BỘ GIÁO DỤC VÀ ĐÀO TẠO
ĐẠI HỌC KINH TẾ TP HỒ CHÍ MINH (UEH)
TRƯỜNG CÔNG NGHỆ VÀ THIẾT KẾ

ĐỒ ÁN MÔN HỌC

ĐỀ TÀI

ỨNG DỤNG CẤU TRÚC DICTIONARY ĐỂ TẠO RA MỘT BỘ TỪ ĐIỂN ANH - VIỆT

Học Phần: Cấu Trúc Dữ Liệu & Giải Thuật

Danh Sách Nhóm:
31231021575 - Đỗ Thái Gia Hy
31231020517 - Lê Hoàng
31231021637 - Phạm Anh Kiệt
31231023784 - Trần Huỳnh Huy Thông

Chuyên Ngành: KHOA HỌC DỮ LIỆU

Khóa: K49

Giảng Viên: TS. Đặng Ngọc Hoàng Thành

Tp. Hồ Chí Minh, Ngày 12 tháng 12 năm 2024

MỤC LỤC

CHƯƠNG 1. CẤU TRÚC DICTIONARY

1.1. Các Khái Niệm Liên Quan

Dictionary trong C# là một Collections lưu trữ dữ liệu dưới dạng cặp Key - Value. Key đại diện cho 1 khoá giống như chỉ số phần tử của mảng và Value chính là giá trị tương ứng của khoá đó. Ta sẽ sử dụng Key để truy cập đến Value tương ứng.

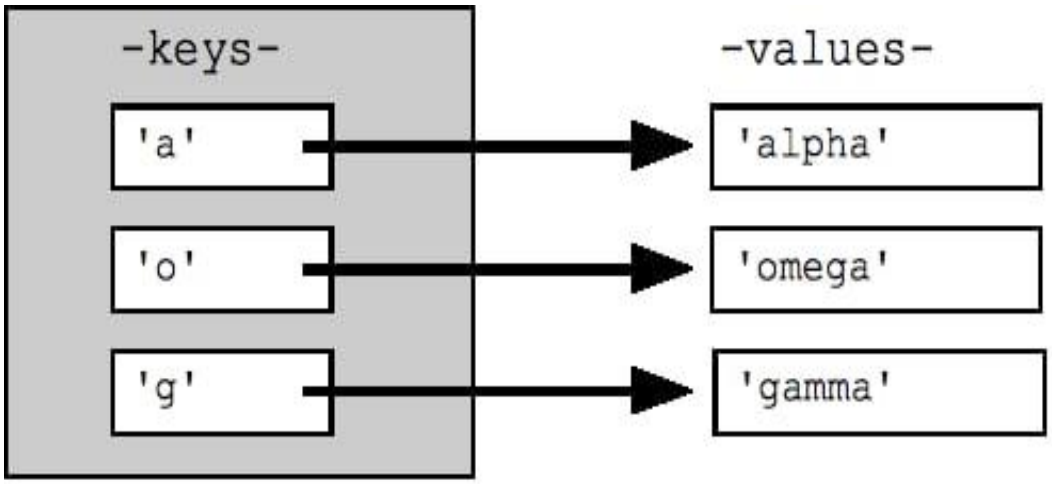
1.2. Cấu Trúc và Cài Đặt DICTIONARY

1.2.1 Cấu Trúc Từ Điển (Dictionary)

Định Nghĩa

-Từ điển là cấu trúc dữ liệu lưu trữ các cặp khóa (key) và giá trị (value).

-Mỗi khóa là duy nhất và ánh xạ tới một giá trị.



Cài Đặt Từ Điển

1. DictionaryBase:

- Sử dụng lớp cơ bản DictionaryBase để quản lý các cặp Key-Value trị.

- Người dùng có thể mở rộng lớp này để định nghĩa cách thêm, xóa và truy cập dữ liệu.

2. Generic Dictionary:

- Hỗ trợ khai báo rõ kiểu dữ liệu cho cả khóa và giá trị.

- Linh hoạt và an toàn khi làm việc với dữ liệu được định kiểu rõ ràng.

3. SortedList:

- Là một cấu trúc danh sách kết hợp, trong đó các cặp Key-Value được lưu trữ theo thứ tự tăng dần của khóa.

- Phù hợp khi cần dữ liệu luôn được sắp xếp.

Ví Dụ (Dạng Code)

DictionaryBase:

```
public class IPAddress : DictionaryBase
{
    0 references
    public void Add(string name, string ip)
    {
        base.InnerHashtable.Add(name, ip);
    }
    0 references
    public string Item(string name)
    {
        return base.InnerHashtable[name].ToString();
    }
    0 references
    public void Remove(string name)
    {
        base.InnerHashtable.Remove(name);
    }
}
```

Generic Dictionary:

```
Dictionary<string, string> myips = new Dictionary<string, string>();
myips.Add("Mike", "192.151.0.1");
```

SortedList:

```
SortedList<string, string> myips = new SortedList<string, string>();
myips.Add("Mike", "192.151.0.1");
```

1.2.2 Cấu Trúc Bảng Băm (Hash-Table)

Định Nghĩa

Bảng băm là một cấu trúc dữ liệu dùng để ánh xạ các khóa (Key) tới các giá trị (Value) thông qua hàm băm (Hash Function). Hàm băm chuyển đổi một khóa thành một chỉ số trong mảng. Bảng băm giúp tối ưu hóa việc tìm kiếm và quản lý dữ liệu.

Ưu điểm: Tìm kiếm nhanh, quản lý dữ liệu hiệu quả→ thích hợp với các ứng dụng cần truy cập dữ liệu nhanh chóng.

Cài Đặt Bảng Băm

1. Lớp BucketHash:

- Tạo cấu trúc bảng băm cơ bản với kích thước mảng cố định.

- Dữ liệu được phân tán vào các ô nhớ dựa trên hàm băm.

2. Lớp Hashtable:

- Là một lớp có sẵn trong C#, hỗ trợ quản lý các cặp khóa-giá trị với hiệu suất cao.

- Có khả năng xử lý xung đột khóa thông qua cơ chế liên kết.

Ví Dụ (Dạng Code)

BucketHash:

```
public class BucketHash
{
    private const int SIZE = 10;
    private ArrayList[] data;

    0 references
    public BucketHash()
    {
        data = new ArrayList[SIZE];
        for (int i = 0; i < SIZE; i++)
        {
            data[i] = new ArrayList();
        }
    }
```

Hashtable:

```
Hashtable infos = new Hashtable();
infos.Add("name", "David");
infos.Add("age", 45);
```

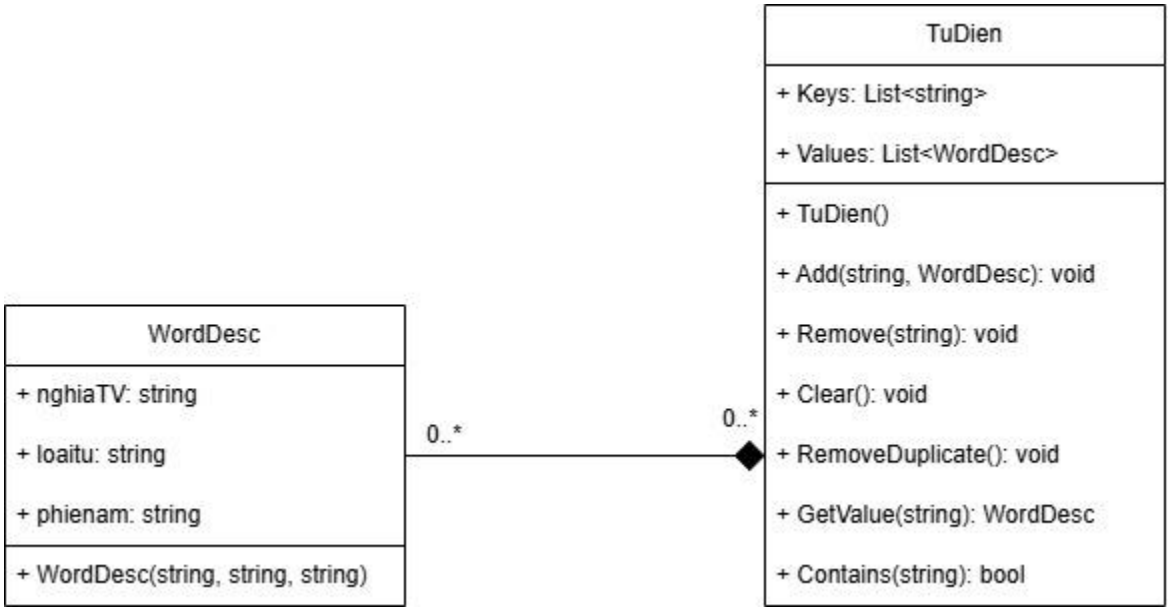
CHƯƠNG 2. PHÂN TÍCH VÀ THIẾT KẾ LỚP

2.1. Phân Tích Bài Toán Ứng dụng cấu trúc Dictionary để tạo ra một bộ từ điển Anh-Việt.

Để sử dụng cấu trúc Dictionary, đầu tiên ta cần xác định tập key và tập value sẽ chứa giá trị gì. Trong tình huống này, vì đang xây dựng bộ từ điển nên ta cho tập key là các từ tiếng Anh cần tra và tập value là thông tin của các từ đó bao gồm:

- Nghĩa tiếng Việt
- Loại từ (danh từ, động từ, tính từ, trạng từ, giới từ,...)
- Phiên âm (IPA)

2.2. Sơ Đồ Lớp



2.3. Cài Đặt Lớp

Để chứa các thông tin của từ, ta sẽ xây dựng một lớp có tên "WordDesc" gồm các thuộc tính muốn có. Lớp sẽ có một constructor (hàm tạo) để tạo ra đối tượng:

```
public class WordDesc
{
    public string nghiaTV;
    public string loaitu;
    public string phienam;

    1 reference
    public WordDesc(string nghiaTV, string loaitu, string phienam)
    {
        this.nghiaTV = nghiaTV;
        this.loaitu = loaitu;
        this.phienam = phienam;
    }
}
```

Tiếp theo, ta xây dựng lớp TuDien với 2 list, một cho Keys và một cho Values để tạo thành một cấu trúc từ điển:

- List Keys có kiểu dữ liệu là <string> để chứa các từ tiếng Anh cần tra.
- List Values có kiểu dữ liệu là lớp <WordDesc> mà ta đã tạo ở trên.

Ngoài ra lớp cũng có một hàm tạo để tạo từ điển. Hàm tạo hoạt động như sau:

- Đầu tiên, khi một đối tượng được tạo, 2 lists Keys và Values sẽ được khai báo.

- Để đọc file, string[] lines = resourceContent.Split(new[] { "\r\n", "\r", "\n" }, StringSplitOptions.None); được sử dụng để chia nội dung của file text dữ liệu thành các dòng riêng biệt. Ta bao gồm cả 3 ký tự "\r\n", "\r", "\n", để đảm bảo rằng nó có thể xử lý các file văn bản được tạo trên các hệ điều hành khác nhau.

- Các dòng trong file text được lưu vào một mảng "lines". Bằng vòng lặp foreach, từng dòng trong "lines" sẽ được lưu vào biến "line". Sau đó, sử dụng phương thức Split, ta tách "line" thành các chuỗi con bằng phân cách là dấu "," và lưu vào một mảng tên là "words".

- Vì file dữ liệu có định dạng là “chữ tiếng Anh,Nghĩa tiếng Việt,Loại từ,Phiên âm” , ta sẽ lưu được từng giá trị vào mảng “words” theo thứ tự tương ứng words[0], words[1], words[2],words[3].

- Khi đã có đủ giá trị, ta chỉ việc thêm chúng vào từ điển. words[0] sẽ được thêm vào list Keys là một khóa và words[1], words[2],words[3] sẽ được đưa vào hàm tạo để tạo nên một đối tượng của lớp WordDesc để đưa vào list Values. Nhờ vậy, với mỗi dòng ta sẽ có thêm một cặp Key - Value mới.

```
public class TuDien
{
    public List<string> Keys;
    public List<WordDesc> Values;


        5 references
        public TuDien()
        {
            Keys = new List<string>();
            Values = new List<WordDesc>();

            // Đọc nội dung từ tài nguyên (Resources)
            string resourceContent = Properties.Resources.anhviet; // Tên của file trong Resources
            string[] lines = resourceContent.Split(new[] { "\r\n", "\r", "\n" }, StringSplitOptions.None);

            foreach (var line in lines)
            {
                string[] words = line.Split(',');
                if (words.Length >= 4)
                {
                    Keys.Add(words[0]);
                    WordDesc word = new WordDesc(words[1], words[2], words[3]);
                    Values.Add(word);
                }
            }
        }
```

2.4. Các phương thức của lớp

a) Thêm một cặp Key - Value vào từ điển (Add)

```
public void Add(string key, WordDesc value)
{
    Keys.Add(key);
    Values.Add(value);
}
```

Mặc dù trong hàm tạo không sử dụng tới, phương thức Add vẫn cần thiết trong những trường hợp ta cần tự thêm một cặp giá trị cụ thể nào đó. Phương thức nhận 2 tham số là key kiểu string và value kiểu WordDesc, key sẽ được thêm vào list Keys và value sẽ được thêm vào list Values.

b) Xóa một cặp Key - Value trong từ điển (Remove)

```
public void Remove(string key)
{
    int index = Keys.IndexOf(key);
    if (index >= 0)
    {
        Keys.RemoveAt(index);
        Values.RemoveAt(index);
    }
}
```

Xóa một cặp Key - Value dựa vào tham số là Key. Phương thức tìm vị trí của Key trong list và thực hiện xóa trên list Key và Value tại vị trí đó.

```
public void Clear()
{
    Keys.Clear();
    Values.Clear();
}
```

c) Xóa hết phần tử trong lớp (Clear)

Khi gọi, các trị trong list Keys và list Values sẽ bị xóa hết.

d) Xóa các cặp Key - Value trùng nhau dựa vào Key (RemoveDuplicate)

```
public void RemoveDuplicate()
{
    int newCount = Keys.Count;
    for (int i = 0; i < newCount; i++)
    {
        for (int j = i + 1; j < newCount; j++)
        {
            if (Keys[i] == Keys[j])
            {
                this.Remove(Keys[j]);
                newCount = Keys.Count;
            }
        }
    }
}
```

Khi lấy một file text lớn là dữ liệu, khả năng hai hay nhiều Key bị trùng lặp là không thấp. Do đó, phương thức này sẽ làm sạch dữ liệu và đảm bảo mỗi khóa đều riêng biệt.

e) Truy cập một value thông qua key (GetValue)

```
public WordDesc GetValue(string key)
{
    int index = Keys.IndexOf(key);
    if (index == -1)
    {
        throw new KeyNotFoundException($"The key '{key}' was not found in the dictionary.");
    }
    return Values[index];
}
```

Phương thức trả về giá trị Value thông qua Key. Nếu không tìm được Key sẽ báo lỗi.

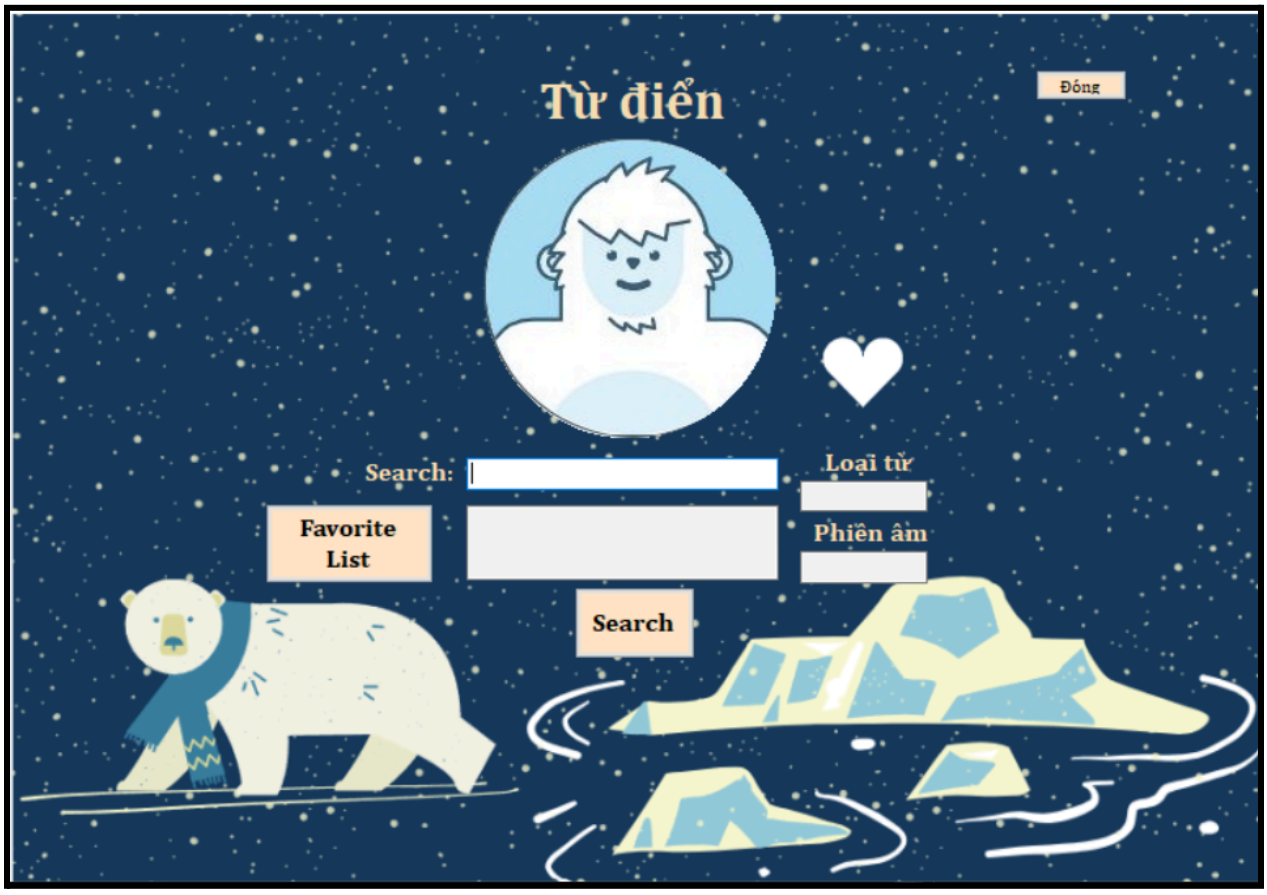
f) Kiểm tra sự tồn tại của một khóa (Contains)

```
public bool Contains(string key)
{
    if(Keys.Contains(key))
        return true;
    else return false;
}
```

Kiểm tra sao một khóa có tồn tại trong từ điển hay không. Nếu có trả true và nếu không trả false.

CHƯƠNG 3. THIẾT KẾ GIAO DIỆN

3.1. Giao Diện Menu Chính



3.2. Chi Tiết Chức Năng

3.2.1 Chức năng tìm kiếm và phiên dịch

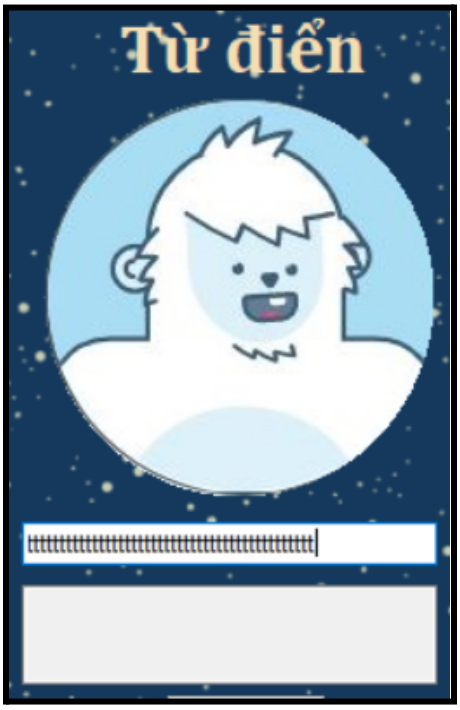
Chương trình có khả năng tìm kiếm và phiên dịch 1 từ Tiếng Anh sang Tiếng Việt, đồng thời cung cấp thêm thông tin về loại từ và phiên âm. Để chương trình hoạt động, người dùng sẽ nhập 1 từ bất kì vào thanh search ngay dưới logo sau đó nhấn vào nút "Search" bên dưới, hệ thống sẽ trả về các thông tin như đã nêu trên. Nếu từ đã nhập không có trong từ điển, hệ thống sẽ thông báo "Không tìm thấy từ vừa nhập"


Từ điển
Đóng
Search:
apple
Loại từ
danh từ
Favorite List
quả táo
Phiên âm
/ˈæp.əl/
Search


Từ điển
Đóng
Không tìm thấy từ: abcdefg
OK
Search:
abcdefg
Favorite List
Phiên âm
Search

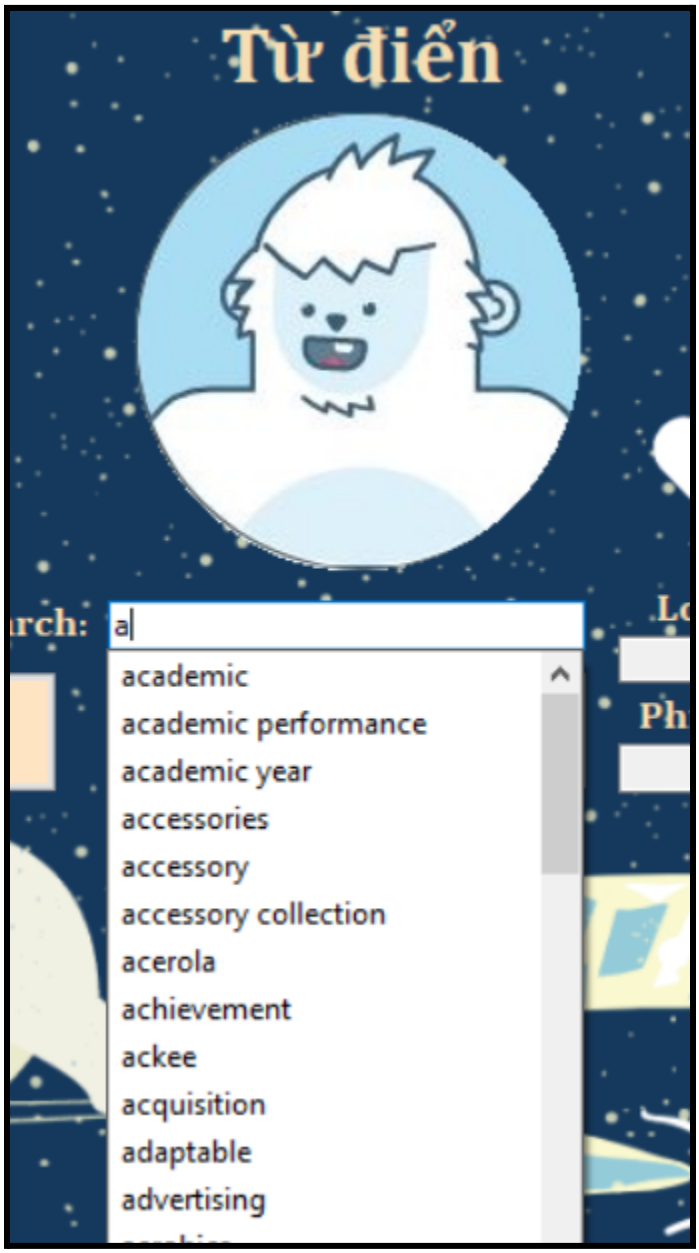
3.2.2 Chức năng hoạt ảnh động của logo

Logo của chương trình có thể chuyển động dõi theo theo từ mà người dùng nhập vào và sẽ chuyển từ mở mắt sang che mắt nếu người dùng search 1 từ không có trong từ điển, để logo trở về trạng thái mở mắt chỉ cần nhấp chuột vào thanh search.



3.2.3 Chức năng gợi ý từ nhập vào

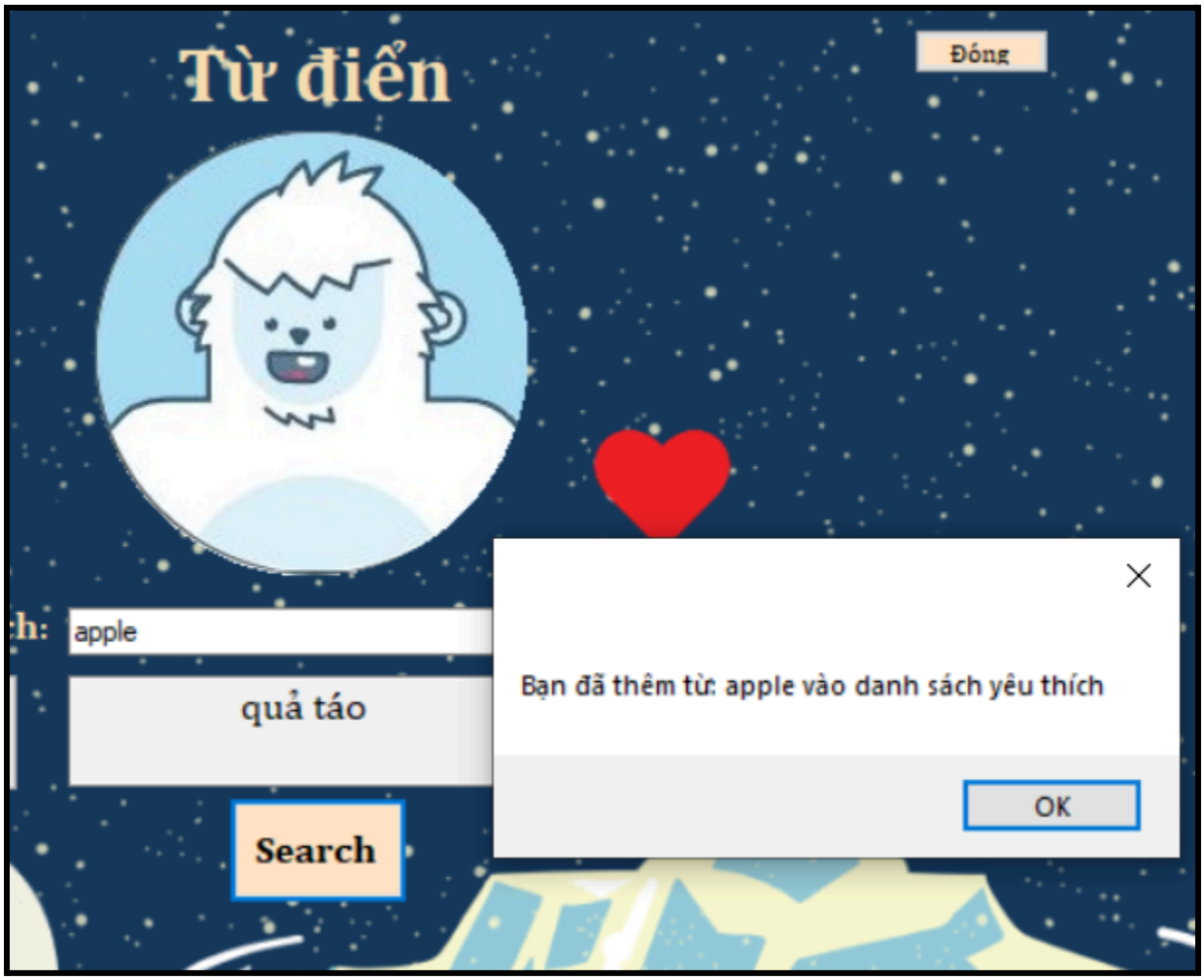
Khi người dùng nhập từ vào thanh search hệ thống sẽ hiện ra 1 dòng gợi ý các từ gần giống với từ người dùng nhập vào, nếu nhấp chuột chọn từ trong dòng gợi ý thì thanh search sẽ tự động điền đầy đủ

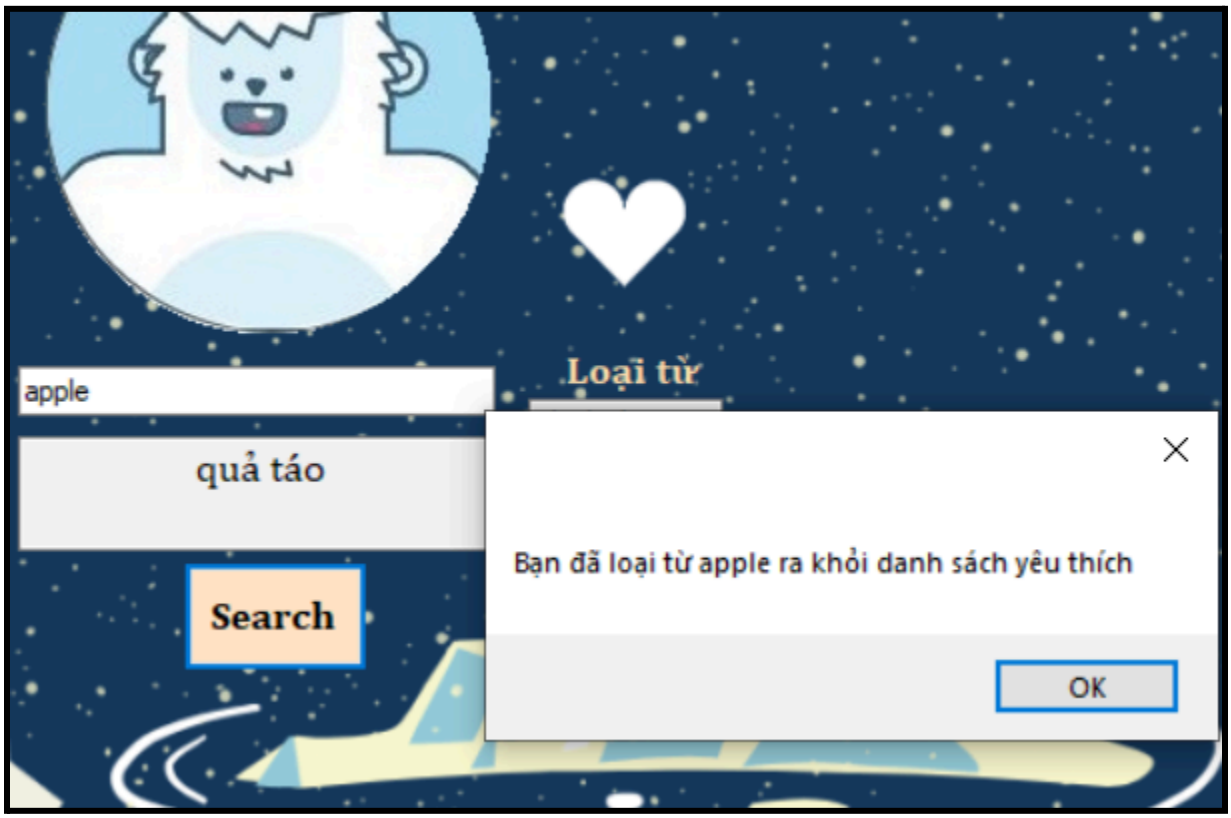


3.2.4 Chức năng thêm vào danh sách yêu thích

Chương trình sẽ có chức năng thêm các từ vào danh sách yêu thích nhằm tối đa hiệu quả học tập, từ sẽ được thêm vào danh sách yêu thích với điều kiện từ đó có trong từ điển và từ đó đã được search, để thêm từ vào danh sách yêu thích người dùng sẽ nhấn vào icon trái tim màu trắng lập tức icon sẽ chuyển thành trái tim màu đỏ và thêm từ vào danh sách

yêu thích. Nếu người dùng search lại các từ đã có trong danh sách yêu thích, icon trái tim sẽ vẫn ở màu đỏ và nếu nhấn vào icon trái tim màu đỏ, từ sẽ bị loại ra khỏi danh sách yêu thích và chuyển lại thành icon trái tim màu trắng(tương tự chức năng tim của ứng dụng Tiktok)





3.2.5 Chức năng danh sách yêu thích (Favorite List)

Khi bấm vào nút danh sách yêu thích, một cửa sổ khác sẽ hiện lên và nó như một từ điển thứ hai bao gồm các từ mà ta đã tim. Trong cửa sổ này, người dùng có thể click từng chữ trong listbox và xem các thông tin của từ như thường.

Ngoài ra, người dùng còn có thể search từ mình muốn tìm trong textbox ở trên trong trường hợp có quá nhiều từ trong danh sách.

Nếu có từ nào muốn xóa, người dùng chỉ cần chọn từ đó và nhấn nút “Xóa khỏi Favorite”, danh sách sẽ động cập nhật và từ đó sẽ biến mất ngay.

CHƯƠNG 4. THẢO LUẬN & ĐÁNH GIÁ

4.1. Các Kết Quả Nhận Được

Sau khi hoàn thành việc thiết kế và cài đặt, đạt được các kết quả sau:

- **Xây dựng cơ sở dữ liệu từ điển:** Từ điển bao gồm tập các từ tiếng Anh (Key) và các thông tin nghĩa tiếng Việt, loại từ, phiên âm.
- **Thiết kế giao diện người dùng:**
  - Giao diện đơn giản, trực quan.
  - Menu chính cho phép truy cập nhanh tới các chức năng: tìm kiếm, danh sách favorite.
  - Giao diện có tính năng gợi ý/ autocomplete khi nhập.
- **Chức năng danh sách yêu thích:**
  - Lưu trữ các từ mà người dùng đánh dấu.
  - Hỗ trợ tìm kiếm nhanh trong danh sách yêu thích.
  - Dễ dàng xóa từ khỏi danh sách yêu thích.
- **Hiệu năng, tính linh hoạt:**
  - Các thao tác tìm kiếm, sắp xếp, thêm/xóa từ trong favorite được tự động hóa và nhanh chóng.
  - Xử lý dữ liệu linh hoạt từ file text và giao diện.
  - Loại bỏ các mục trùng lặp trong từ điển.

4.2. Một Số Tồn Tại

- **Tính năng hạn chế:**
  - Chương trình chưa hỗ trợ phát âm từ hoặc các tính năng tra ngược (Việt - Anh).
  - Chưa tích hợp cơ sở dữ liệu lưu trữ bên ngoài (như SQLite hoặc API).
- **Dữ liệu từ điển:**
  - File text được tạo thủ công, là tập hợp các từ của nhiều chủ đề khác nhau nhưng vẫn còn nhiều thiếu sót.
  - Chỉ hiển thị 1 nghĩa đối với từ nhiều nghĩa.
- **Lỗi tính năng**

  -Có thể xuất hiện 1 số lỗi do hành vi bất thường của người dung

4.3. Hướng Phát Triển

- Mở rộng tính năng:
  - Tích hợp API như Oxford Dictionary API hoặc Google Translate API để lấy thêm nghĩa từ và phát âm.

- Phát triển chức năng tra ngược từ Việt sang Anh.
- Hỗ trợ giao diện mobile (Android/iOS).

- Quản lý dữ liệu:
  - Chuyển sang sử dụng cơ sở dữ liệu như SQLite hay MySQL thay vì file text.
  - Xây dựng các tính năng backup và khôi phục dữ liệu.
- Cải thiện giao diện:
  - Thêm hình ảnh minh họa giao diện trong chương trình.
  - Phát triển giao diện theo phong cách hiện đại (Material Design).

CHƯƠNG 5. PHỤ LỤC

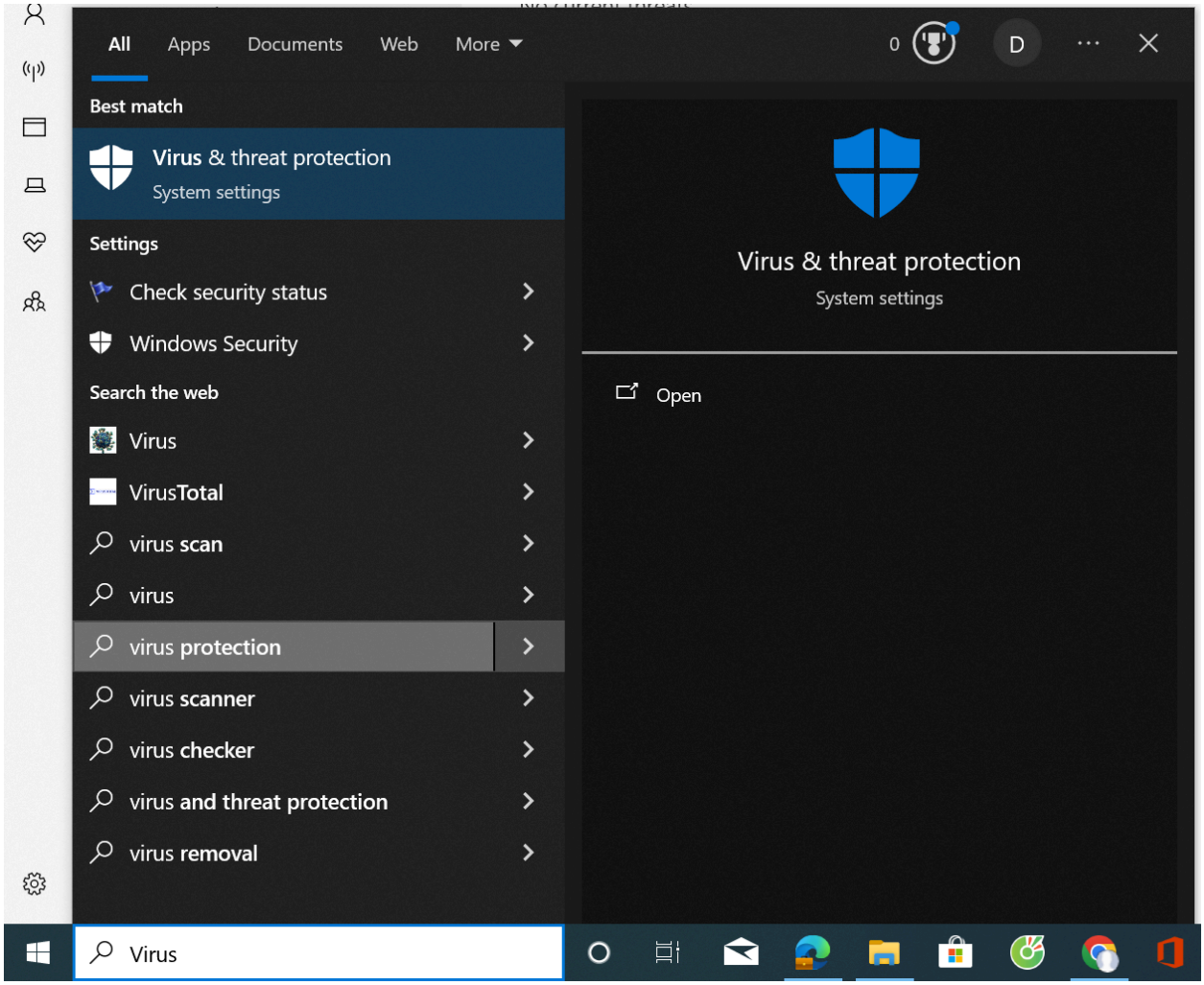
5.1. Mã nguồn Github

https://github.com/hydo1/DSA-Project-ENtoVN-Dictionary/tree/main

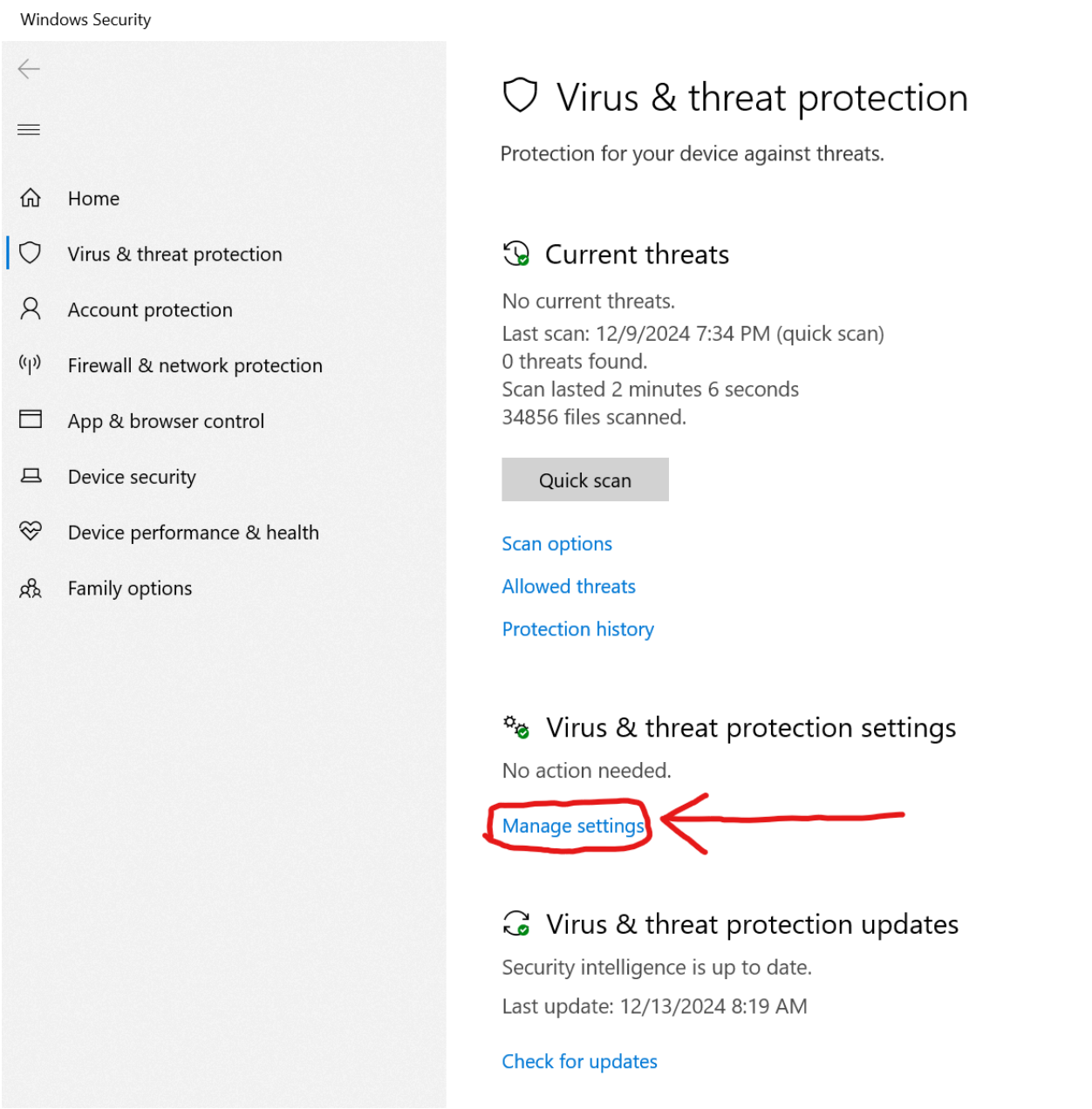
5.2. Hướng dẫn cài đặt

Bước 0: Tắt diệt virus

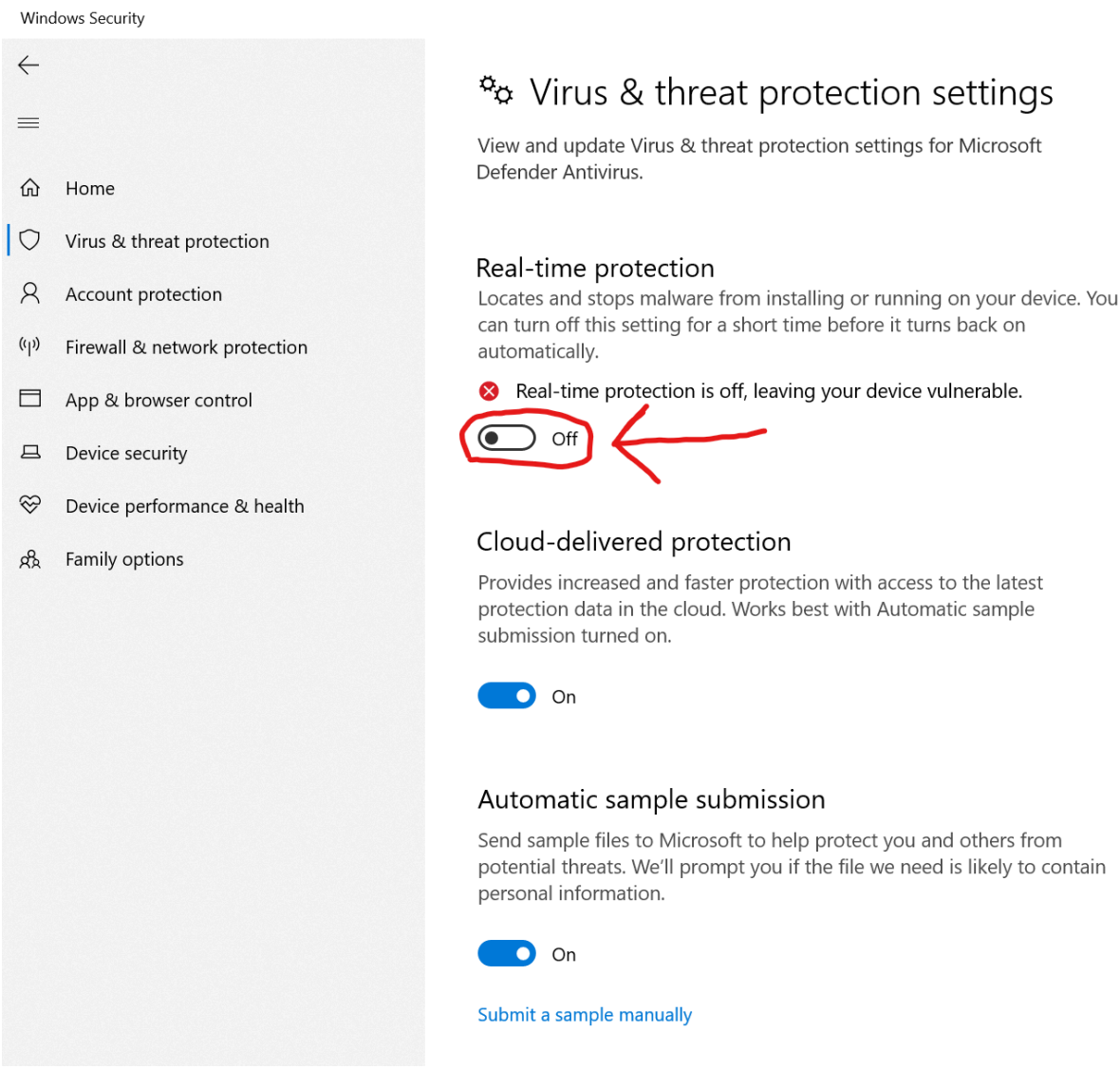
Khi tải file trên GitHub, khả năng cao file sẽ bị chặn vì virus. Để tải, ta phải tắt phần mềm diệt virus của Windows. **Nhóm cam kết các file đều không độc hại.**



Search và ấn vào Virus & threat protection.



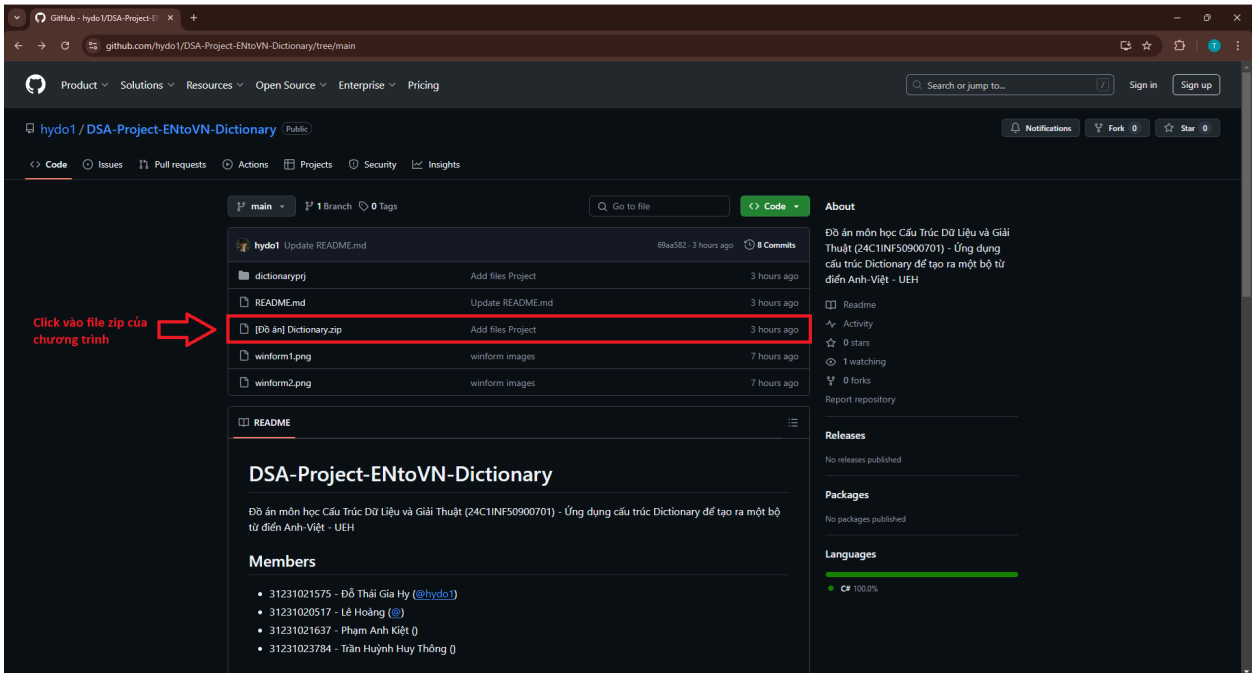
Dưới Virus & threat protection settings, bấm vào manage settings.

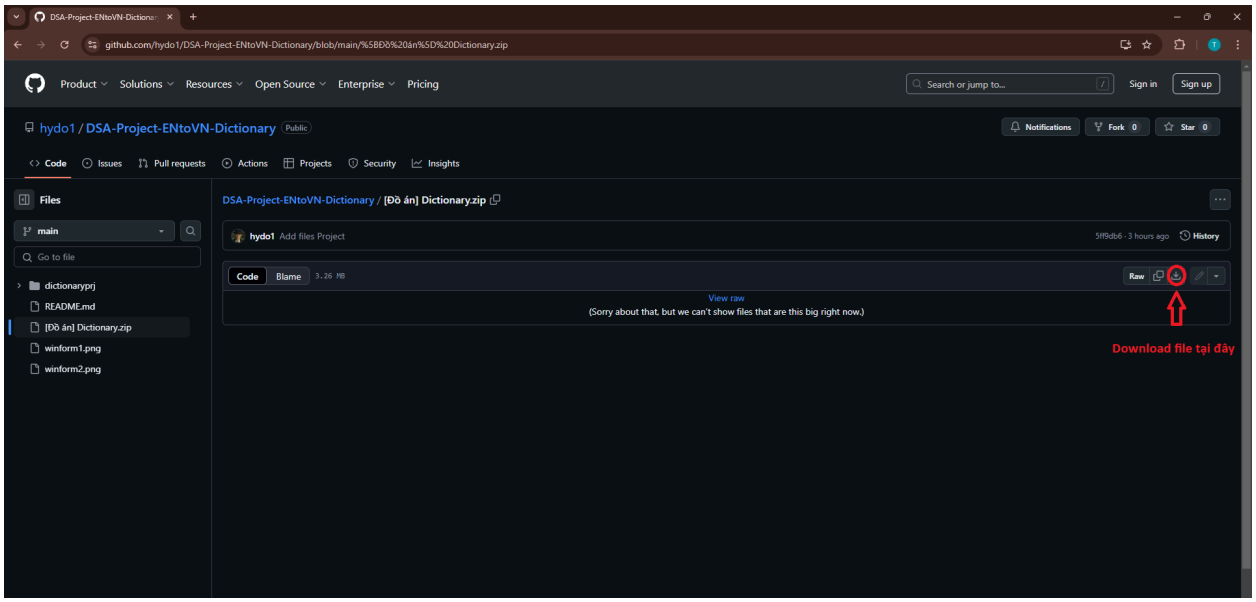


Dưới Real-time protection, đảm bảo nó được tắt.

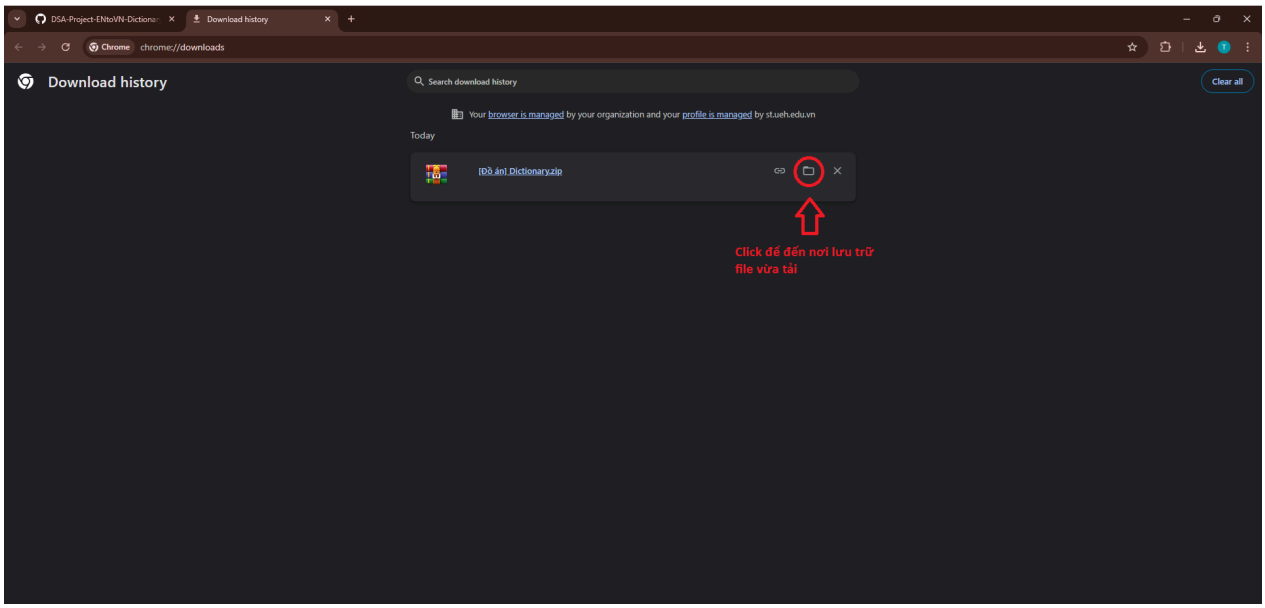
Bước 1: Truy cập đường link GitHub để xem dự án được nhóm tải lên.

Bước 2: Truy cập thư mục của chương trình và tải theo hướng dẫn.



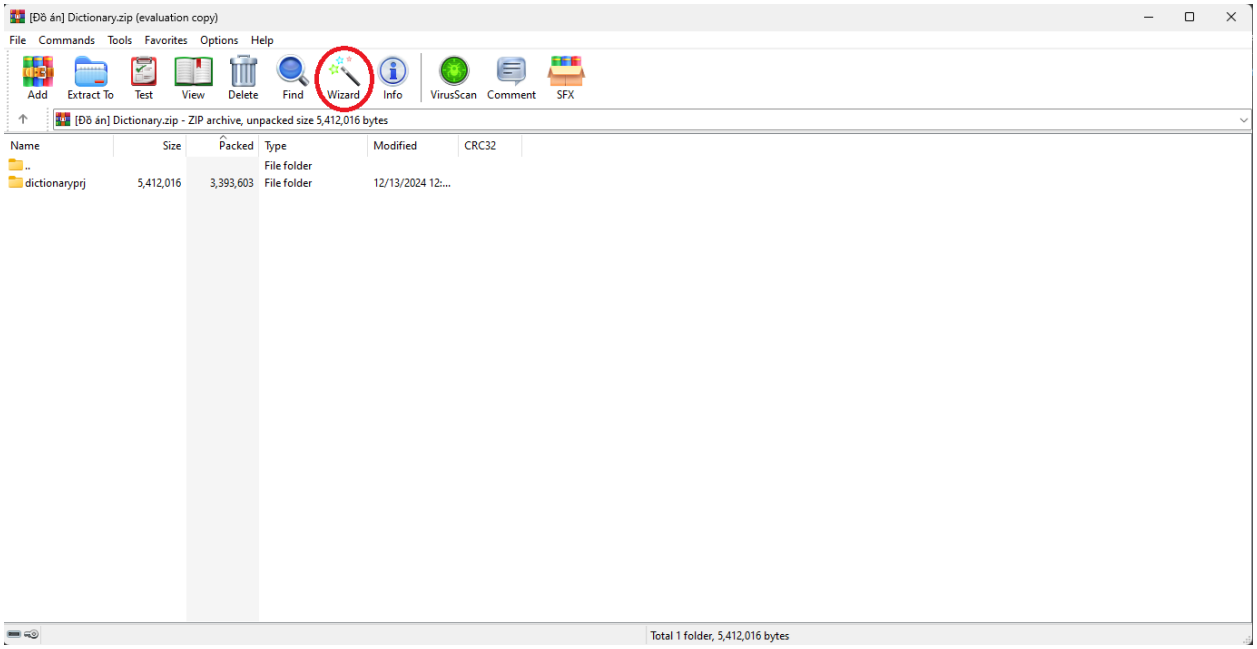


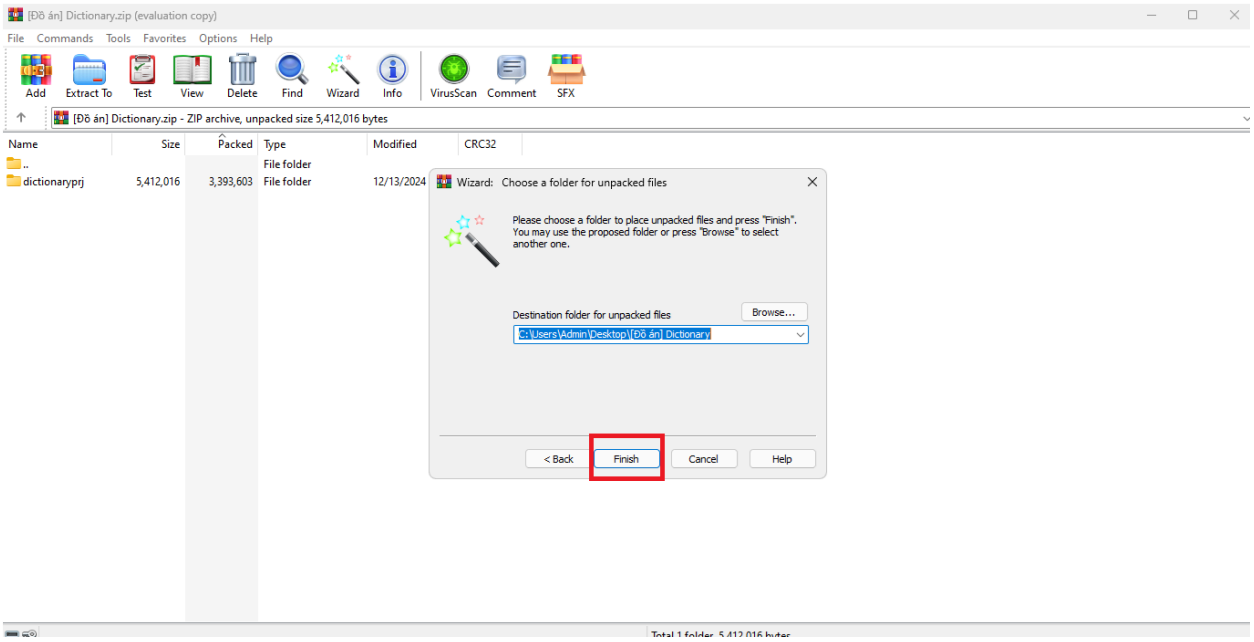
Để dễ dàng biết file vừa tải nằm ở đâu, ta truy cập vào lịch sử tải xuống thông qua shortcut Ctrl + J và ấn vào biểu tượng thư mục như trong hình (Show in folder).



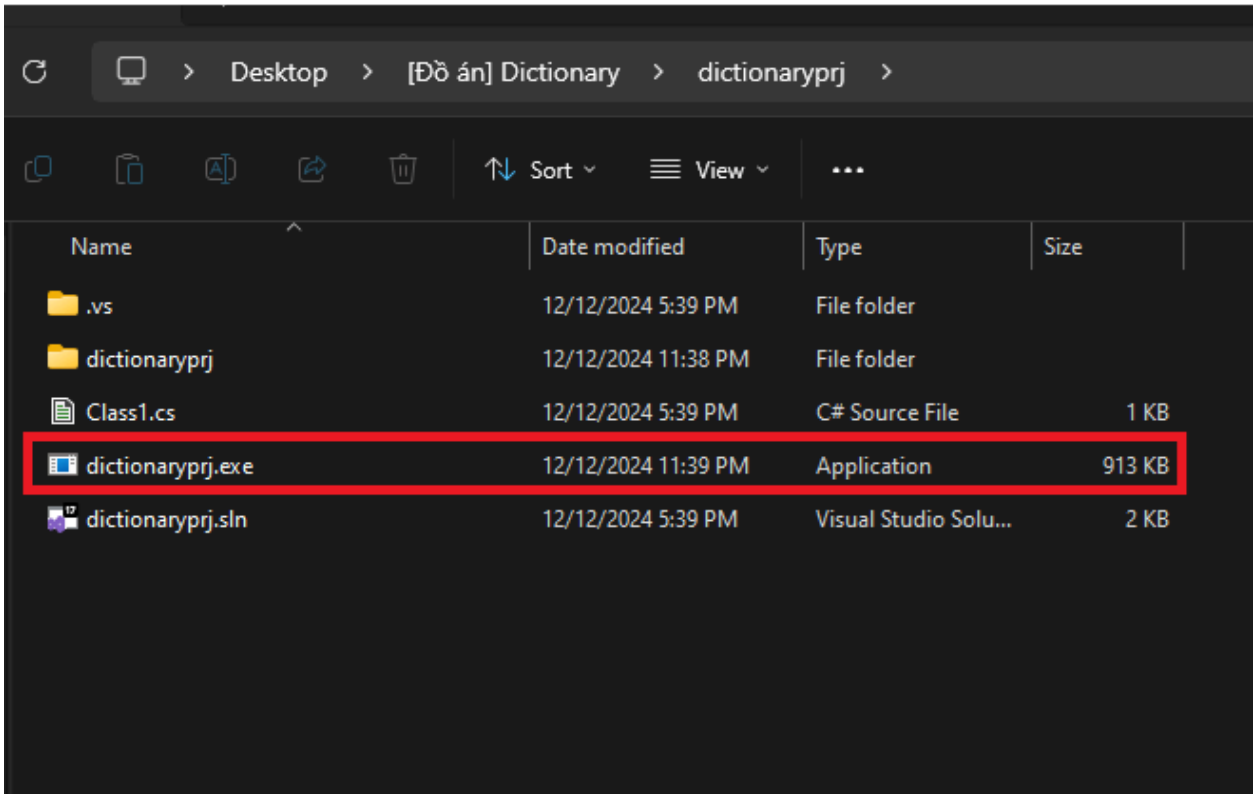
Bước 3: Cài đặt chương trình. (Đối với người dùng WinRAR)

Mở file zip vừa tải về. Ấn chọn "Wizard" để giải nén file ra desktop. Những lần truy cập sau chỉ cần mở folder đã giải nén nằm ở desktop.





Sau khi giải nén xong sẽ có window của file giải nén pop-up, mở folder "dictionaryprj" và chạy file dictionaryprj.exe.



5.3. Phân công

Thành Viên	Nhiệm Vụ
Phạm Anh Kiệt	Giới thiệu khái niệm cấu trúc Dictionary.
Đỗ Thái Gia Hy	Phân tích và thiết kế lớp. Thiết kế FavoriteList (form2) và chức năng trong đó.
Lê Hoàng	Thiết kế giao diện, các chức năng chính trong winform. (form1)
Trần Huỳnh Huy Thông	Tạo file dữ liệu cho từ điển. Thảo luận, đánh giá về các kết quả nhận được và trình bày hướng phát triển.

5.4. Tài liệu tham khảo

1. Michael McMillan, Data Structures and Algorithms Using C#

-HẾT-