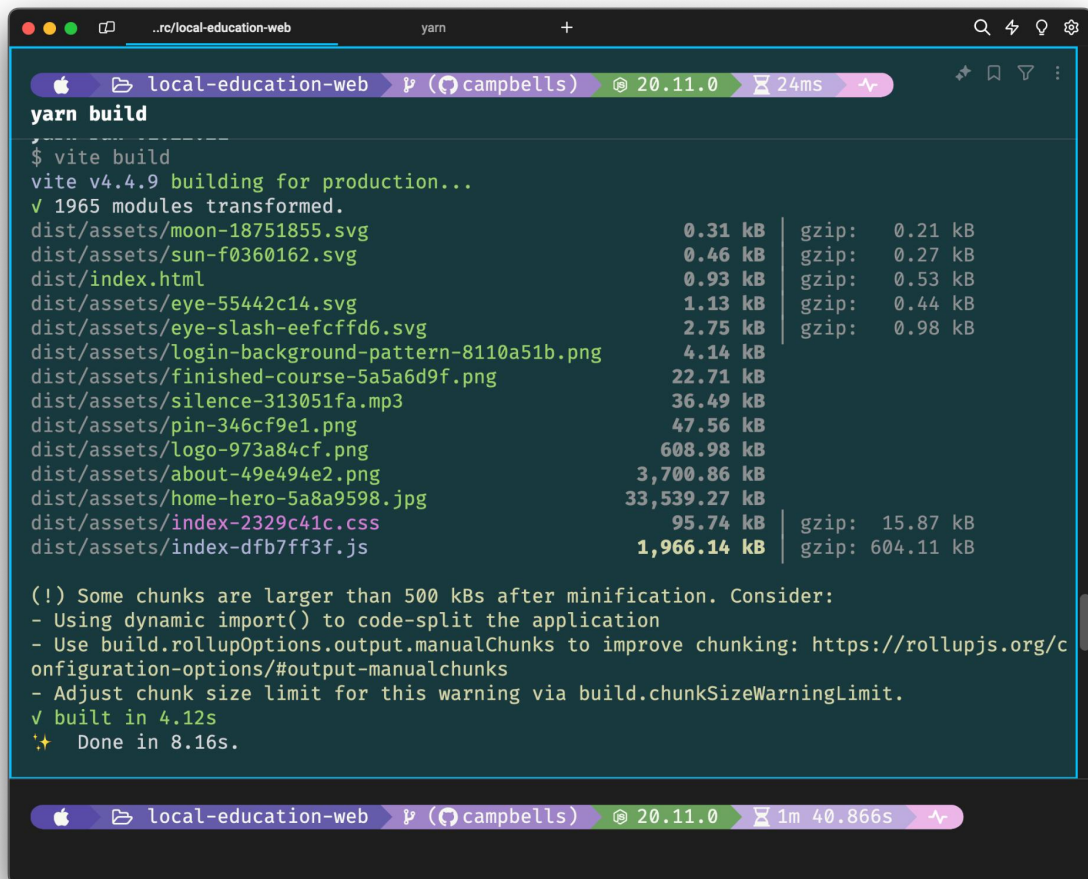


Hình 2.1 Cài đặt thành công các gói thư viện cho ứng dụng web

Bước 2: Cài đặt ứng dụng web

Sau khi hoàn tất, tiếp tục chạy lệnh sau để cài đặt ứng dụng web:

```
yarn build
```



```
yarn build
$ vite build
vite v4.4.9 building for production...
✓ 1965 modules transformed.
dist/assets/moon-18751855.svg                0.31 kB | gzip: 0.21 kB
dist/assets/sun-f0360162.svg                0.46 kB | gzip: 0.27 kB
dist/index.html                             0.93 kB | gzip: 0.53 kB
dist/assets/eye-55442c14.svg                1.13 kB | gzip: 0.44 kB
dist/assets/eye-slash-eefcffd6.svg          2.75 kB | gzip: 0.98 kB
dist/assets/login-background-pattern-8110a51b.png 4.14 kB
dist/assets/finished-course-5a5a6d9f.png    22.71 kB
dist/assets/silence-313051fa.mp3           36.49 kB
dist/assets/pin-346cf9e1.png                47.56 kB
dist/assets/logo-973a84cf.png              608.98 kB
dist/assets/about-49e494e2.png              3,700.86 kB
dist/assets/home-hero-5a8a9598.jpg          33,539.27 kB
dist/assets/index-2329c41c.css              95.74 kB | gzip: 15.87 kB
dist/assets/index-dfb7ff3f.js               1,966.14 kB | gzip: 604.11 kB

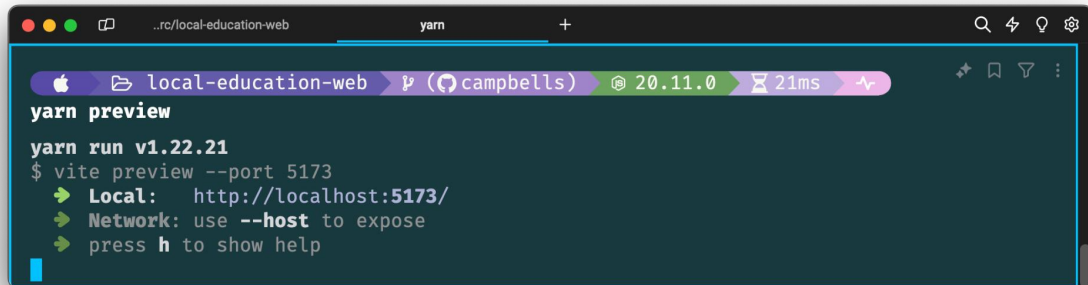
(!) Some chunks are larger than 500 kB after minification. Consider:
- Using dynamic import() to code-split the application
- Use build.rollupOptions.output.manualChunks to improve chunking: https://rollupjs.org/configuration-options/#output-manualchunks
- Adjust chunk size limit for this warning via build.chunkSizeWarningLimit.
✓ built in 4.12s
!⚡ Done in 8.16s.
```

Hình 2.2 Cài đặt thành công ứng dụng web

Bước 3: Chạy ứng dụng web

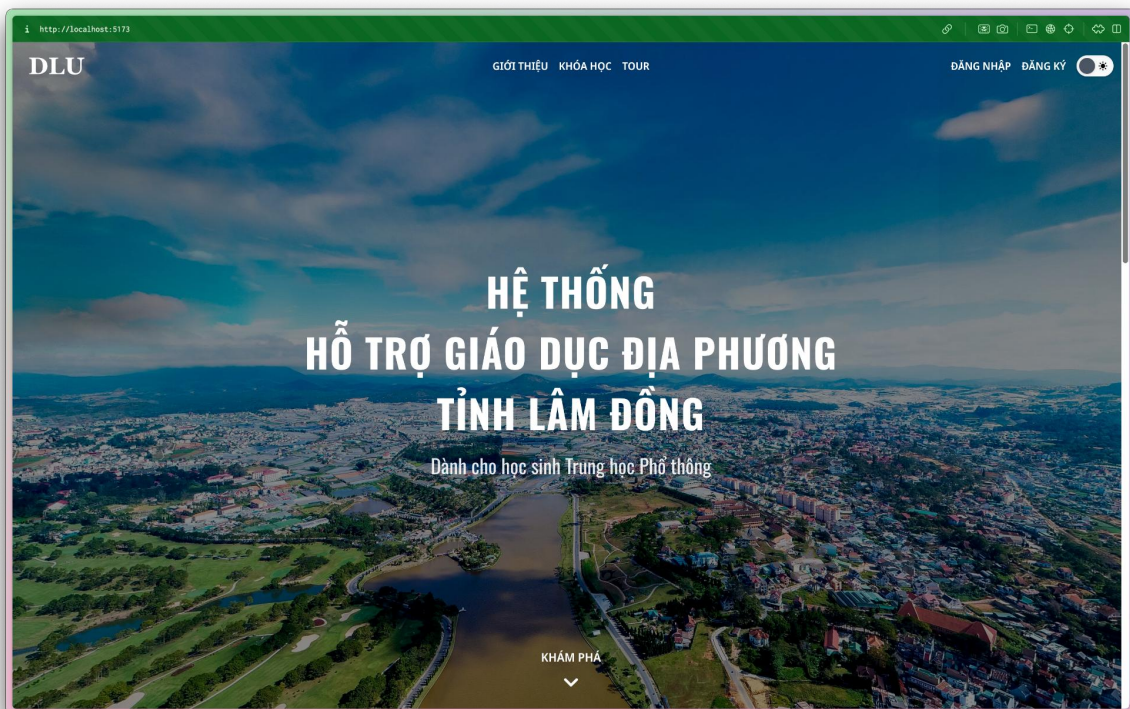
Sau khi hoàn tất, tiếp tục chạy lệnh sau để chạy ứng dụng web:

```
yarn preview
```



Hình 2.3: Chạy thành công ứng dụng web

Sau khi cửa sổ dòng lệnh hiển thị thông tin tương tự như **hình 2.3** là ứng dụng web đã chạy thành công, có thể truy cập thông qua đường dẫn <http://localhost:5173> để sử dụng.



Hình 2.4: Ứng dụng web

3. Ứng dụng di động

3.1 Phần mềm cần thiết

- Android studio: Trình quản lý và kết nối thiết bị di động
- Flutter đã cấu hình đầy đủ, có thể kiểm tra thông qua lệnh

```
| flutter doctor -v |
```

- ngrok (nếu chưa có domain đã được publish trên internet)

3.2 Hướng dẫn cài đặt

a) Kết nối thiết bị di động

Để cài đặt ứng dụng di động flutter, cần một thiết bị di động để có thể khởi chạy ứng dụng. Hiện nay, có 2 loại thiết bị di động phổ biến được sử dụng, đó là trình giả lập thiết bị di động (emulator) hoặc thiết bị di động vật lý.

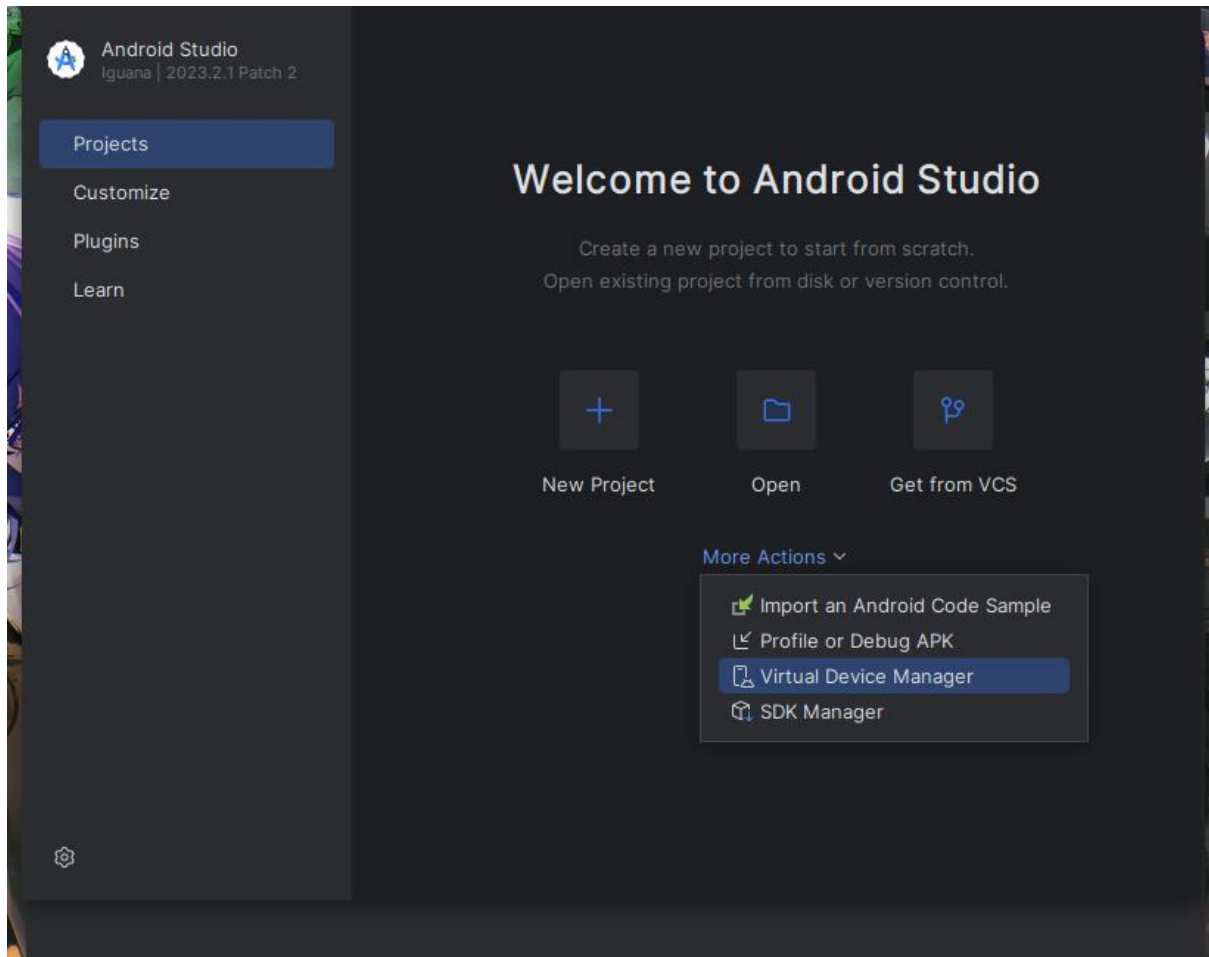
Khuyến nghị sử dụng trình giả lập nếu máy tính đã đạt đủ yêu cầu về cấu hình và phần cứng. (Có thể tham khảo ở Link sau:

<https://developer.android.com/codelabs/basic-android-kotlin-compose-install-android-studio#0>)

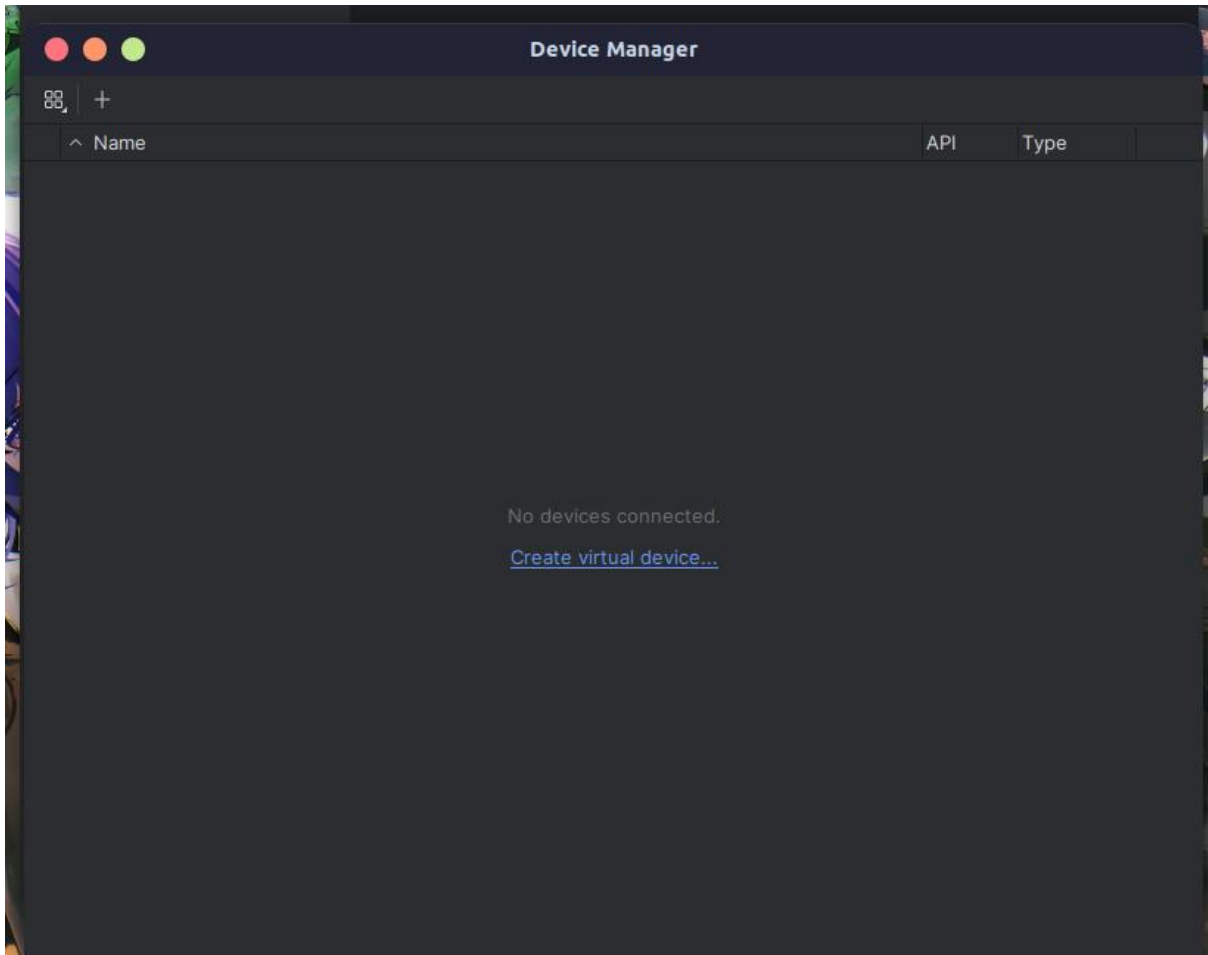
Nếu máy tính quá yếu hoặc không đáp ứng đầy đủ yêu cầu về phần cứng, nên sử dụng thiết bị vật lý. Tuy nhiên, rủi ro thiết bị di động bị ảnh hưởng bởi các lỗi nghiêm trọng trong quá trình khởi chạy là rất cao. Do đó, cần cân nhắc giữa việc sử dụng thiết bị vật lý hoặc sử dụng thiết bị giả lập

Đối với thiết bị giả lập

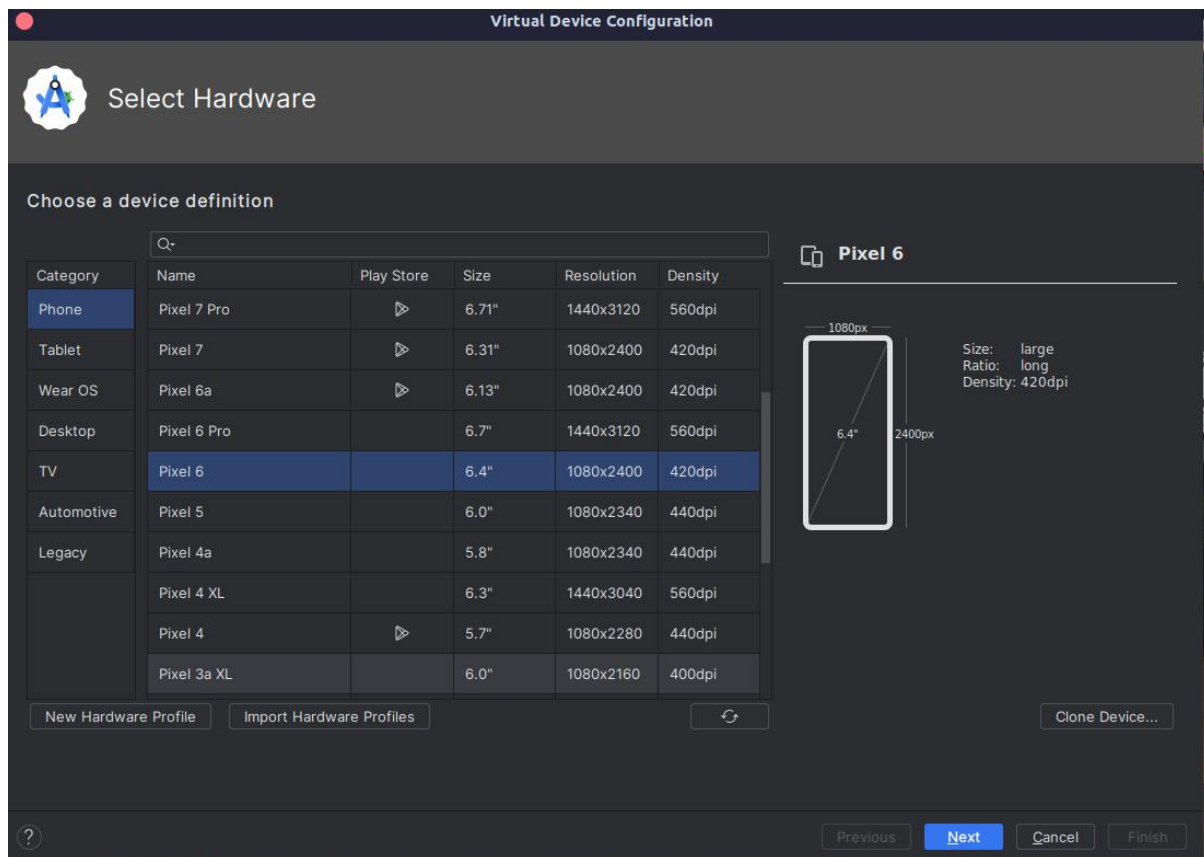
Bước 1: Mở Android Studio, truy cập vào phần Quản lý thiết bị ảo (Virtual Device Manager)



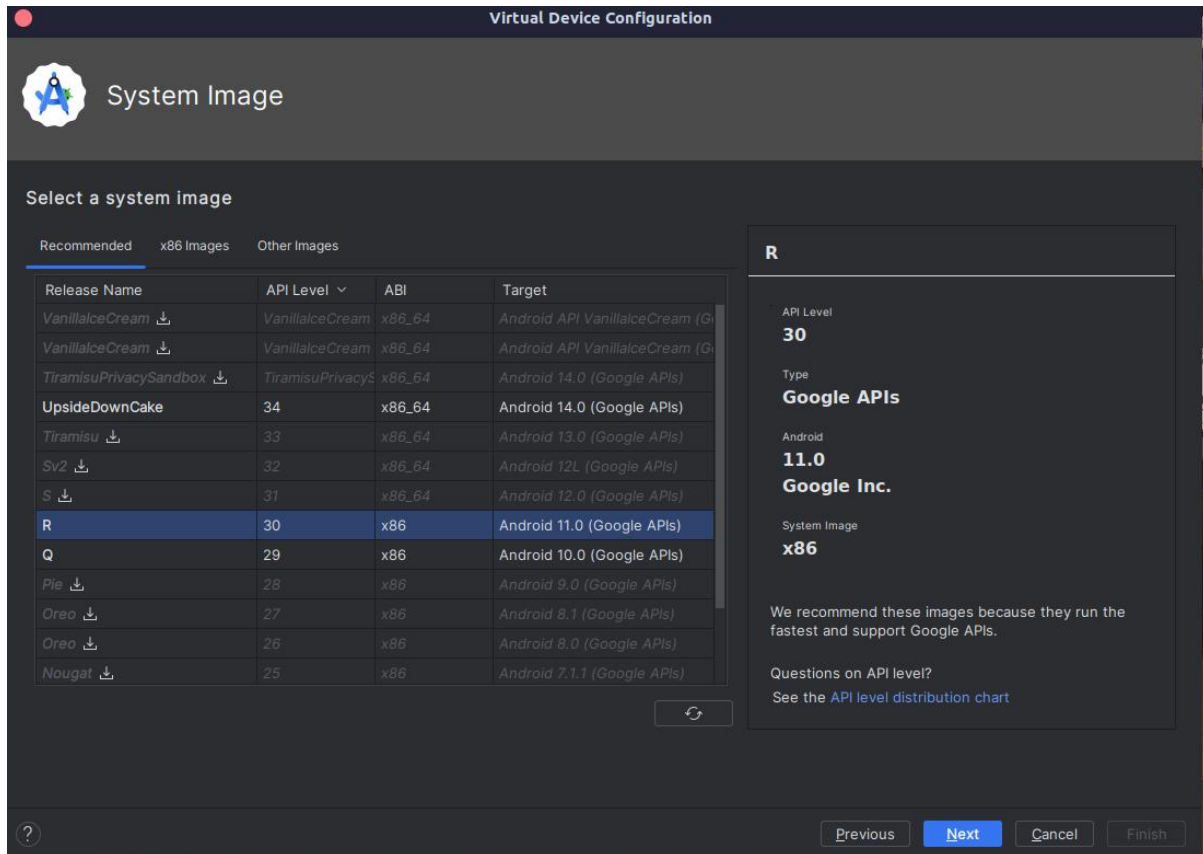
Bước 2: Nhấn vào biểu tượng dấu “+” góc trên phía bên trái cửa sổ để tạo một thiết bị ảo:



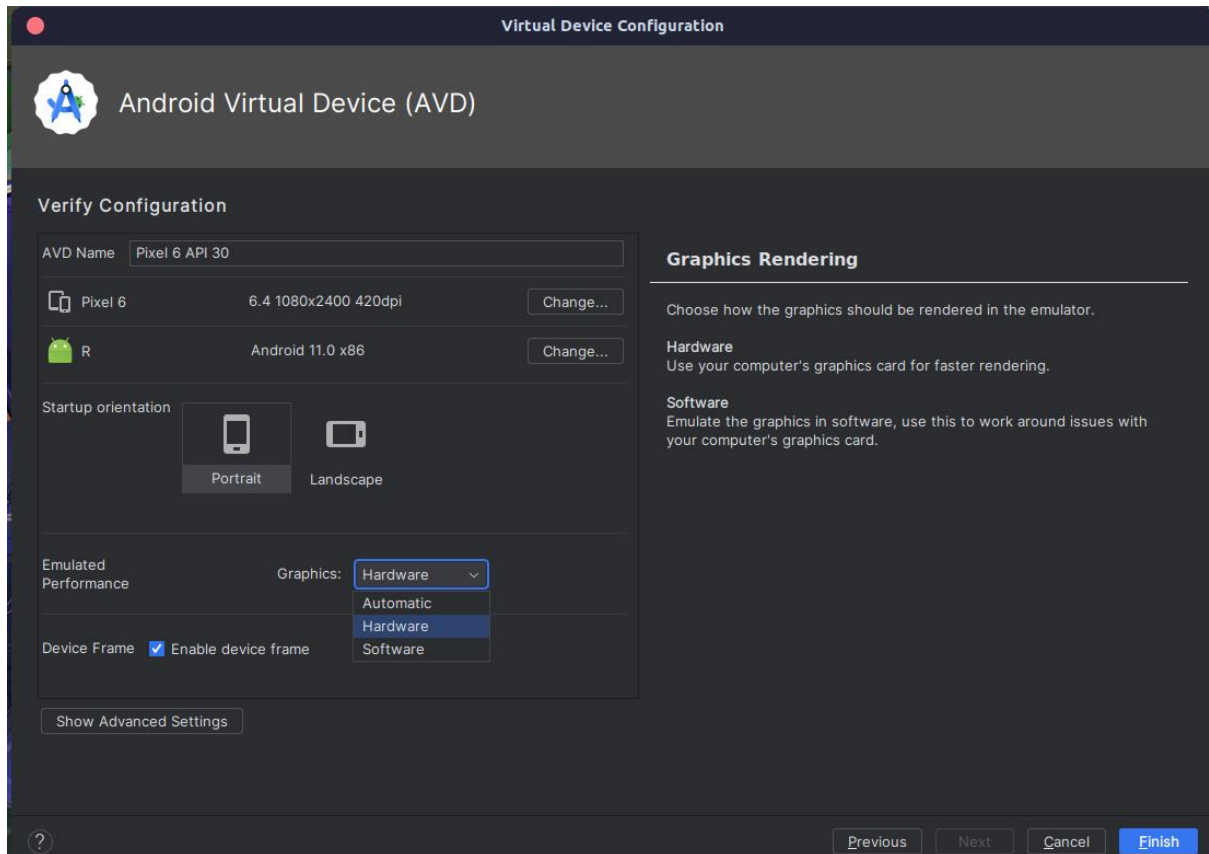
Bước 3: Cửa sổ Cấu hình thiết bị ảo xuất hiện (Virtual Device Configuration). Tại mục chọn loại thiết bị, có thể chọn bất cứ thiết bị nào mong muốn. Trong hướng dẫn này, tôi chọn thiết bị Pixel 6. Sau khi chọn được thiết bị, nhấn Next để tiếp tục



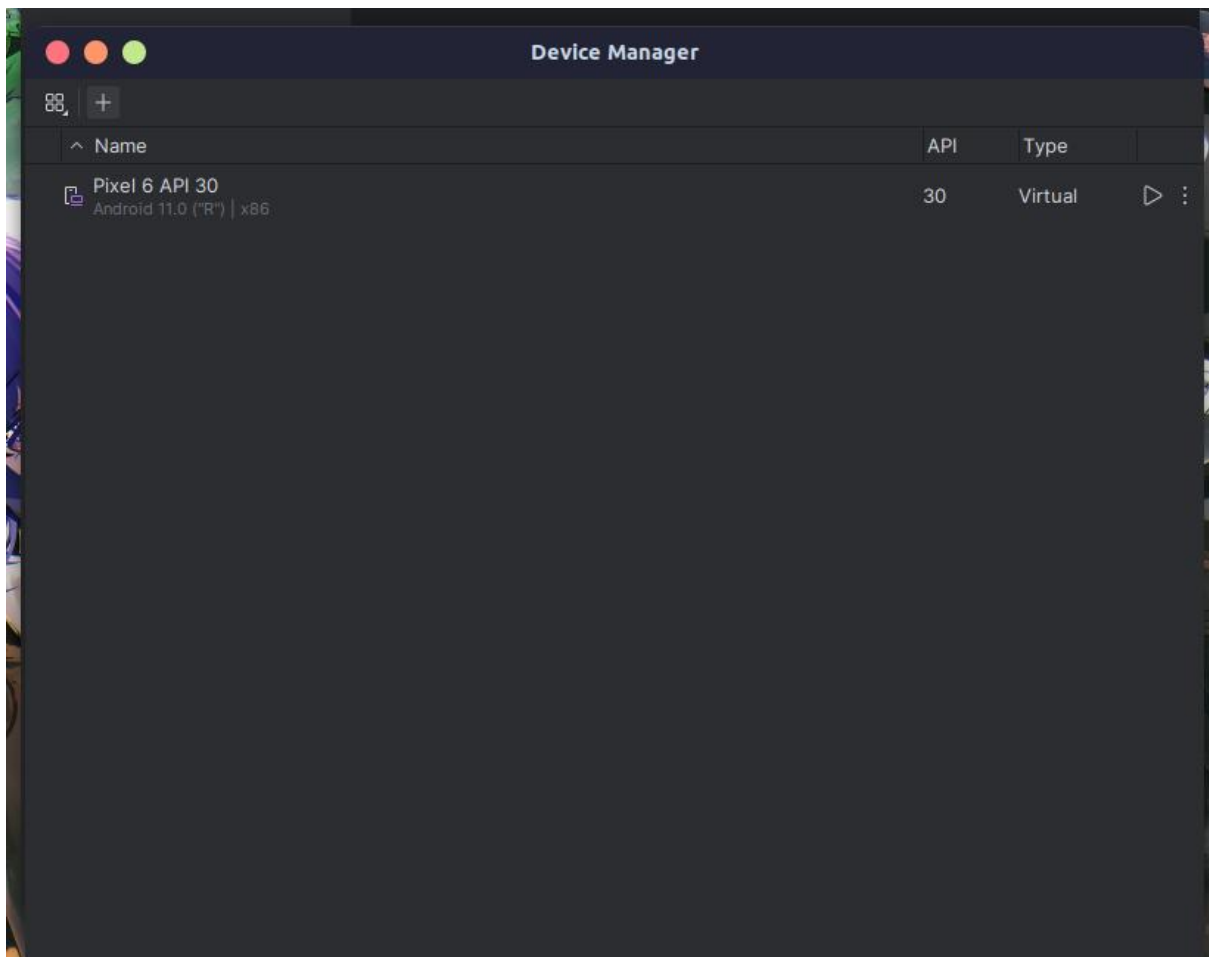
Bước 4: Tại phần System Image, chọn Loại hệ điều hành Android phù hợp, Trong hướng dẫn này, tôi chọn hệ điều hành Android 11. Lưu ý, Hệ điều hành quá mới có thể gây giật lag nếu máy không đủ mạnh. Nhấn Next để tiếp tục



Bước 5: Tại phần xác minh thông tin cấu hình, có thể đổi tên thiết bị theo mong muốn. Ở phần Emulated Performance → Graphic, nên chọn Hardware. Sau khi xác minh thông tin, nhấn Finish để trình quản lý máy ảo tạo máy ảo.



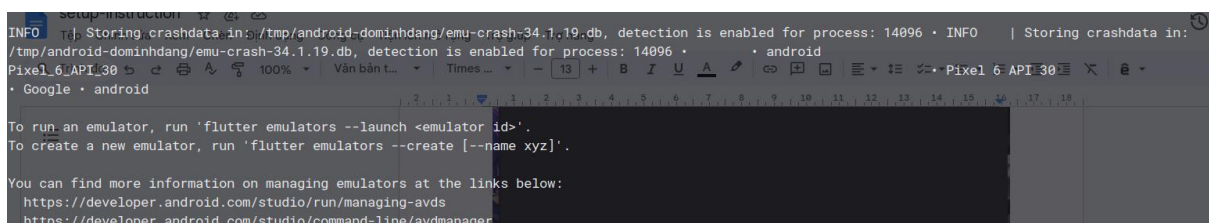
Sau khi tạo máy ảo, trên cửa sổ quản lý máy ảo sẽ xuất hiện máy ảo vừa tạo như hình dưới:



Để đảm bảo thiết bị ảo đã được kết nối, mở cửa sổ terminal hoặc command prompt, gõ lệnh

```
flutter emulators
```

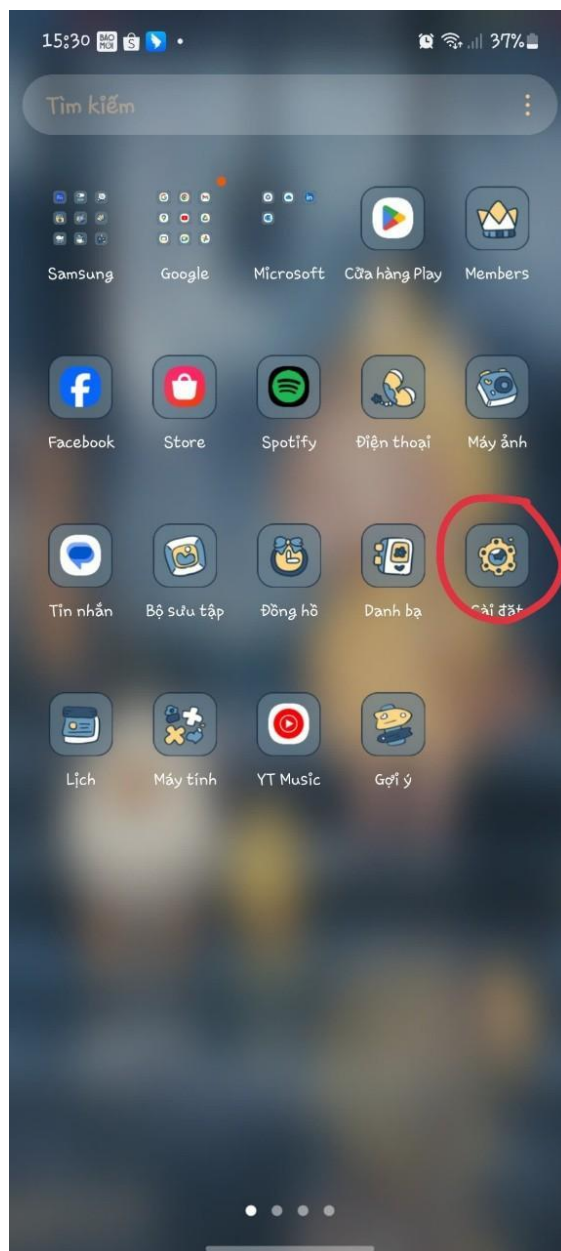
Nếu tên thiết bị xuất hiện trong danh sách, thiết bị đã được kết nối thành công với flutter



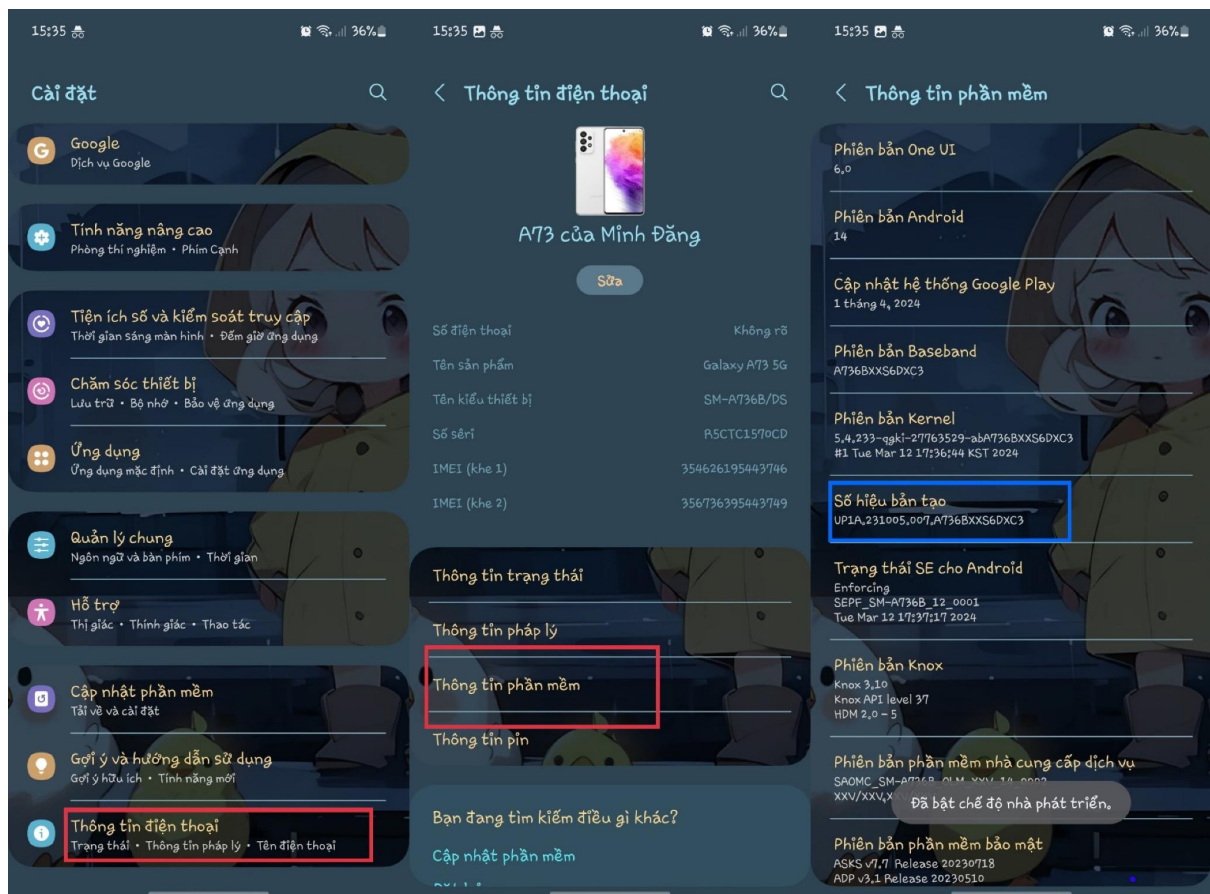
Đối với thiết bị vật lý

Bước 1: Chuẩn bị một điện thoại hệ điều hành Android và sợi cáp kết nối (Đa số các điện thoại khi mua sẽ được trang bị sẵn một sợi cáp để sạc và kết nối với máy tính)

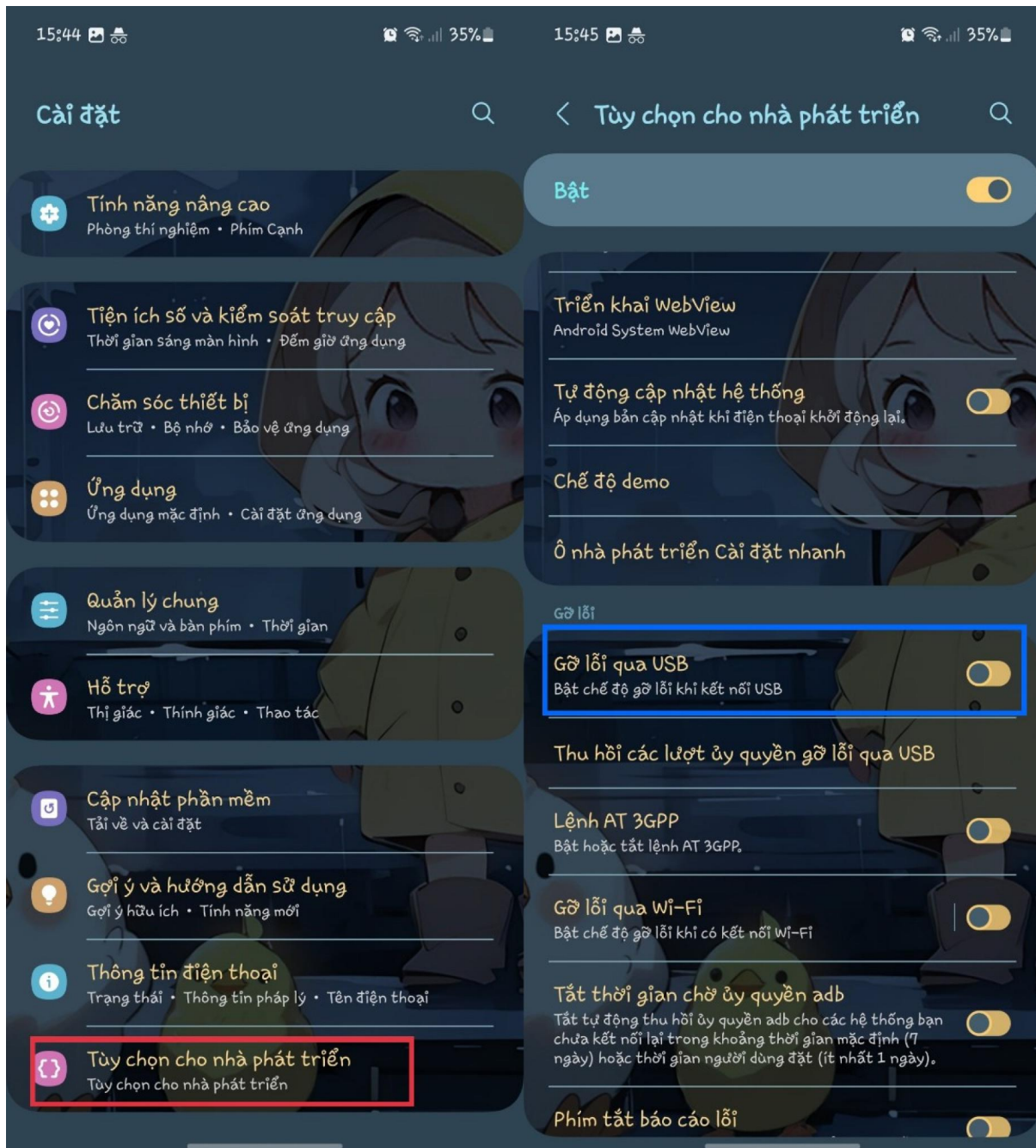
Bước 2: Trong điện thoại di động, Mở phần Cài đặt và lướt xuống dưới cùng. Nếu Đã có mục Tùy chọn cho nhà phát triển, Thực hiện bước 4



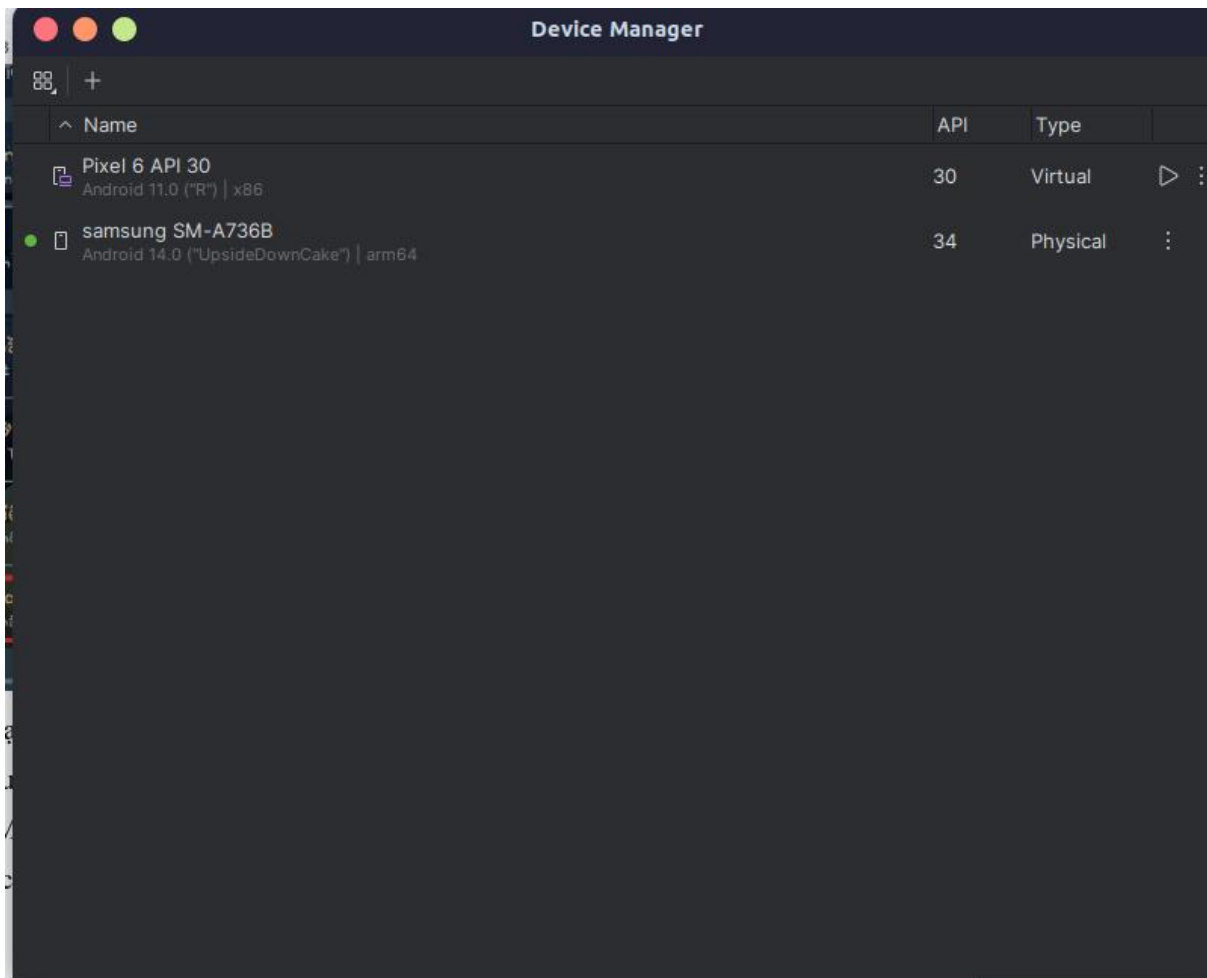
Bước 3: Vào mục Thông tin điện thoại > Thông tin phần mềm, nhấn liên tục vào mục Số hiệu bản tạo cho đến khi có thông báo Chế độ Nhà Phát triển đã được kích hoạt



Bước 4: Quay lại màn hình Cài Đặt, vào mục Tùy chọn cho nhà phát triển. Tìm mục Gỡ lỗi và bật chế độ Gỡ lỗi qua USB



Bước 5: Kết nối điện thoại di động với máy tính bằng dây cáp đã được chuẩn bị ở bước 1. Nếu điện thoại có yêu cầu quyền gỡ lỗi qua USB, nhấn Cho phép. Mở Android Studio > Virtual Device Manager. Điện thoại di động sẽ xuất hiện trong danh sách Trình quản lý thiết bị (Device Manager)



Để kiểm tra flutter đã nhận được thiết bị hay chưa, vào terminal, gõ lệnh sau và kiểm tra kết quả:

```
| flutter devices |
```

Dòng lệnh sẽ xuất ra danh sách các thiết bị đã được kết nối với Flutter. Lưu ý, Lệnh này không hoạt động đối với thiết bị giả lập không hoạt động

```
| flutter emulators |
```

b) Cấu hình domain của server api

Đối với ứng dụng web, Có thể sử dụng địa chỉ Localhost vì Cả 2 server đều hoạt động trên cùng một máy tính. Tuy nhiên, đối với ứng dụng di động yêu cầu phải chạy trên một thiết bị di động, mà thiết bị di động và máy tính là 2 môi trường phân biệt (kể cả thiết bị giả lập). Do đó, địa chỉ localhost sẽ không hoạt động trên các thiết bị di động. Giải pháp cho vấn đề này là làm cho server backend có thể truy cập được thông qua Internet.

Trong hướng dẫn này sẽ sử dụng ngrok để tạo một domain ảo cho domain của server backend, giúp cho địa chỉ server backend có thể được truy cập thông qua internet. Tuy nhiên, giải pháp này chỉ là tạm thời trong quá trình phát triển và kiểm thử. Nếu server backend đã được host lên một server và có thể truy cập được thông qua Internet, có thể bỏ qua bước này.

Lưu ý các hướng dẫn dưới đây chỉ hoạt động khi Ngrok đã được cài đặt và cấu hình đầy đủ. Chi tiết xem tài liệu của Ngrok sau đây: <https://ngrok.com/docs/getting-started/>

Bước 1: Thực hiện chạy dự án API (Cách thực hiện được giới thiệu ở phần 1)

Bước 2: Mở 1 terminal bất kỳ, gõ lệnh sau:

```
-----  
|  ngrok http https://localhost:7272  |  
-----
```

Kết quả khi chạy lệnh:

```
ngrok (Ctrl+C to quit)  
  
Try our new Traffic Inspector Dev Preview: https://ngrok.com/r/ti  
  
Session Status      online  
Account             dominhdangdalat@gmail.com (Plan: Free)  
Version             3.9.0  
Region              Asia Pacific (ap)  
Web Interface       http://127.0.0.1:4040  
Forwarding          https://9f24-104-28-205-70.ngrok-free.app -> https://localhost:7  
  
Connections         ttl      opn      rt1      rt5      p50      p90  
                   0        0        0.00     0.00     0.00     0.00
```

Đề ý tại dòng Forwarding, có một tên miền dạng .ngrok-free.app. Đây chính là domain có thể sử dụng được trên Internet. Lưu tên miền này lại/

Bước 4: Trong thư mục src/ của dự án, mở thư mục dự án Flutter local_education_app/. Trong dự án, tiếp tục mở file lib/constants/api_constant.dart. Trong file này, tìm thuộc tính tĩnh domain như trong hình:

```
class ApiEndpoint {
  static const String domain = '';

  // Authorization
  static const String authRoot = '$domain/api/users';
  static const String authLogin = '$authRoot/login';
  static const String authGetProfile = '$authRoot/getProfile';
  static const String authRegister = '$authRoot/register';
  static const String authRefreshToken = '$authRoot/refreshToken';

  // Course
```

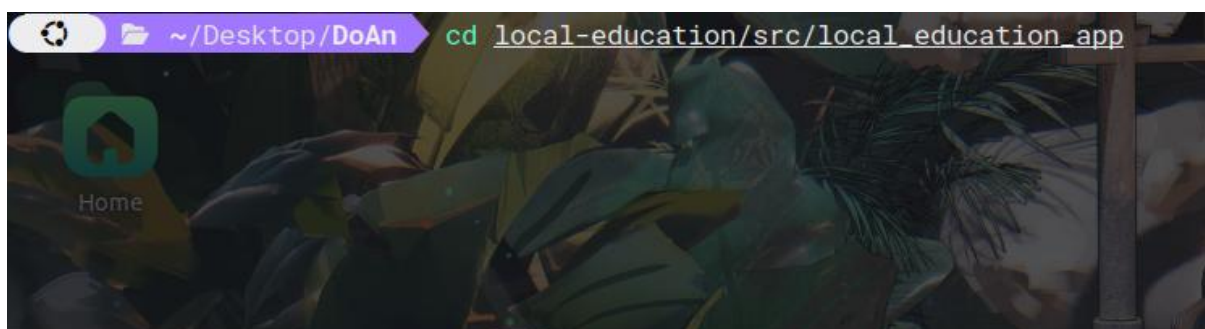
Bước 5: Dán tên miền dạng ngrok-free.app vừa lưu vào trong cặp dấu ngoặc đơn “”. Lưu lại thay đổi của file.

```
You, 1 second ago | 1 author (You)
class ApiEndpoint {
  static const String domain = 'https://9f24-104-28-205-70.ngrok-free.app';

  // Authorization
  static const String authRoot = '$domain/api/users';
  static const String authLogin = '$authRoot/login';
  static const String authGetProfile = '$authRoot/getProfile';
  static const String authRegister = '$authRoot/register';
  static const String authRefreshToken = '$authRoot/refreshToken';
```

c) Cài đặt và chạy ứng dụng di động

Bước 1: Sử dụng terminal, truy cập vào thư mục src/local_education_app/



Bước 2: Chạy lệnh sau để tải các package cần thiết trong dự án Flutter:

```
| flutter pub get
```

Bước 3: Kết nối điện thoại di động với máy tính (đối với thiết bị vật lý) hoặc khởi động thiết bị giả lập. Có thể sử dụng lệnh sau để khởi động:

Lưu ý, thay thế <device-id> bằng id của thiết bị giả lập cần khởi động.

```
| flutter emulators --launch <device-id> |
```

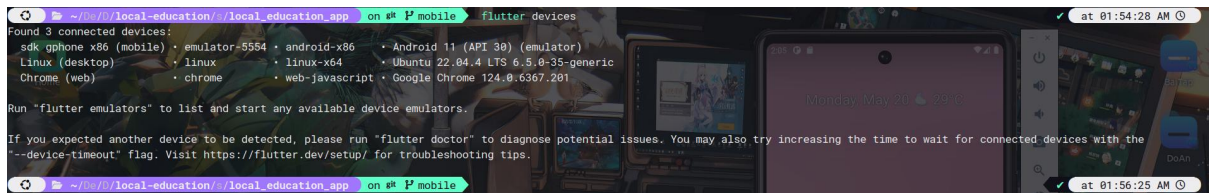
Sau khi chạy lệnh trên, thiết bị giả lập sẽ xuất hiện trên màn hình:



Bước 4: Trên Terminal, gõ lệnh sau để kiểm tra Thiết bị di động đã được kết nối hay chưa

```
| flutter devices
```

Kết quả với thiết bị di động đã được kết nối.



Có thể thấy thiết bị di động được đưa lên hàng đầu tiên, nghĩa là ứng dụng di động được ưu tiên triển khai.

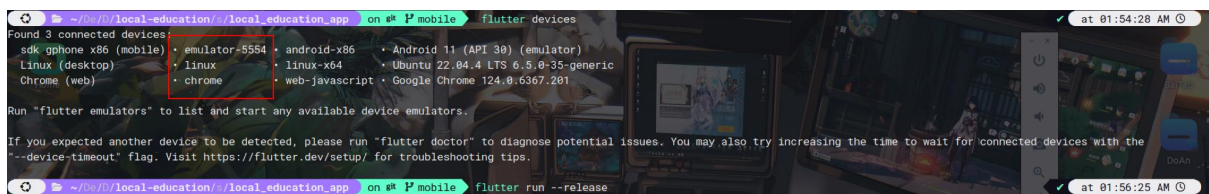
Bước 5: Gõ lệnh sau để tiến hành chạy dự án:

```
| flutter run
```

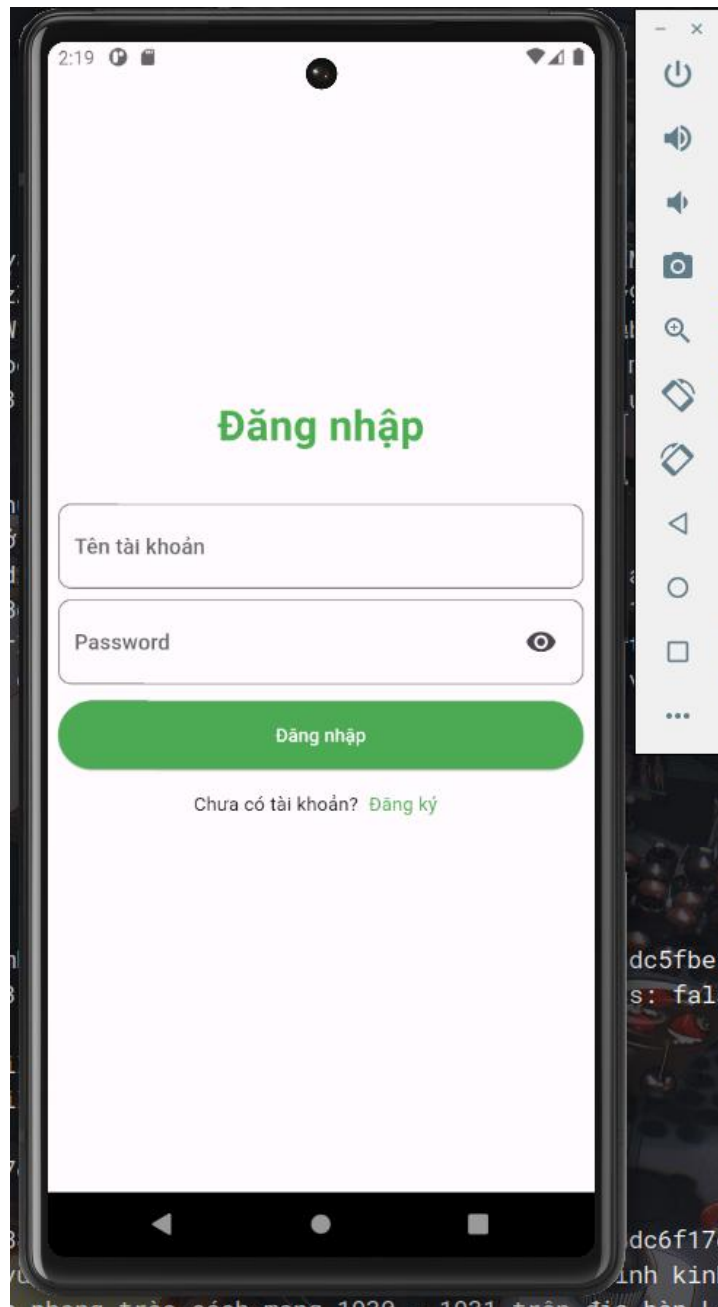
Do thiết bị di động được ưu tiên hàng đầu trong danh sách các thiết bị đang hoạt động, nên khi chạy dòng lệnh trên, Flutter sẽ tự động chạy ứng dụng trên thiết bị giả lập mà không cần phải khai báo trước. Tuy nhiên, nếu muốn chạy trên thiết bị di động khác, sử dụng lệnh sau:

```
| flutter run -d <device-id>
```

Thay <device-id> thành id của thiết bị cần chạy. Có thể xem id của thiết bị trong kết quả của lệnh *flutter devices* thông qua cột thứ 2:



Kết quả sau khi chạy lệnh:



– Hết –