TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

PROJECT 3

Ứng dụng cho PC: từ điển Anh - Việt

NGÔ MINH QUANG
quang.nm173326@sis.hust.edu.vn

Ngành Công nghệ thông tin
Chuyên ngành Công nghệ thông tin

Giảng viên hướng dẫn: THS. Nguyễn Mạnh Tuấn

Bộ môn: Công nghệ phần mềm
Viện: Công nghệ thông tin – Truyền thông

HÀ NỘI, 11/2020

Lời cam kết

Họ và tên sinh viên: Ngô Minh Quang

Điện thoại liên lạc: 0989598543 Email:quang.nm173326@sis.hust.edu.vn

Lớp: CNTT.09 Hệ đào tạo: Cử nhân Kỹ thuật

Tôi Ngô Minh Quang cam kết PROJECT 3 là công trình nghiên cứu của bản thân tôi dưới sự hướng dẫn của THS. Nguyễn Mạnh Tuấn. Các kết quả nêu trong PROJECT 3 là trung thực, là thành quả của riêng tôi, không sao chép theo bất kỳ công trình nào khác. Tất cả những tham khảo trong PROJECT 3 bao gồm hình ảnh, bảng biểu, số liệu, và các câu từ trích dẫn đều được ghi rõ ràng và đầy đủ nguồn gốc trong danh mục tài liệu tham khảo. Tôi xin hoàn toàn chịu trách nhiệm với dù chỉ một sao chép vi phạm quy chế của nhà trường.

Hà Nội, ngày 30 tháng 11 năm 2020

Tác giả PROJECT 3

Quang

Họ và tên sinh viên

Ngô Minh Quang

Lời cảm ơn

Em xin chân thành cảm ơn thầy Nguyễn Mạnh Tuấn, người đã tận tình hướng dẫn em trong suốt thời gian thực hiện đề tài này.

Trong thời gian thực hiện đề tài, bằng những kiến thức đã được nhà trường trang bị và tự tìm hiểu qua sách vở, mạng internet, sự hướng dẫn nhiệt tình của giảng viên hướng dẫn … em đã vận dụng triệt để hoàn thành đề tài của mình một cách tốt nhất. Nhưng kinh nghiệm trong lĩnh vực thiết kế, lập trình và trình độ còn nhiều hạn chế nên không thể tránh khỏi thiếu sót. Em rất mong nhận thêm được những ý kiến đóng góp để em có thêm kinh nghiệm, hoàn thiện và phát triển đề tài đồ án này trong tương lai.

Một lần nữa em xin chân thành cảm ơn !

Tóm tắt

Đồ án với đề tài "Xây dựng ứng dụng từ điển Anh – Việt cho PC" sẽ đưa ra những cái nhìn tổng quát về nền tảng Windows Forms, đây là một nền tảng ứng dụng phổ biến nhất cho PC. Dựa trên nền tảng này để xây dựng một ứng dụng cho PC, cụ thể là ứng dụng từ điển. Qua đó, đồ án đi sâu vào phân tích cách thức để làm một ứng dụng từ điển, từ việc thiết kế cơ sở dữ liệu, lập trình, đến cách cài đặt và hướng phát triển của ứng dụng. Nguồn dữ liệu sử dụng trong đồ án được thu thập từ dữ liệu từ điển được chia sẻ công khai trên google, đã được định dạng lại để phù hợp với đồ án. Với dung lượng nhỏ (khoảng 44 MB) ứng dụng này có thể cài đặt lên tất cả các PC. Với ứng dụng từ điển này cho phép: tra từ Anh – Việt; nghe phát âm từ; phân tích nghĩa của một từ với nguồn dữ liệu phong phú gồm 109.000 từ; lưu những từ mà người dung cần lưu vào Danh mục yêu thích để tiện sử dụng khi cần, dễ dàng cho việc học từ vựng, lưu xong nếu người dùng cảm thấy không cần thiết có thể xóa bỏ và luyện tập trắc nghiệm với các từ có trong mục yêu thích.

Mục lục

Danh mục hình vẽ

Danh mục bảng

Danh mục các từ viết tắt

PK	Primary Key
FK	Foreign Key
PC	Personal Computer
GUI	Graphical User Interface
IDE	Integrated Development Environment
CNTT	Công nghệ thông tin
PROJECT 3	Đồ án 3 (Project 3)
API	Application Programming Interface
WinForm	Windows Form

Danh mục thuật ngữ

Windows Form	Là một công nghệ của Microsoft, công cụ dùng để tạo các ứng dụng Windows
Personal Computer	Máy tính cá nhân
Integrated Development Environment	Là môi trường phát triển tích hợp
Primary Key	Là một hoặc nhiều column co tính chất đặc biệt đó là dùng để phân biệt sự khác nhau giữa các dòng dữ liệu
Foreign Key	Là những column đặc biệc dùng để thể hiện mối liên kết giữa hai bảng
Application Programming Interface	Là một giao diện mà một hệ thống máy tính hay ứng dụng cung cấp để cho phép các yêu cầu dịch vụ có thể được tạo ra từ các chương trình máy tính khác, và/hoặc cho phép dữ liệu có thể được trao đổi qua lại giữa chúng.
Button	Là một thành phần tương tác cho phép người dùng giao tiếp với một ứng dụng
HashTable	Lưu trữ bất kì kiểu dữ liệu nào dưới dạng cặp Key – Value.
HashFunction	Là hàm mã hoá một chiều (hàm băm).

Chương 1 Giới thiệu đề tài

1.1 Đặt vấn đề

Ngày nay, đối với học sinh, sinh viên và công chức thì việc thông thạo tiếng Anh là một nhiệm vụ gần như là tối thiểu và bắt buộc. Với việc Việt Nam gia nhập WTO và ngày càng vươn rộng ra thế giới thì phần lớn tất cả các công ty, tập đoàn trong cũng như ngoài nước đều đòi hỏi người làm việc cho họ phải thông thạo tiếng Anh. Do đó, nhu cầu học tiếng Anh trở nên cấp thiết và là xu thế chung của xã hội hiện nay. Tuy nhiên hiện nay học tiếng Anh với ai, ở đâu, như thế nào và có công cụ gì hỗ trợ không đang là câu hỏi khó dành cho các học viên.

1.2 Mục tiêu và phạm vi đề tài

- Giới thiệu một số từ điển thương mại phổ biến hiện nay:
    - Lạc việt mtd : có thêm xoá từ , tra từ click and see , chạy ổn định , dữ liệu từ điển tương đối đầy đủ .
    - Just click n see: chỉ có tra từ click and see(tốt hơn lạc việt) , rất ít tính năng.
    - Evtran 2.0 : có thêm xoá, nó là phần mềm dịch thì đúng hơn nhưng bản 3.0 không có khả năng thêm xoá, chức năng click and see lên vista thì liệt. các bạn có thể vào trang vdict.com để dịch trực tuyến miễn phí.
    - English Study 4.0: đây là phần mềm ngữ pháp tiếng anh + từ điển + luyện nghe.
    - babylon: không có khả năng thêm xoá nhưng khả năng click and see thuộc loại tốt nhất hiện nay cùng với khả năng tìm kiếm từ gần đúng hoàn hảo, nó rất mạnh.
    - prodict thuộc cùng một hãng, đặc điểm của loại này là dữ liệu lớn nhất hiện nay gồm nhiều chuyên ngành, nhưng không thêm, tra từ click and see ngang lạc việt.
- Mục tiêu: Em chọn đề tài này với mong muốn tạo một chương trình từ điển có nguồn dữ liệu phong phú, có âm thanh với tính năng đa dạng và phong phú, giao diện đẹp … để trở thành một công cụ hỗ trợ phát triển kỹ năng ngoại ngữ tiện dụng cho người dung.
- Các chức năng chính:
    - Tra từ Anh - Việt.

- Nghe phát âm từ.
- Lưu những từ mà người dung cần lưu vào Danh mục yêu thích để tiện sử dụng khi cần, dễ dàng cho việc học từ vựng, lưu xong nếu người dùng cảm thấy không cần thiết có thể xóa bỏ.
- Hiển thị danh mục yêu thích.
- Luyện tập trắc nghiệm với các từ có trong mục yêu thích.

1.3 Định hướng giải pháp;

Để lập trình một ứng dụng từ điển Anh-Việt cho PC thì em chọn lập trình WinForm với ngôn ngữ C# trên môi trường Visual Studio. Và em dùng bảng băm (HashTable) để làm từ điển vì đặc điểm của nó là dung mã băm để nhảy đến danh sách chứa vị trí nghĩa, khả năng này là nhảy trực tiếp đến nghĩa nên tốc độ sẽ rất nhanh.

Giải pháp cho từng chức năng chính:

- Dữ liệu của từ điển và mục yêu thích được lưu trong file txt. Chương trình sử dụng dữ liệu bằng cách đọc và thêm dữ liệu vào bảng băm dưới dạng key và value, với key là từ tiếng Anh và value là nghĩa của từ đó. Dữ liệu sẽ được chia thành từng nhóm với các từ trong nhóm có cùng giá trị HashFunction là tổng giá trị các chữ cái trong key là kiểu char được ép kiểu về dạng int.
- Tra từ Anh - Việt: Tính giá trị HashFunction của từ cần tìm kiếm, sau đó kiểm tra trong nhóm có cùng giá trị HashFunction có từ cần tìm kiếm không. Nếu có thì lấy giá trị value của từ vừa tìm được hiển thị ra mục kết quả tìm kiếm, nếu không thì hiển thị thông báo “không tìm thấy kết quả”.
- Phát âm từ tiếng Anh: Dùng thư viện phát âm của C# text to speech SpeechLib.
- Thêm từ vào danh mục yêu thích: với thêm từ vào mục yêu thích ta tính HashFunction, sau đó thêm từ vào nhóm có cùng HashFuntion. Khi ngừng sử dụng hệ thống lưu dữ liệu của bảng băm vào file txt.
- Hiển thị danh mục yêu thích: Dùng ComboBox của WinForm để hiển thị danh sách các từ có trong mục yêu thích, mỗi khi chọn một từ từ ComboBox thì hệ thống thực hiện tìm kiếm tương tự tra từ ở trên rồi hiển thị nội dung của từ ra màn hình.
- Xóa từ trong danh mục yêu thích: Chọn từ cần xóa rồi nhấn nút xóa, sau đó hệ thống sẽ tìm kiếm từ đó trong bảng băm tương tự như tra từ rồi xóa từ đó khỏi bảng băm. Khi ngừng sử dụng hệ thống lưu dữ liệu của bảng băm vào file txt.
- Luyện tập trắc nghiệm với các từ có trong mục yêu thích: Tương tự dùng ComboBox để hiển thị danh sách các từ có trong mục yêu thích, mỗi khi chọn một từ từ ComboBox thì hệ thống thực hiện tìm kiếm tương tự tra từ ở trên rồi lấy một phần

nghĩa của nó, sau đó lấy random 3 từ và lấy 1 phần nghĩa của 3 từ đó, rồi random nghĩa của từ và 3 nghĩa sai mà vừa được lấy để được 4 phương án cho bài trắc nghiệm. Kiểm tra đúng sai, hệ thống so sánh phương án người dùng chọn với đáp án và trả về thông báo.

- ❖ Kết quả đạt được:
  - Ứng dụng đã hoàn thành các yêu cầu được đặt ra.
  - Chương trình không cần người dung phải có kết nối internet khi sử dụng.
  - Chương trình có giao diện đơn giản, dễ sử dụng./

1.4 Bố cục đồ án

Phần còn lại của báo cáo Đồ án 3 (Project 3) này được tổ chức như sau.:

Chương 2 trình bày khảo sát hiện trạng, tổng quan và đặc tả các chức năng của usecase, quy trình nghiệp vụ và yêu cầu phi chức năng.

Trong Chương 3, em giới thiệu về ngôn ngữ lập trình C#, lập trình hướng đối tượng với C# và giới thiệu về WinForm. Chương 4 này sẽ trình bày về kiến trúc và thiết kế của phần mềm, xây dựng ứng dụng, tiến hành kiểm thử và triển khai.

Chương 5 trình bày tất cả những nội dung đóng góp mà em cảm thấy tâm đắc nhất trong suốt quá trình làm PROJECT 3 của mình. Sau đó chương 6 sẽ trình bày kết luận và hướng phát triển cho đề tài và sản phẩm của mình.

Chương 2 Khảo sát và phân tích yêu cầu

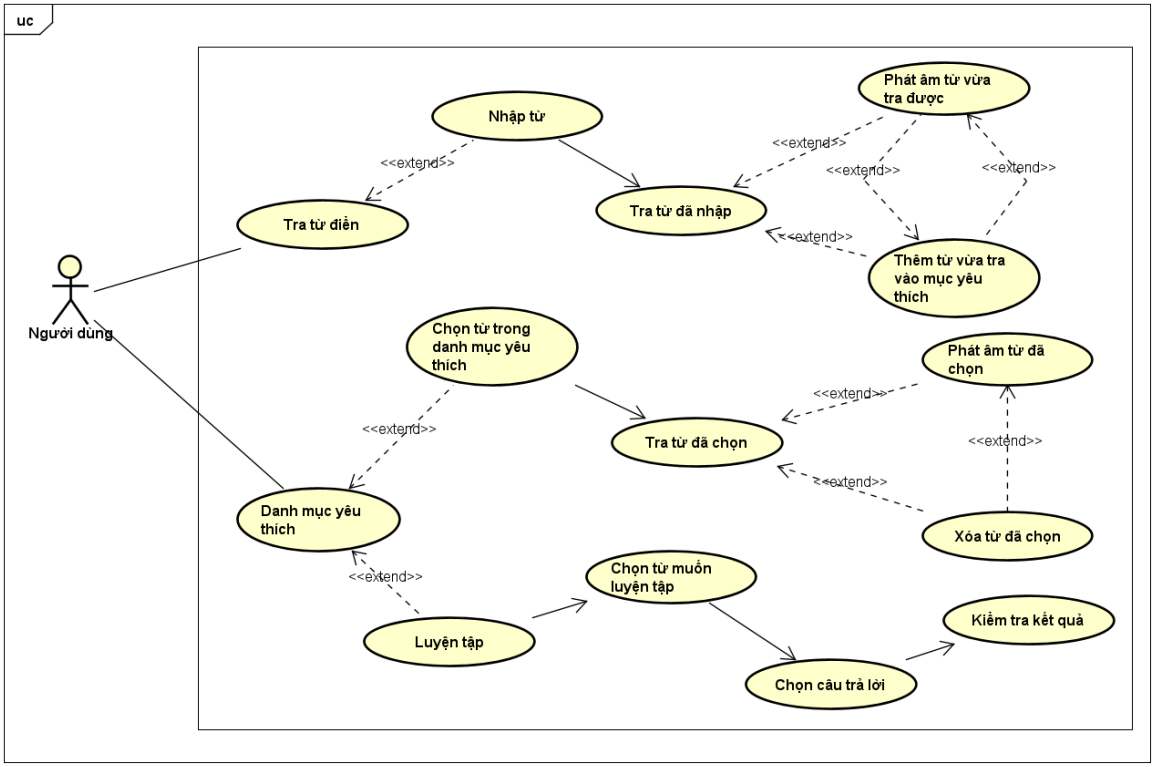
2.1 Khảo sát hiện trạng

Hiện nay, đối với học sinh, sinh viên và công chức thì việc thông thạo tiếng Anh là một nhiệm vụ gần như là tối thiểu và bắt buộc. Với việc Việt Nam gia nhập WTO và ngày càng vươn rộng ra thế giới thì phần lớn tất cả các công ty, tập đoàn trong cũng như ngoài nước đều đòi hỏi người làm việc cho họ phải thông thạo tiếng Anh. Do đó, nhu cầu học tiếng Anh trở nên cấp thiết và là xu thế chung của xã hội hiện nay. Và một công cụ hỗ trợ rất tốt cho việc học tiếng Anh chính là từ điển Anh Việt.

2.2 Tổng quan chức năng

- Tra từ Anh – Việt.
- Nghe phát âm từ.
- Phân tích nghĩa của một từ với nguồn dữ liệu phong phú gồm 109.000 từ.
- Lưu những từ mà người dung cần lưu vào Danh mục yêu thích để tiện sử dụng khi cần, dễ dàng cho việc học từ vựng, lưu xong nếu người dùng cảm thấy không cần thiết có thể xóa bỏ.
- Luyện tập trắc nghiệm với các từ có trong mục yêu thích.

2.2.1 Biểu đồ use case tổng quan



Hình 1: Biểu đồ use case tổng quan

Tên Actor	**Ý nghĩ / Ghi chú**
Người dung	Sử dụng các chức năng của hệ thống với vai trò là một người dung bình thường.

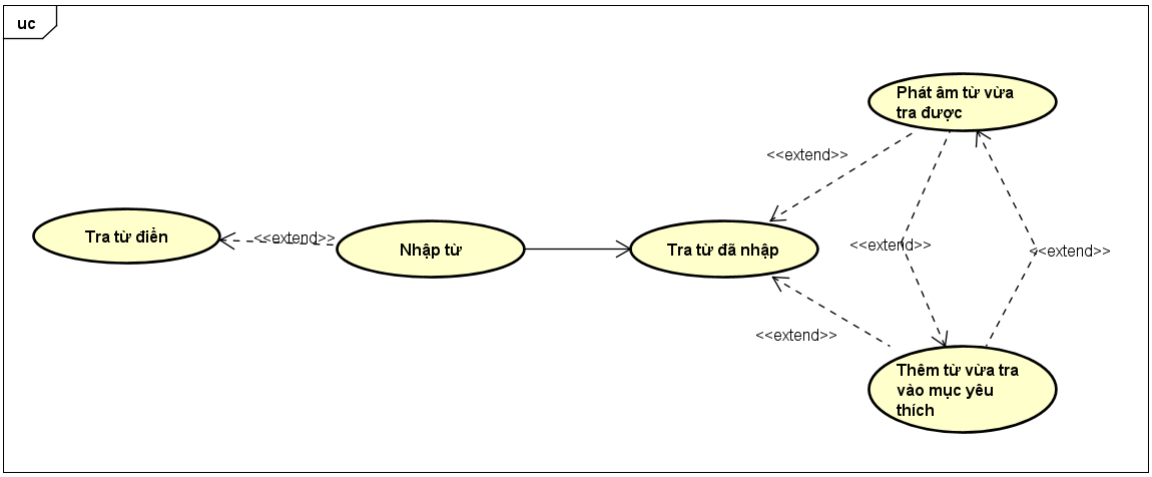
Bảng 1: Danh sách các Actor

Tên use case	**Ý nghĩ / Ghi chú**
Tra từ điển	Use case hệ thống sẽ giúp người dùng tra từ với từ điển Anh Việt.

Danh mục yêu thích	Use case hệ thống sẽ cho phép người dùng thêm từ trong từ điển vào danh mục yêu thích, xóa từ trong mục yêu thích và làm luyện tập trắc nghiệm với các từ có trong mục yêu thích.

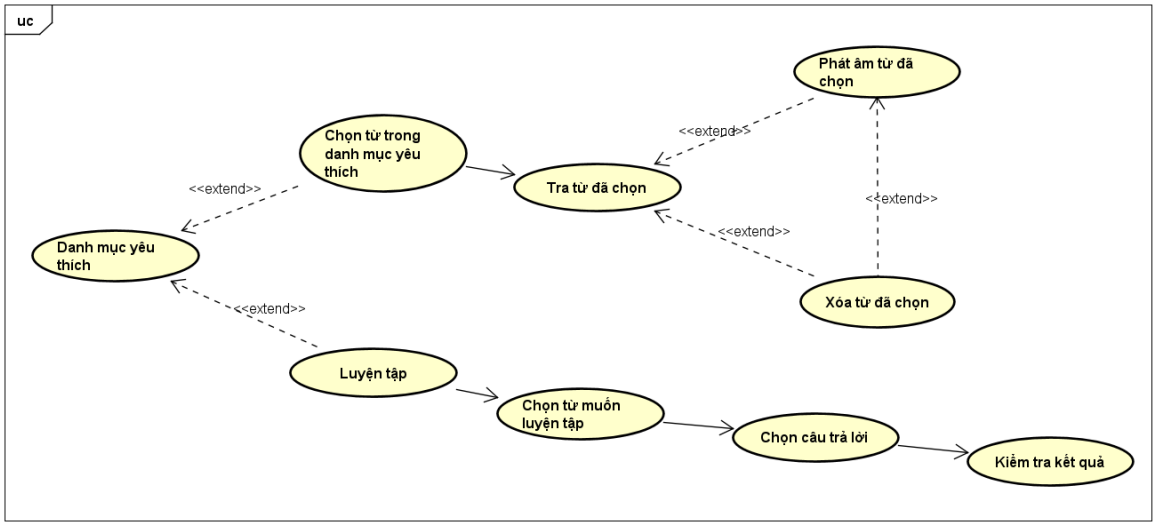
Bảng 2: Danh sách các use case

2.2.2 Biểu đồ use case phân rã "Tra từ điển"



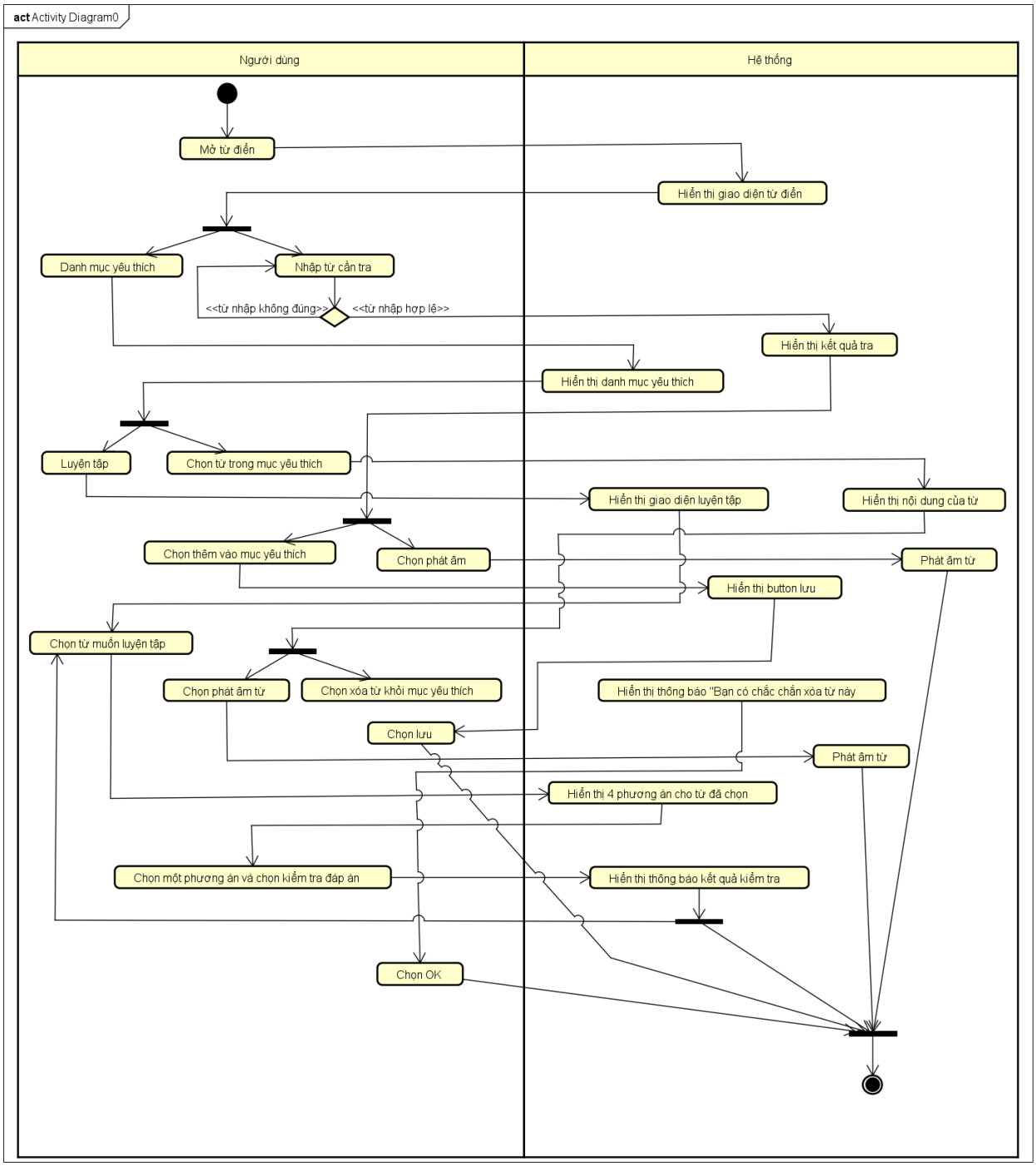
Hình 2: Biểu đồ use case phân rã "Tra từ điển"

2.2.3 Biểu đồ use case phân rã "Danh mục yêu thích"



Hình 3: Biểu đồ use case phân rã "Danh mục yêu thích"

2.2.4 Quy trình nghiệp vụ



Hình 4: Biểu đồ hoạt động minh họa quy trình nghiệp vụ

2.3 Đặc tả chức năng

2.3.1 Đặc tả use case UC001 “Tra từ điển”

Use Case “Tra từ điển”

1. Mã use case

UC001

2. Giới thiệu

Use case mô tả sự tương tác giữa người dùng và hệ thống khi người dùng muốn tra từ hoặc nghe phát âm từ.

3. Tác nhân

Người dùng

4. Tiền điều kiện

Không có

5. Luồng sự kiện chính (Thành công)

1. Người dùng mở từ điển
2. Hệ thống hiển thị giao diện từ điển
3. Người dùng nhập từ cần tra
4. Hệ thống hiển thị nội dung tra được
5. Người dung chọn thêm vào mục yêu thích
6. Hệ thống hiển thị button lưu
7. Người dùng chọn lưu

5a. Người dùng chọn chức năng phát âm từ

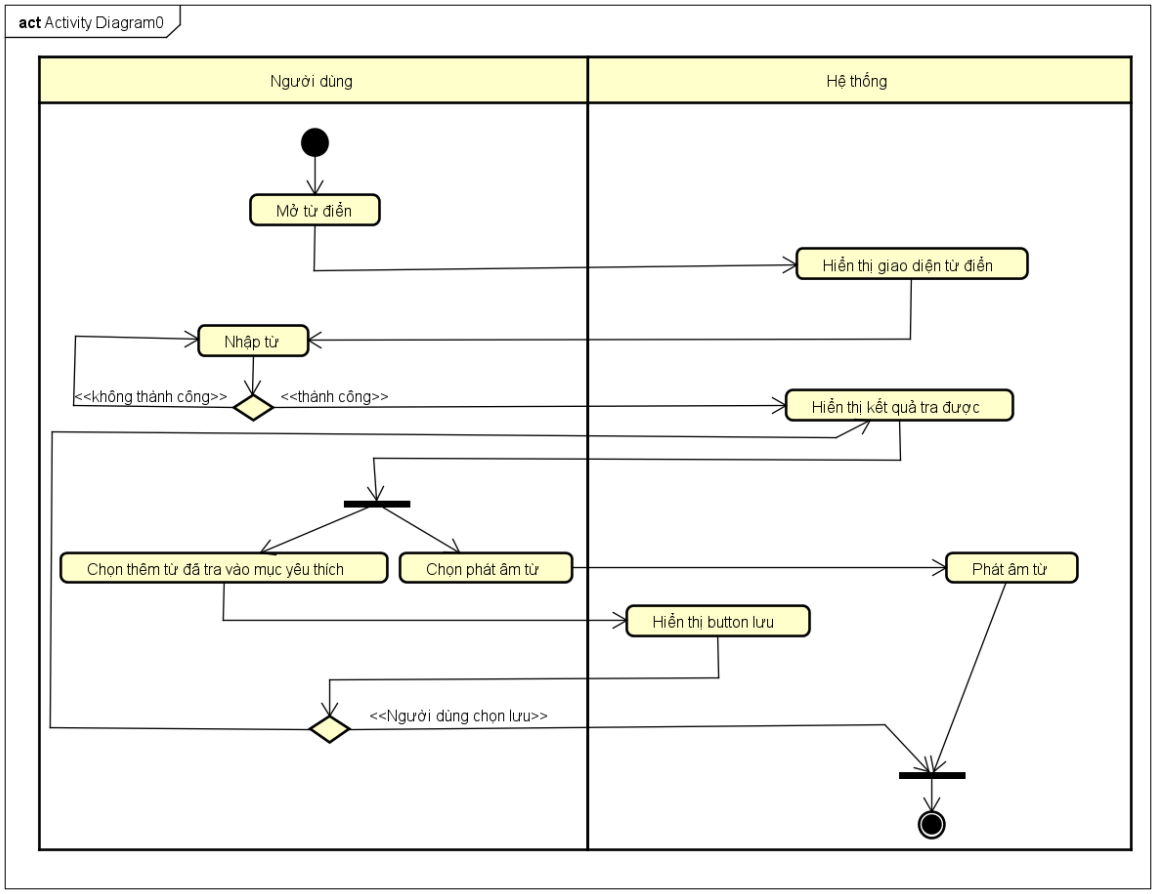
6a. Hệ thống phát âm từ

6. Luồng sự kiện thay thế

STT	Vị trí	Điều kiện	Hành động	Vị trí tiếp tục
1.	Tại bước 4	Nếu hệ thống không tìm được từ mà người dùng đã nhập.	Hệ thống thông báo lỗi: Không tìm thấy kết quả. Yêu cầu nhập lại.	Tiếp tục tại bước 3
2.	Tại bước 7	Nếu người dung không chọn lưu		Tiếp tục tại bước 4

Bảng 3: Luồng sự kiện thay thế của Use case "Sử dụng từ điển"

7. Biểu đồ hoạt động



Hình 5: Biểu đồ hoạt động của use case "Tra từ điển"

8. Dữ liệu đầu vào

STT	Trường dữ liệu	Mô tả	Bắt buộc?	Điều kiện hợp lệ	Ví dụ
1.	Từ cần tra		Có		hi

Bảng 4: Dữ liệu đầu vào

9. Dữ liệu đầu ra

STT	Trường dữ liệu	Mô tả	Bắt buộc?	Điều kiện hợp lệ	Ví dụ
1.	Kết quả tra		Có		/hai/ * thán từ - (từ mỹ,nghĩa mỹ) này!, ê! (gọi, chào)

Bảng 5: Dữ liệu đầu ra

10. Hậu điều kiện

Không

2.3.2 Đặc tả use case UC002 “Danh mục yêu thích”

Use Case “Danh mục yêu thích”

1. Mã use case

UC002

2. Giới thiệu

Use case mô tả sự tương tác giữa người dùng và hệ thống khi người dùng muốn thêm, sửa hoặc xóa để cập nhật từ điển.

3. Tác nhân

Người dùng

4. Tiền điều kiện

Không có

5. Luồng sự kiện chính (Thành công)

1. Người dùng mở từ điển
2. Hệ thống hiển thị giao diện từ điển
3. Người dùng chọn danh mục yêu thích
4. Hệ thống hiển thị giao diện danh mục yêu thích
5. Người dùng chọn từ trong danh mục yêu thích
6. Hệ thông hiển thị nội dung từ
7. Người dùng chọn xóa từ
8. Hệ thống hiển thị thông báo “Bạn có muốn xóa từ này”
9. Người dùng chọn “OK”

5a. Người dùng chọn “Luện tập”
6a. Hiển thị giao diện luyện tập
7a. Người dùng chọn từ muốn luyện tập
8a. Hiển thị 4 phương án cho từ đã chọn
9a. Người dùng chọn 1 phương án và kiểm tra đáp án
10a. Hệ thống kiểm tra phương án và hiển thị thông báo

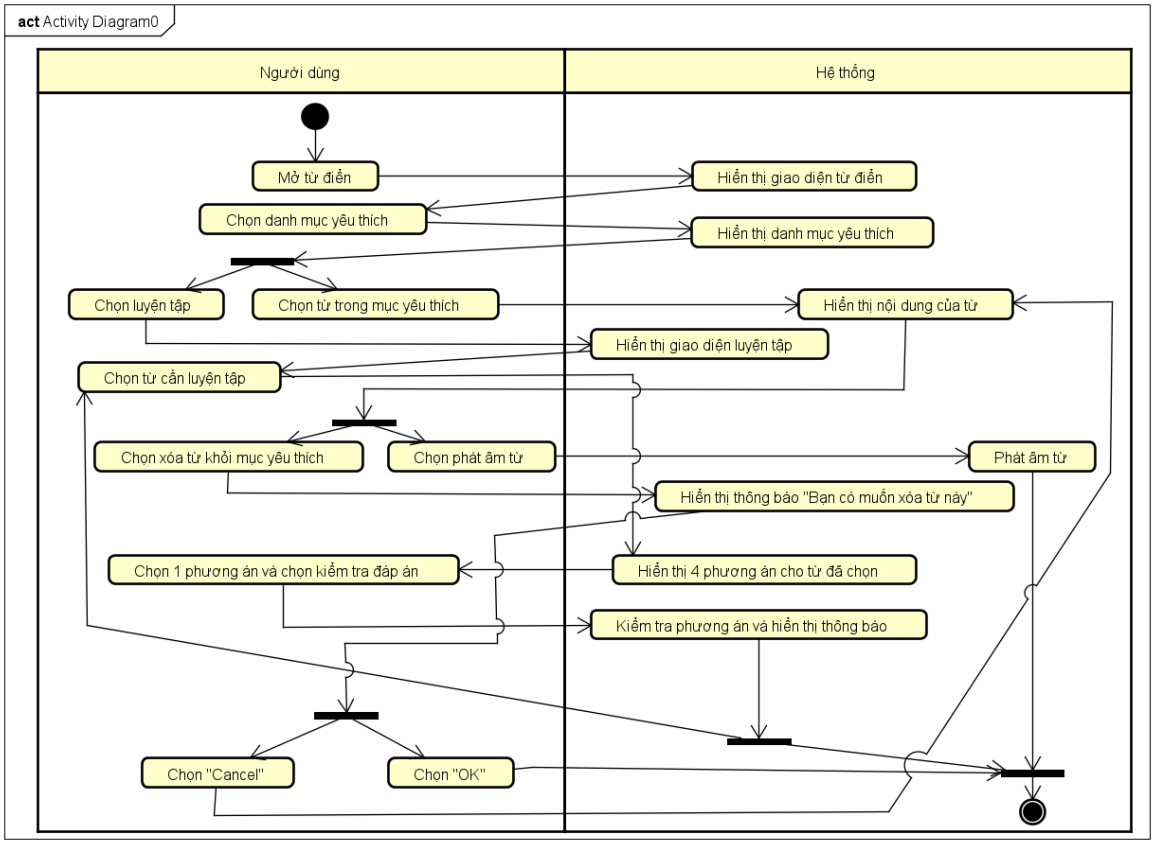
7b. Người dung chọn phát âm từ
8b. Hệ thống phát âm từ.

6. Luồng sự kiện thay thế

STT	Vị trí	Điều kiện	Hành động	Vị trí tiếp tục
1.	Tại bước 9	Nếu người dung chọn “Cancel”.		Tiếp tục tại bước 6

Bảng 6: Luồng sự kiện thay thế của Use case "Danh mục yêu thích"

7. Biểu đồ hoạt động



Hình 6: Biểu đồ hoạt động của use case "Danh mục yêu thích"

8. Dữ liệu đầu vào

STT	Trường dữ liệu	Mô tả	Bắt buộc?	Điều kiện hợp lệ	Ví dụ
1.	Từ được chọn		Có		hi

Bảng 7 : Dữ liệu đầu vào

9. Dữ liệu đầu ra

STT	Trường dữ liệu	Mô tả	Bắt buộc?	Điều kiện hợp lệ	Ví dụ
1.	Nội dung từ		Có		/hai/ * thán từ - (từ mỹ,nghĩa mỹ) này!, ê! (gọi, chào)

Bảng 8 : Dữ liệu đầu ra

10. Hậu điều kiện

Không

2.4 Yêu cầu phi chức năng

- Giao diện: giao diện cần đơn giản, bố trí khoa học.
- Hoạt động chương trình: Chương trình hoạt động ổn định, tránh gây sai sót trong khi hoạt động. Đáp ứng đủ các yêu cầu cơ bản của một chương trình từ điển Anh - Việt.
- Cài đặt: Dễ dàng cài đặt, không đòi hỏi cấu hình máy cao.

Chương 3 Công nghệ sử dụng

3.1 Ngôn ngữ lập trình C#

3.1.1 Giới thiệu

C# (C Sharp, đọc là "xi-sáp") là một ngôn ngữ lập trình hướng đối tượng đa năng vô cùng mạnh mẽ được phát triển bởi Microsoft, C# là phần khởi đầu cho kế hoạch .NET của họ. Tên của ngôn ngữ bao gồm ký tự thăng theo Microsoft nhưng theo ECMA là C#, chỉ bao gồm dấu số thường. Microsoft phát triển C# dựa trên C++ và Java. C# được miêu tả là ngôn ngữ có được sự cân bằng giữa C++, Visual Basic, Delphi và Java.

C# được thiết kế chủ yếu bởi Anders Hejlsberg kiến trúc sư phần mềm nổi tiếng với các sản phẩm Turbo Pascal, Delphi, J++, WFC. Phiên bản gần đây nhất là 9.0, được phát hành vào năm 2020 cùng với Visual Studio 2019 phiên bản 16.8.

3.1.2 Mục tiêu của việc phát triển C#

Tiêu chuẩn ECMA liệt kê các mục tiêu của việc thiết kế ngôn ngữ C#:

- Ngôn ngữ được dự định là một ngôn ngữ lập trình đơn giản, hiện đại, hướng đến nhiều mục đích sử dụng, và là một ngôn ngữ lập trình hướng đối tượng.
- Ngôn ngữ và việc triển khai đáp ứng các nguyên tắc của ngành kỹ thuật phần mềm như kiểm tra chặt chẽ kiểu dữ liệu, kiểm tra giới hạn mảng, phát hiện các trường hợp sử dụng các biến chưa có dữ liệu, và tự động thu gom rác. Tính mạnh mẽ, sự bền bỉ, và năng suất của việc lập trình là rất quan trọng đối với ngôn ngữ này.
- Ngôn ngữ sẽ được sử dụng để phát triển các thành phần của phần mềm theo hướng thích hợp cho việc triển khai trong các môi trường phân tán.
- Khả năng di chuyển (portability) là rất quan trọng, đặc biệt là đối với những lập trình viên đã quen với C và C++.
- Hỗ trợ quốc tế hóa (i18n).
- Ngôn ngữ sẽ được thiết kế để phù hợp với việc viết các ứng dụng cho cả hai hệ thống: hosted và nhúng, từ các phần mềm quy mô lớn, đến các phần mềm chỉ có các chức năng đơn giản.
- Mặc dù các ứng dụng C# có tính kinh tế đối với các yêu cầu về bộ nhớ và chế độ xử lý, ngôn ngữ này không cạnh tranh trực tiếp về hiệu năng và kích thước đối với ngôn ngữ C hoặc assembly.

3.1.3 Ứng dụng của C#

- Phát triển web fullstack backend, front end (ASP.NET MVC, ASP.NET core, Web API, Blazor..)
- Phát triển desktop app (Winform, WPF, UWP, Mono, Uno,MAUI...)

- Phát triển game (Unity, Monogame, Godot, Stride, CryEngine..)
- Phát triển mobile app, IOS native, Android native (Xamarin)
- Phát triển đám mây (Azure,AWS...)
- Phát triển thực tế ảo (VR) và thực tế tăng cường(AR) (HoloLens,CryEngine,Unity, Oculus quest..)
- Học máy và trí tuệ nhân tạo (ML.Net, TensorFlow, csiSharp..)
- Blockchain (NEO, Stratis)
- Microservices and containers
- Internet of thing (IoT,5G)

3.1.4 Đặc điểm ngôn ngữ

C#, theo một hướng nào đó, là ngôn ngữ lập trình phản ánh trực tiếp nhất đến .NET Framework mà tất cả các chương trình.NET chạy, và nó phụ thuộc mạnh mẽ vào framework này. Mọi dữ liệu cơ sở đều là đối tượng, được cấp phát và hủy bỏ bởi trình dọn rác Garbage-Collector (GC), và nhiều kiểu trừu tượng khác chẳng hạn như class, delegate, interface, exception... phản ánh rõ ràng những đặc trưng của.NET runtime.

So sánh với C và C++, ngôn ngữ này bị giới hạn và được nâng cao ở một vài đặc điểm nào đó, nhưng không bao gồm các giới hạn sau đây:

- Các con trỏ chỉ có thể được sử dụng trong chế độ không an toàn. Hầu hết các đối tượng được tham chiếu an toàn, và các phép tính đều được kiểm tra tràn bộ đệm. Các con trỏ chỉ được sử dụng để gọi các loại kiểu giá trị; còn những đối tượng thuộc bộ gom rác (garbage-collector) thì chỉ được gọi bằng cách tham chiếu.
- Các đối tượng không thể được giải phóng tường minh.
- Chỉ có đơn kế thừa, nhưng có thể cài đặt nhiều interface trừu tượng (abstract interfaces). Chức năng này làm đơn giản hóa sự thực thi của thời gian thực thi.
- C# thì an-toàn-kiểu (typesafe) hơn C++.
- Cú pháp khai báo mảng khác nhau("int[] a = new int[5]" thay vì "int a[5]").
- Kiểu thứ tự được thay thế bằng tên miền không gian (namespace).
- C# không có tiêu bản.
- Có thêm Properties, các phương pháp có thể gọi các Properties để truy cập dữ liệu.
- Có reflection.

3.1.5 Đặc trưng của ngôn ngữ C#

C# là ngôn ngữ đơn giản, mạnh mẽ

- C# được dựng trên nền tảng C++ và Java, ảnh hưởng bởi Delphi, VisualBasic nên ngôn ngữ C# được thừa hưởng các ưu điểm vào loại bỏ các yếu điểm của các ngôn ngữ trên, vì vậy nó khá đơn giản, đồng thời loại bỏ các cú pháp dư thừa và thêm vào đó các cú pháp cải tiến hơn
- C# là ngôn ngữ lập trình bậc cao, đa nền tảng vì vậy nó dễ dàng tiếp cận và phù hợp cho người mới bắt đầu học, ví dụ câu lệnh kinh điển dành cho người mới bắt

đầu hoc là in ra dòng chữ "Hello world", với C# ta chỉ cần 1 câu lệnh: System.Console.WriteLine("Hello world");

C# là ngôn ngữ đa năng và hiện đại

- C# phù hợp cho việc phát triển trong thời đại 4.0, bao gồm việc phát triển web, mobile app, game, học máy và trí tuệ nhân tạo, phát triển đám mây, IoT, blockchain, microservices...

C# là một ngôn ngữ lập trình hướng đối tượng đồng thời hỗ trợ lập trình chức năng

- C# hỗ trợ mạnh mẽ cho phương pháp lập trình hướng đối tượng, ngoài ra C# còn hỗ trợ các phương pháp lập trình chức năng thông qua các biểu thức lamba, khớp mẫu, functions, các thuộc tính bất biến.

C# là ngôn ngữ gõ tĩnh, định kiểu mạnh, hỗ trợ gõ động.

- C# được gõ tĩnh nên nó mang đầy đủ các ưu việt của phương pháp gõ tĩnh như bảo đảm an toàn kiểu, tự động phân tích và nhận biết lỗi cú pháp ngay trong quá trình viết mã...
- Ngoài ra khi sử dụng C# kết hợp với IDE Visual Studio, C# được hỗ trợ gợi ý code bởi Visual Studio IntelliCode sử dụng trí tuệ nhân tạo giúp cho việc viết code trở nên nhanh chóng và dễ dàng hơn

C# là một ngôn ngữ ít từ khóa

- C# có khoảng hơn 80 từ khóa

C# là một trong các ngôn ngữ lập trình phổ biến và phát triển nhất

- Theo TIOBE Index, tính đến tháng 10/2020, C# là ngôn ngữ phổ biến thứ 5 thế giới.
- Theo PYPL, tính đến tháng 10/2020, C# là ngôn ngữ được cộng đồng quan tâm và chia sẻ nhiều thứ 4 thế giới.
- Cộng đồng phát triển và số người theo học ngôn ngữ C# tăng khônng ngừng theo mỗi năm.Theo ước tính 10/2020, cộng đồng phát triển C# là hơn 6 triệu người

C# kết hợp chặt chẽ với nền tảng .NET - một khung nền tảng rất mạnh của Microsoft.

Ngoài ra C# còn có những ưu điểm:

- C# là ngôn ngữ lập trình mã nguồn mở,vì vậy C# là miễn phí với tất cả mọi người, đồng thời mọi người đều có thể cùng tham gia phát triển, đề xuất thiết kế ngôn ngữ C#
- C# là ngôn ngữ đa nền tảng vì vậy có thể biên dịch trên nhiều nền tảng máy tính khác nhau (Windows, Linux, MacOS)
- C# có hiệu suất cao và tốc độ thực thi nhanh do sử dụng trình biên dich trung gian, điểm cộng nữa là tốc độ phát triển phần mềm nhanh chóng so với đa số các ngôn ngữ hiện tại.

- C# có IDE Visual Studio cùng nhiều plug-in vô cùng mạnh mẽ. ngoài ra có thể viết C# bằng bất kỳ text editor nào khác như Visual studio code, Vim, Netbeam...
- C# có cấu trúc khá gần gũi với các ngôn ngữ lập trình truyền thống, song cũng được bổ sung các yếu tố mang tính hiện đại nên dễ dàng tiếp cận cho người mới học và học nhanh với C#.
- C# có cộng đồng nhà phát triển vô cùng lớn mạnh.
- C# được phát triển và cải tiến không ngừng với tần suất 1 phiên bản/ 1 năm, đáp ứng các mong muốn cải thiện, cải tiến cho phù hợp với nhu cầu công nghệ của các nhà phát triển.
- C# có tài liệu tham khảo và hướng dẫn vô cùng phong phú và chất lượng, đồng thời có các buổi hội thảo giới thiệu tính năng mới và định hướng phát triển ngôn ngữ trong tương lai.
- C# và.NET được đánh giá là có design tốt, vì vậy cú pháp và logic rất nhất quán, mã nguồn C# dễ đọc và mở rộng.
- C# được thiết kế và phát triển bởi Microsoft nên rất được Microsoft quan tâm và hỗ trợ.

3.1.6 Lập trình hướng đối tượng với C#

3.1.6.1 Lớp và đối tượng

a) Đối tượng

Trong lập trình hướng đối tượng, tất cả thực thể trong hệ thống đều được coi là các đối tượng cụ thể. Đối tượng là một thực thể hoạt động khi chạy chương trình.

Một đối tượng là một thực thể đang tồn tại trong hệ thống và được xác định bằng ba yếu tố:

- **Định danh đối tượng:** xác định duy nhất cho mỗi đối tượng trong hệ thống, nhằm phân biệt các đối tượng với nhau.
- **Trạng thái của đối tượng:** là sự tổ hợp của các giá trị của các thuộc tính mà đối tượng đang có.
- **Hoạt động của đối tượng:** là các hành động mà đối tượng có khả năng thực hiện được.

b) Class (lớp)

Định nghĩa lớp: Trong lập trình hướng đối tượng, đối tượng là một thực thể cụ thể, tồn tại trong hệ thống. Trong khi đó, lớp là một khái niệm trừu tượng, dùng để chỉ một tập hợp các đối tượng cùng loại.

Khai báo một lớp:

[attribute][bổ từ truy xuất] class định danh [: lớp cơ sở]

{

thân lớp

}

Lớp và đối tượng

Lớp và đối tượng mặc dù có mối tương quan, nhưng bản chất lại khác nhau:

- Lớp là sự trừu tượng hóa các đối tượng. Trong khi đó, đối tượng là thể hiện cụ thể của một lớp.
- Đối tượng là một thực thể cụ thể, có thực, tồn tại trong hệ thống. Trong khi đó lớp là một khái niệm trừu tượng, chỉ tồn tại ở dạng khái niệm để mô tả các đặc tính chung của một nhóm đối tượng.
- Tất cả các đối tượng thuộc cùng một lớp giống nhau về thuộc tính và phương thức.

3.1.6.2 Kế thừa lớp

Một tính năng then chốt của lập trình hướng đối tượng đó là tính kế thừa. Nhờ vào tính kế thừa, nó cho phép một lớp có thể dẫn xuất từ một lớp khác hay nói cách khác một lớp có thể sử dụng lại các thuộc tính và phương thức của lớp bị kế thừa, chính vì thế chúng sẽ tự động tiếp nhận các thành viên của bố mẹ và bổ sung thêm các thành viên của riêng chúng. Tính kế thừa cho phép lớp mới có thể nhận mọi dữ liệu thành viên (private, protected, public) và các hàm thành viên (trừ hàm tạo, hàm hủy, hàm bạn và hàm toán tử gán =).

Khai báo một lớp kế thừa

Trong C#, khi ta tạo một lớp kế thừa bằng cách thêm một dấu ":" vào sau tên của lớp kế thừa và theo sau đó là lớp cơ sở như sau:

public class <Tên lớp dẫn suất> : <Tên lớp cơ sở>

3.1.6.3 Giao diện (interface)

Giao diện giống như một khuôn mẫu, các class sử dụng giao diện đều phải thực hiện tất cả các phương thức của giao diện , điều này giúp đồng nhất về phương thức giúp các lớp khác nhau có thể làm việc được với nhau.

Cú pháp của việc định nghĩa một giao diện:

[attributes] [access-modifier] interface interface-name [:base-list]

{

interface-body

}

Ý nghĩa của từ thành phần như sau:

- **attributes**: sẽ đề cập ở phần sau.
- **modifiers**: bổ từ phạm vi truy xuất của giao diện.
- **identifier**: tên giao diện muốn tạo.
- **base-list**: danh sách các giao diện mà giao diện này thừa kế.
- **interface-body**: than giao diện luôn nằm giữa cặp dấu {}.

Cài đặt nhiều giao diện:

Lớp có thể và đặt một hoặc nhiều giao diện.

Mở rộng giao diện

Chúng ta có thể mở rộng (thừa kế) một giao diện đã tồn tại bằng cách thêm vào đó những phương thức hoặc thành viên mới.

Kết hợp các giao diện khác nhau

Tương tự, chúng ta có thể tạo một giao diện mới bằng việc kết hợp nhiều giao diện và ta có thể tùy chọn việc có thêm nhưng phương thức hoặc thuộc tính mới.

3.1.6.4 Không gian tên (Namespace)

Namespace trong ngôn ngữ C#, nhằm tránh sự xung đột giữa việc sử dụng các thư viện khác nhau từ các nhà cung cấp. Ngoài ra, namespace được xem như là tập hợp các lớp đối tượng và cung cấp duy nhất các định dạng cho các kiểu dữ liệu và được đặt trong một cấu trúc phân cấp. Việc sử dụng namespace trong khi lập trình là một thói quen tốt, bởi vì công việc này chính là cách lưu các mã nguồn để sử dụng về sau. Ngoài thư viện namespace do MS.NET và các hãng thứ ba cung cấp, ta có thể tạo riêng cho mình các namespace. C# đưa ra từ khóa using để khai báo sử dụng namespace trong chương trình:

Using < Tên namespace >

Để tạo một namespace dung cú pháp sau:

Namespace <Tên namespace>

{

< Định nghĩa lớp A>

< Định nghĩa lớp B>

…

}

C# còn cho phép trong một namespace có thể tạo một namespace khác lồng bên trong và không giới hạn mức độ phân cấp này.

3.1.7 Giới thiệu về Windows Form

Windows Forms hay viết tắt là WinForm là thuật ngữ chỉ việc phát triển các ứng dụng giao diện người dùng bằng cách sử dụng các thành phần xây dựng sẵn (buil in component) còn được gọi là các điều khiển. Hay nói cách khác Windows Forms là một API cho phép tạo GUI cho các ứng dụng chạy trên desktop.

Các điều khiển này dùng để hiển thị thông tin cho người dùng cũng như cho người dùng nhập thông tin vào Windows Forms ra đời đáp ứng nhu cầu tạo ứng dụng nhanh (Rapid Application Development).

- Windows Form cho phép người phát triển tạo ra các giao diện người dùng sử dụng các thành phần khác nhau (components). Các thành phần này còn được gọi là các điều khiển (controls). Những điều khiển này cho phép chúng ta thu thập thông tin từ người dùng cũng như trình bày các thông tin để người dùng có thể xem.
- Một Form được chạy trên một máy tính cục bộ (local machine) và một form có thể truy cập đến các tài nguyên khác nhau như bộ nhớ, các thư mục, các tệp tin, các cơ sở dữ liệu…
- Do đó Windows Form phù hợp cho các ứng dụng desktop như các ứng dụng quản lý thông tin, các ứng dụng tương tác trực tiếp với người dùng.
- Vai trò của Windows Form:
    - Các Form có thể chứa các điều khiển (các thành phần) khác nhau.
    - Xử lý dữ liệu được nhập bởi người dùng.
    - Hiển thị (trình bày) các thông tin tới người dùng.
    - Kết nối đến các nguồn CSDL khác nhau trên các máy tính cục bộ hoặc máy tính khác

Chương 4 Phát triển và triển khai ứng dụng

4.1 Thiết kế kiến trúc

4.1.1 Lựa chọn kiến trúc phần mềm

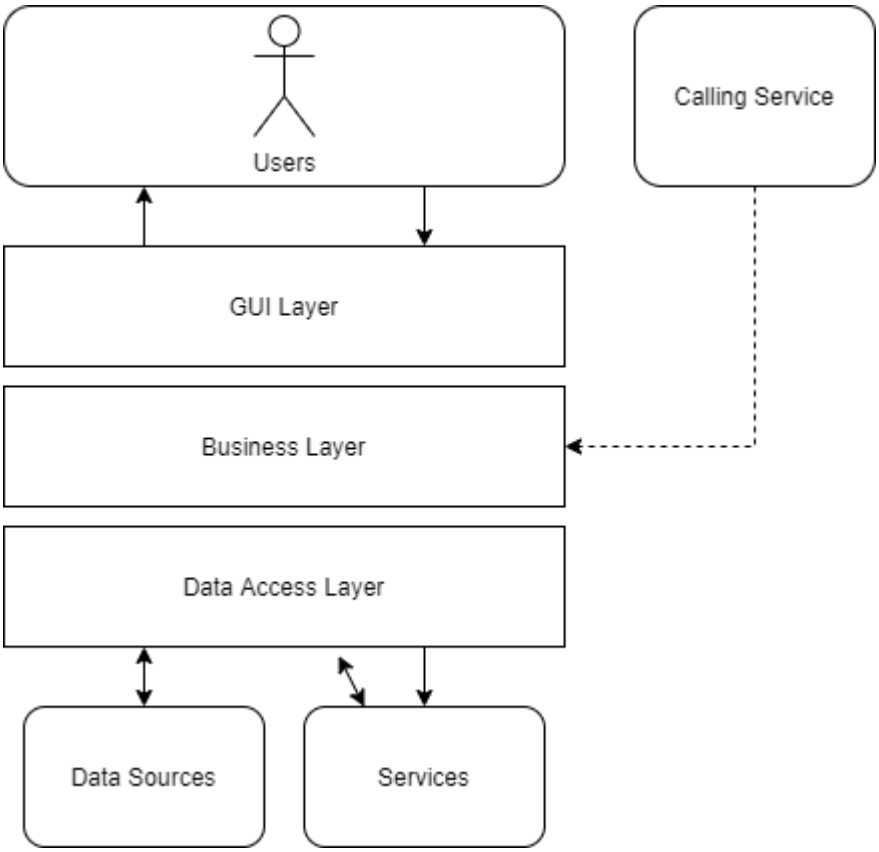
Em lựa chọn mô hình 3 lớp, gồm 3 lớp là:

- GUI Layer: hiển thị các thành phần giao diện để tương tác với người dùng như tiếp nhận thông tin, thông báo lỗi, …
- Business (BUS) Layer: thực hiện các hành động nghiệp vụ của phần mềm như tính toán, đánh giá tính hợp lệ của thông tin, … Tầng này còn di chuyển, xử lí thông tin giữa 2 tầng trên dưới.
- Data Access Layer: nơi lưu trữ và trích xuất dữ liệu từ các hệ quản trị CSDL hay các file trong hệ thống, chỉ duy nhất lớp này được làm việc với database. Cho phép tầng Business logic thực hiện các truy vấn dữ liệu.

Trong kiến trúc phần mềm của em, những thành phần cụ thể của từng tầng như sau:

- GUI Layer: frmDictionary, frmFavorite, frmLearn.
- Business (BUS) Layer: MyDictionary, MyHashTable, LINKEDLIST, Node.
- Data Access Layer: MyData.

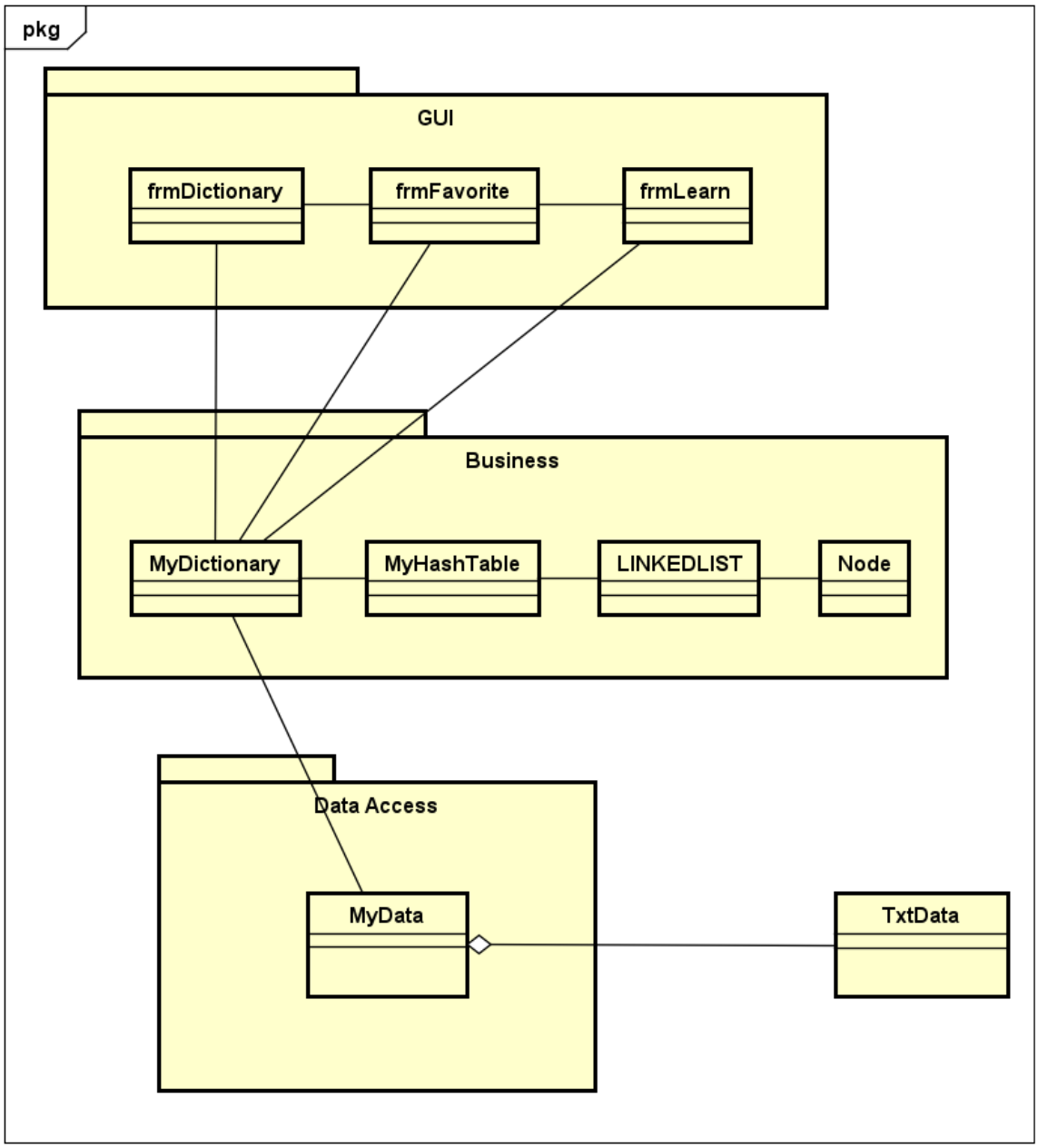
4.1.2 Thiết kế tổng quan



Hình 7: Biểu đồ phụ thuộc gói

- GUI Layer: package này là lớp hiển thị giao diện và các chức năng để người dung cuối sử dụng.
- Business (BUS) Layer: package này là lớp nhận các yêu cầu từ lớp GUI và truy xuất lên lớp Data để lấy thông tin và trả về GUI.
- Data Access Layer: package này là lớp để truy xuất với CSDL, chỉ duy nhất lớp này được làm việc với database.

4.1.3 Thiết kế chi tiết gói



Hình 8: Biểu đồ gói chi tiết

4.2 Thiết kế chi tiết

4.2.1 Thiết kế giao diện

Display

Số lượng màu được hỗ trợ: 16,777,216 màu

Độ phân giải: 800 x 500 pixels

Screen

Vị trí của button: Ở dưới cùng (theo chiều dọc) và căn giữa (theo chiều ngang) của khung.

Vị trí của message: Ở giữa khung màn hình.

Vị trí của screen title: Title đặt ở góc trên bên trái của khung hình.

Sự nhất quán trong hiển thị chữ số: dấu phẩy để phân cách hàng nghìn và chuỗi chỉ bao gồm các ký tự, chữ số, dấu phẩy, dấu chấm, dấu cách, dấu gạch dưới và ký hiệu gạch nối.

Control

Kích thước text: medium size (10px). Font: Microsoft Sans Serif. Font style: Bold.

Xử lý check input: Nên kiểm tra xem input có empty hay không, tiếp theo kiểm tra input có kí tự đặc biệt không.

Dịch chuyển màn hình: Các màn hình được tách biệt và màn hình chính ở dưới sẽ không thể thao tác trong khi có một màn hình khác đang được hiển thị.

Thứ tự các màn hình trong hệ thống

Màn hình cho chức năng “Tra từ điển”

Màn hình cho chức năng “Danh mục yêu thích”

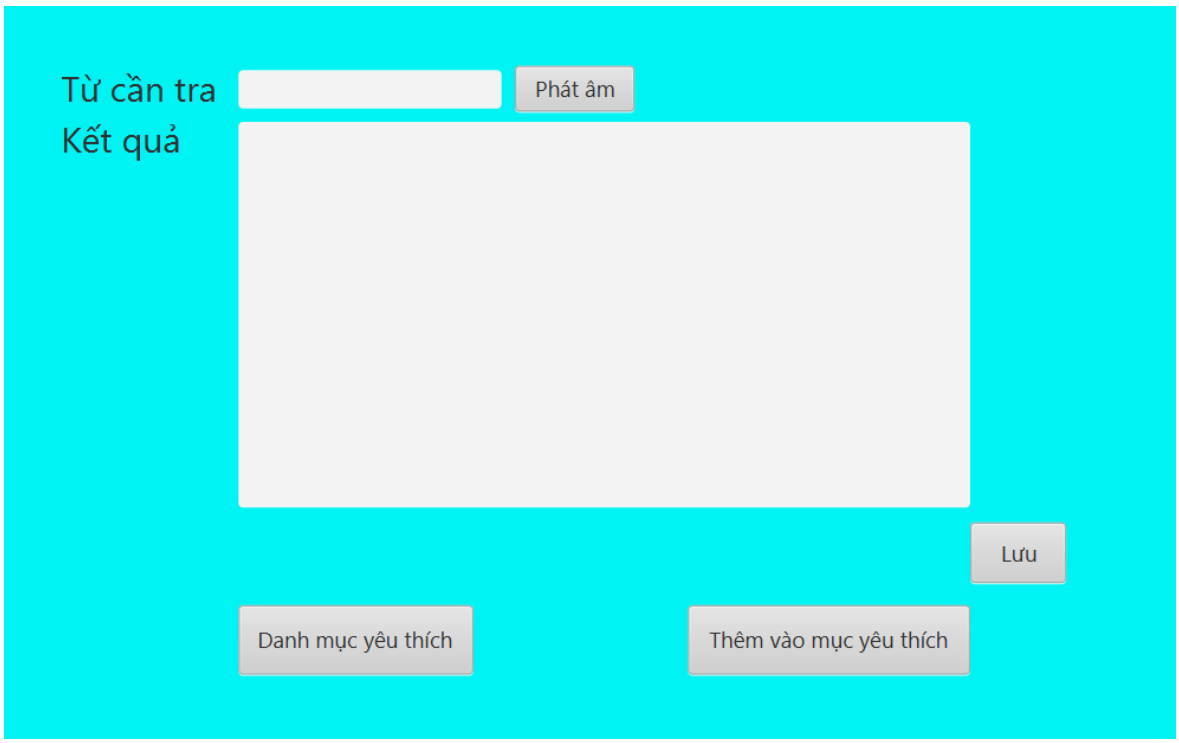
Màn hình cho chức năng “Luyện tập”

Nhập input từ bàn phím

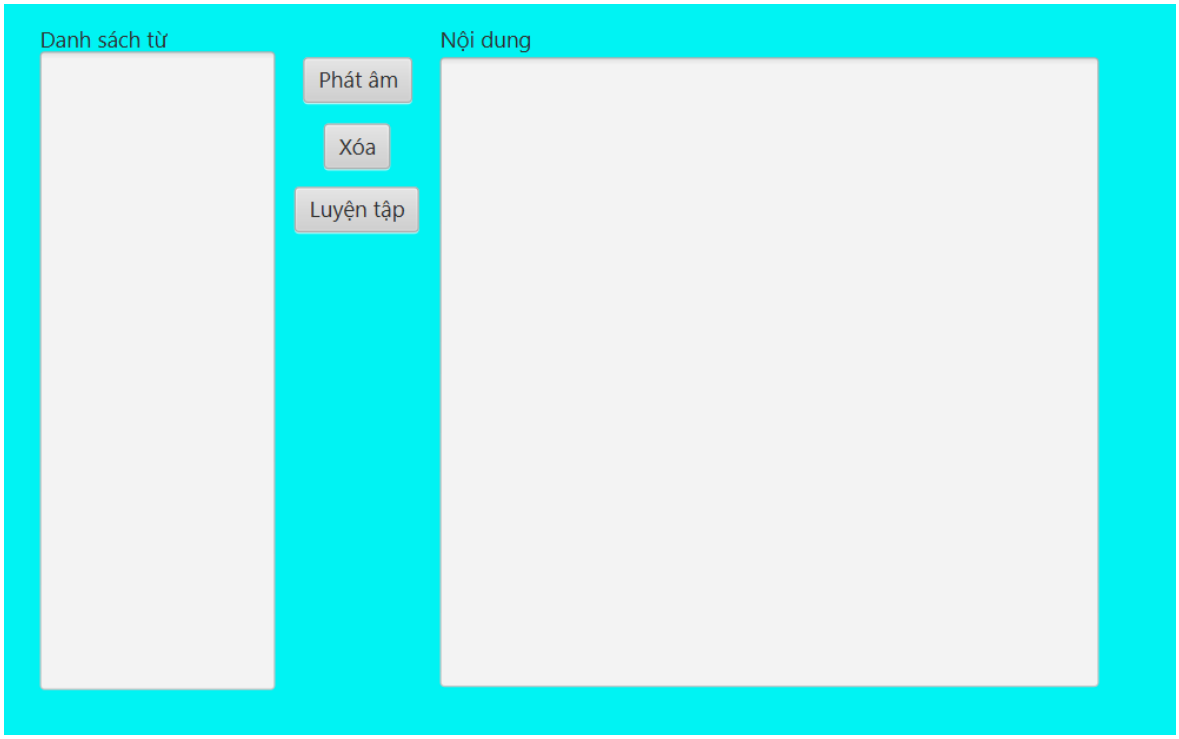
Sẽ không có phím tắt. Có button “X” nằm ở thanh tiêu đề bên phải để đóng screen đang hiển thị và quay lại screen trước đó.

Error

Một thông điệp sẽ được hiện lên để thông báo cho người dùng biết vấn đề đang gặp phải là gì.



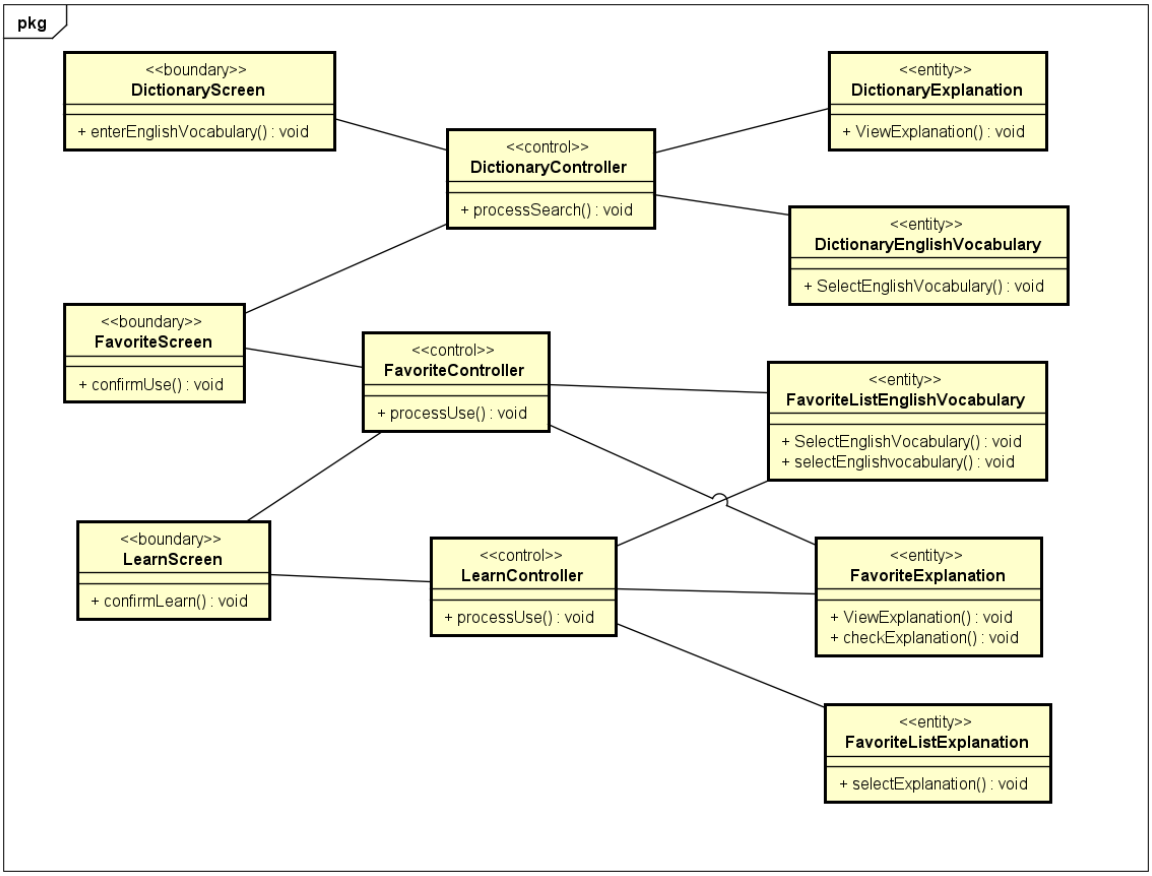
Hình 9: Hình ảnh minh họa thiết kế giao diện cho chức năng "Tra từ điển"



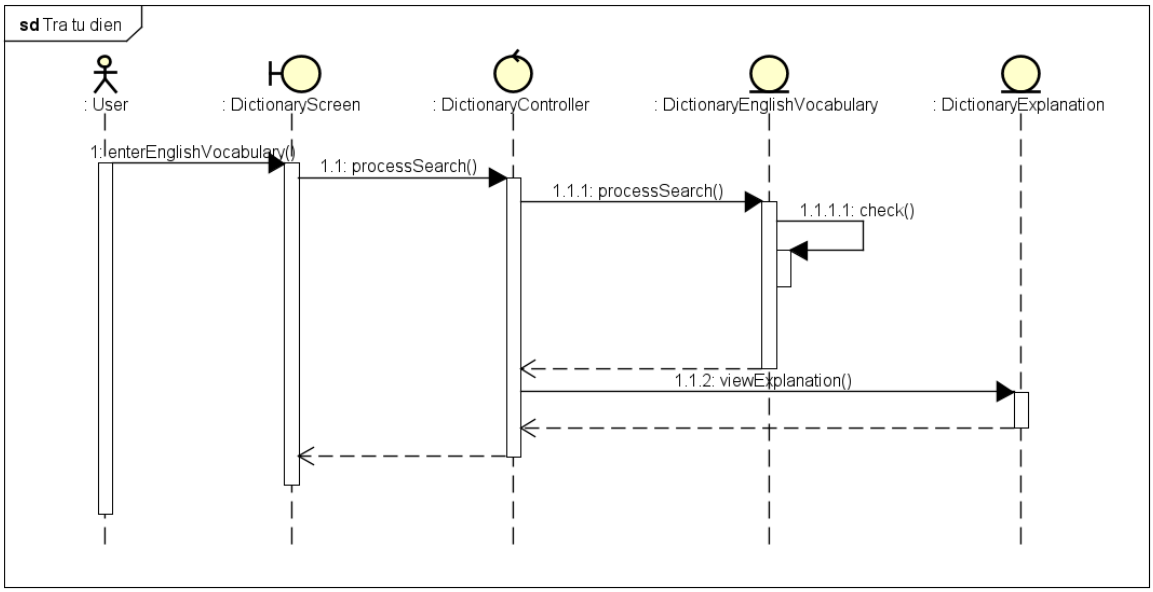
Hình 10: Hình ảnh minh họa thiết kế giao diện cho chức năng "Danh mục yêu thích"

Hình 11: Hình ảnh minh họa thiết kế giao diện cho chức năng “Luyện tập”

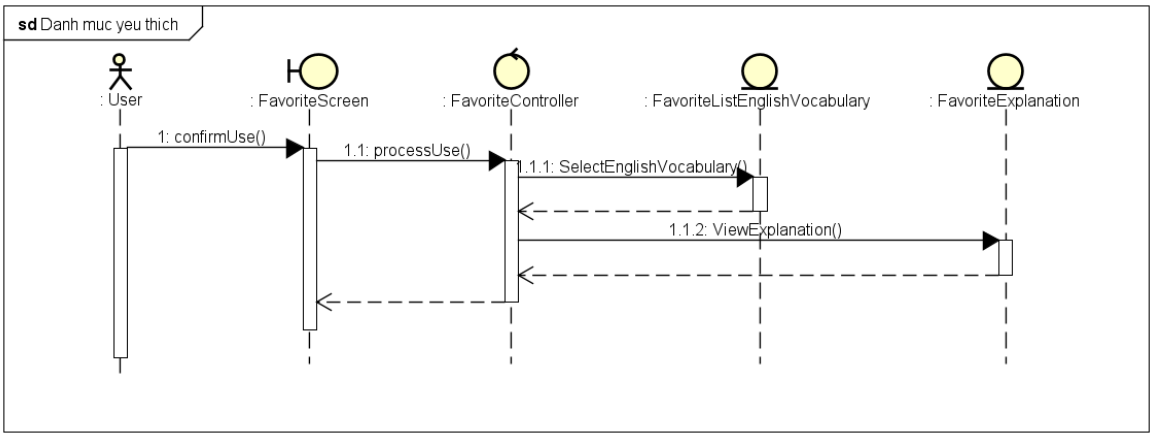
4.2.2 Thiết kế lớp



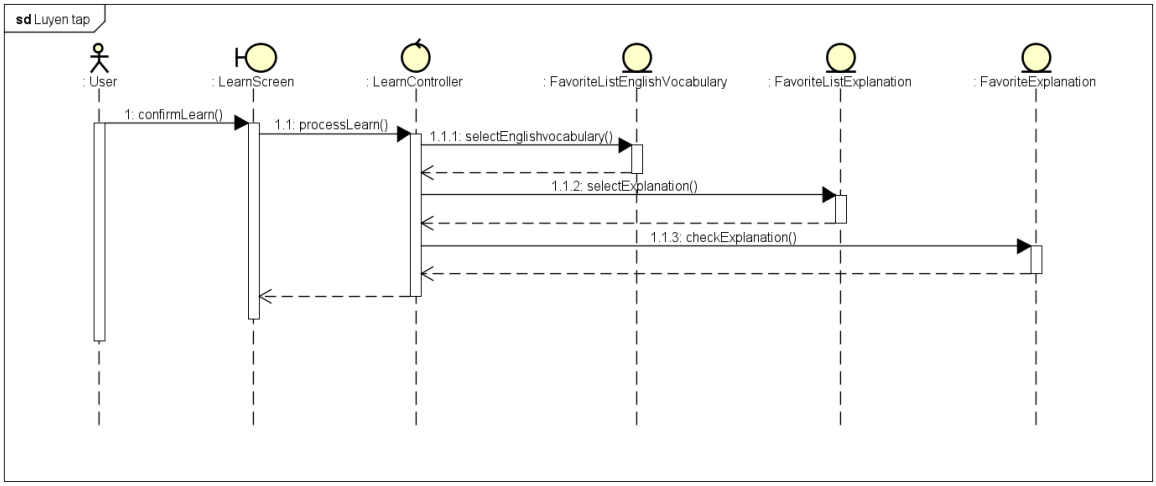
Hình 12: Biểu đồ lớp thiết kế



Hình 13: Biểu đồ trình tự cho use case "Tra từ điển"



Hình 14: Biểu đồ trình tự cho use case "Danh mục yêu thích"



Hình 15: Biểu đồ trình tự cho use case "Luyện tập"

4.2.3 Thiết kế cơ sở dữ liệu

Hình 16: Biểu đồ thực thể liên kết (E-R diagram)

Cơ sở dữ liệu có một bảng duy nhất là Dictionary và gồm có 2 cột là :

- English vocabulary (từ vựng tiếng Anh.)
- Explanation (Cột này có thể bao gồm phiên âm, loại từ, nghĩa tiếng Việt của từ vựng, một số cụm từ hay được sử dụng với từ vựng đó + nghĩa tiếng Việt của cụm từ.)

Dictionary

#	PK	FK	Column Name	Data Type	Mandatory
1.	X		English vocabulary	String	Yes
2.			Explanation	String	Yes

4.3 Xây dựng ứng dụng

4.3.1 Thư viện và công cụ sử dụng

Bảng 9: Danh sách thư viện và công cụ sử dụng

Mục đích	Công cụ	Địa chỉ URL
IDE lập trình	Visual Studio 2017 64 bit	https://visualstudio.microsoft.com/
Ngôn ngữ lập trình	C#	
Thiết kế giao diện	Windows Form	
Phát âm từ tiếng Anh	Thư viện SpeechLib	

4.3.2 Kết quả đạt được

- Ứng dụng đã hoàn thành các yêu cầu được đặt ra.
- Chương trình không cần người dùng phải có kết nối internet khi sử dụng.
- Chương trình có giao diện đơn giản, dễ sử dụng.
- Chương trình từ điển có nguồn dữ liệu khá phong phú, có âm thanh và có nhiều tính năng.

Số dòng code	Số lớp	Số gói	Dung lượng toàn bộ mã nguồn	Dung lượng của sản phẩm đóng gói
931	10	3	44 MB	0.772 MB

4.3.3 Minh hoạ các chức năng chính

- **Chức năng tra từ điển:**

Hình 17: Hình minh họa chức năng tra từ điển

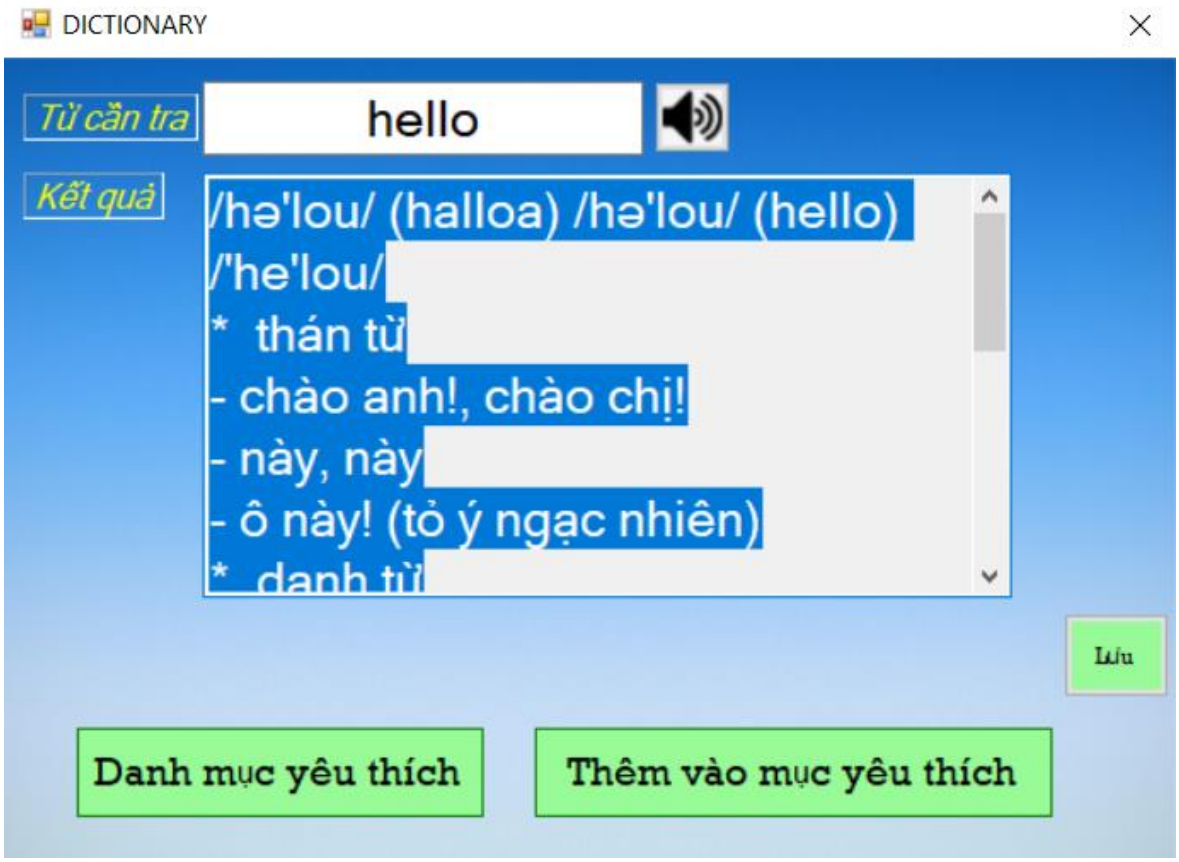
Người dùng nhập từ tiếng Anh cần tra sau đó hệ thống sẽ hiển thị kết quả tìm kiếm nếu từ đó tồn tại trong từ điển, nếu từ đó không tồn tại 1 thông báo "Không tìm thấy kết quả" sẽ được hiển thị.

- **Chức năng phát âm từ**

Người dùng nhập từ tiếng Anh cần nghe phát âm hoặc sau khi tra từ ở màn hình từ điển; hoặc sau khi chọn từ trong mục yêu thích ở màn hình danh mục yêu thích, người dung nhấn button có hình loa và hệ thống sẽ phát âm từ mà người muốn nghe phát âm.

Hình 18: Hình minh họa button phát âm từ

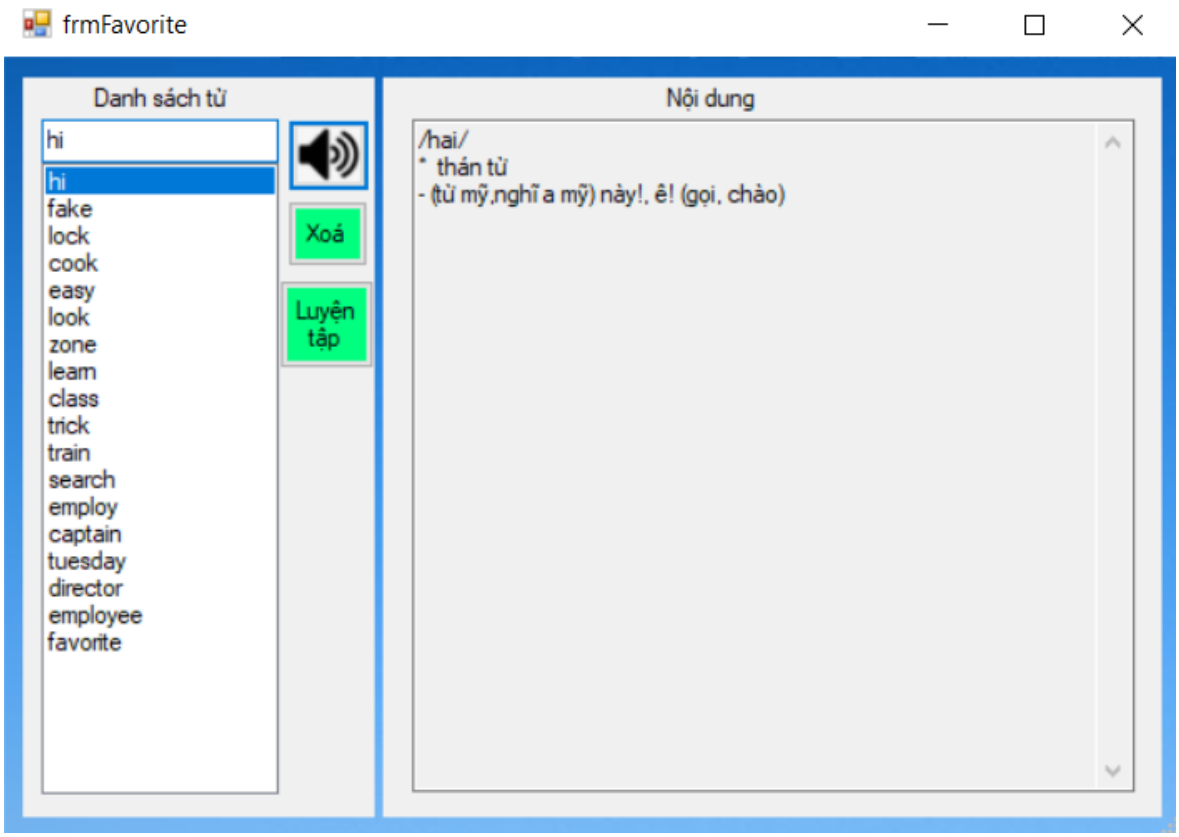
- **Chức năng thêm từ vào mục yêu thích:**



Hình 19: Hình minh họa chức năng thêm từ vào mục yêu thích

Sau khi tra từ điển, người dùng muốn thêm từ vừa tra được vào mục yêu thích thì người dùng nhấn button “Thêm vào mục yêu thích”. Sau khi nhấn, button Lưu sẽ được hiển thị, người dung chọn button Lưu để thêm thành công từ vào mục yêu thích.

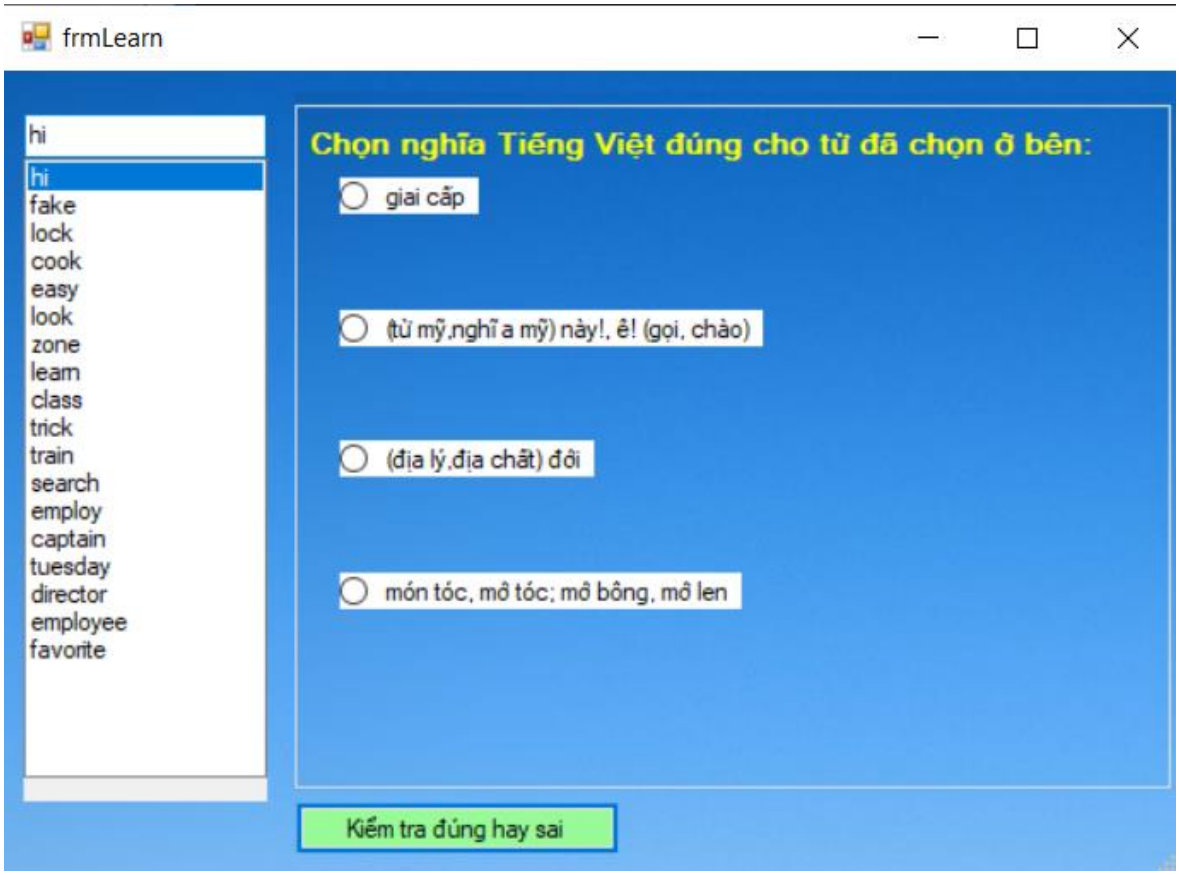
- **Chức năng hiển thị danh mục yêu thích:**



Hình 20: Hình minh họa chức năng hiển thị danh mục yêu thích

Người dùng nhấn button “Danh mục yêu thích”, sau đó danh sách các từ trong mục yêu thích sẽ được hiển thị. Người dùng nhấn chuột trái vào từ cần xem nội dung sau đó nội dung từ đó sẽ được hệ thống hiển thị ở cột “Nội dung”.

- **Chức năng luyện tập trắc nghiệm:**



Hình 21: Hình minh họa chức năng luyện tập trắc nghiệm

Từ giao diện mục yêu thích, người dung nhấn button “Luyện tập” sau đó danh sách các từ trong mục yêu thích sẽ được hiển thị. Người dung chọn từ mà mình muốn luyện tập, sau đó 4 phương án sẽ được hiển thị để người dung có thể chọn phương án mà người dung nghĩ là nghĩa Tiếng Việt của từ tiếng Anh mà mình đã chọn. Sau đó nhấn button “Kiểm tra đáp án” để biết mình làm đúng hay sai. Sau đó có thể chọn từ khác để tiếp tục luyện tập.

4.4 Kiểm thử

Kỹ thuật kiểm thử: Kiểm thử tĩnh (Static Testing):

+ Kiểm thử dữ liệu đầu vào cho chức năng tra từ điển:

- Dữ liệu đầu vào khác null: dữ liệu đầu vào là null - Hệ thống sẽ thông báo “Xin mời nhập dữ liệu”.
- Dữ liệu đầu vào không chứa kí tự đặc biệt: Dữ liệu đầu vào chứa kí tự đặc biệt - Hệ thống sẽ thông báo “Không tìm thấy kết quả”.

+ Kiểm thử dữ liệu đầu vào cho chứ năng kiểm tra đáp án của phần luyện tập:

- Yêu cầu người dung đã chọn đáp án của mình: người dùng chưa chọn đáp án - Hệ thống sẽ thông báo “Bạn chưa chọn đáp án của mình”.

+ Kiểm thử chức năng luyện tập với các từ có trong mục yêu thích:

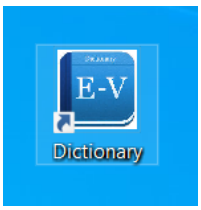
- Mục yêu thích phải có 5 từ trở lên: mục yêu thích ít hơn 5 từ - Hệ thống sẽ thông báo “Mục yêu thích cần có 5 từ trở lên”.

4.5 Triển khai

- Với một chiếc PC có kết nối Internet thì người dùng có thể truy cập vào địa chỉ được cung cấp để tải file đóng gói của sản phẩm về.

Hình 22: Hình minh họa file đóng gói sản phẩm

- Sau đó tiến hành mở và một vài tùy chọn nữa để cài đặt.
- Sau khi cài đặt xong phần mềm sẽ có dạng:



Hình 23: Hình minh họa sản phẩm sau khi cài đặt trên PC

- Sau đó tiến hành kiểm tra xem ứng dụng có hoạt động tốt hay không.

- Qua quá trình chạy thử trên PC thấy rằng phần mềm hoạt động tốt, không mắc lỗi nào đối với 5 chức năng chính của chương trình đó là tra từ Anh – Việt, phát âm từ, thêm từ vào mục yêu thích, xem mục yêu thích và luyện tập với các từ trong mục yêu thích.

Chương 5 Các giải pháp và đóng góp nổi bật

5.1 Thuật toán làm từ điển

- Đầu tiên là làm từ điển dùng cây nhị phân, một thuật toán từ điển khá nhanh, tuy nhiên cây sẽ bị mất đối xứng trong quá trình thêm, xóa; code hơi phức tạp và dễ lỗi. Một số cây thường được sử dụng để làm từ điển : cây AVL, cây cờ bạc (cây đỏ đen).
- Tiếp theo là xài cơ sở dữ liệu access, xml, sqlite … nếu dung nó thì rất dễ code, nhiều tính năng nhưng cần tự tạo dữ liệu từ điển hoặc tìm được dữ liệu sẵn có là file access, xml hay sqlite.
- Hoặc có thể dùng tìm kiếm nhị phân, thuật toán tìm kiếm khá nhanh và dễ sử dụng.

Cuối cùng là thuật toán mà em đã chọn: dùng bảng băm (hashtable) để làm từ điển, đặc điểm là từ từ khóa cần tìm dung mã băm để nhảy trực tiếp đến danh sách chứa vị trí nghĩa, nên tốc độ tìm kiếm là rất nhanh. Tốc độ phụ thuộc vào hàm băm (hashfunction) và dữ liệu nhập vào. Ở bài này em chọn hàm băm là tổng giá trị các chữ cái trong key là kiểu char được ép kiểu về dạng int. Sẽ chỉ có một vài phần tử có cùng giá trị băm được tính như trên, điều này làm cho việc tìm kiếm rất hiệu quả.

```
private int HashFunction(string key)
{
    int k = 0;
    for (int i = 0; i < key.Length; i++)
        k = k + key[i] - 48;
    return k % N;
}
```

Hình 24: Hình ảnh mô tả code của hàm băm

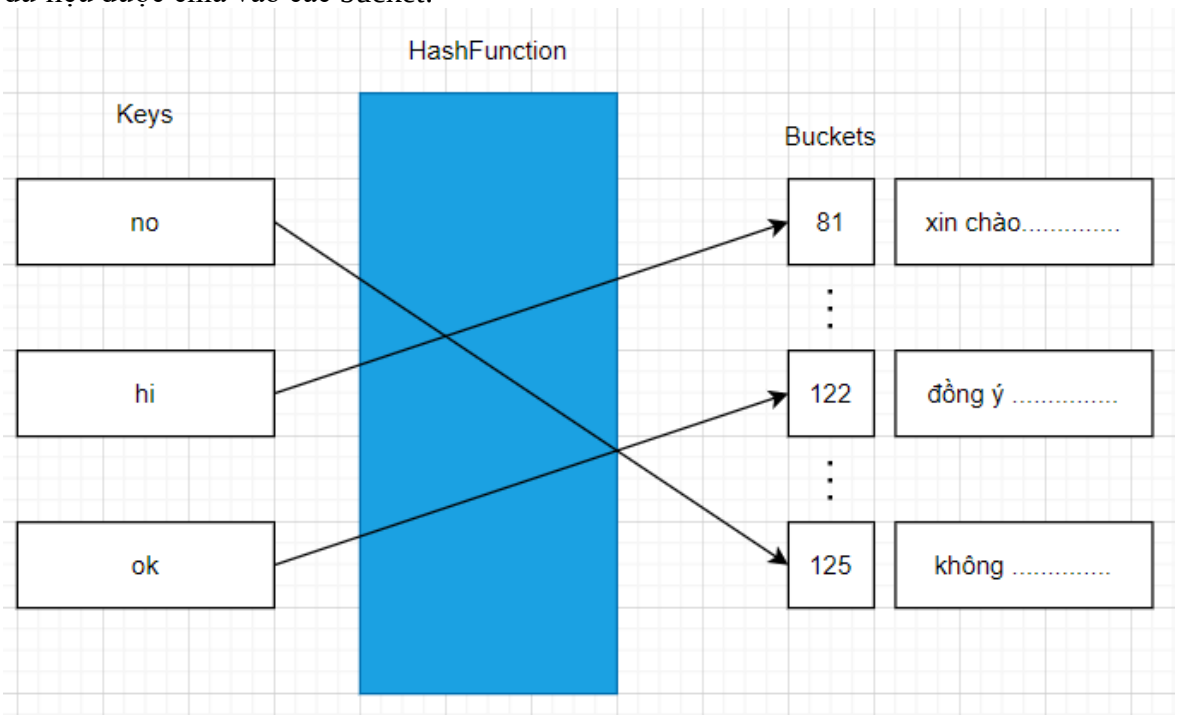
5.2 Bảng băm (Hash Table)

5.2.1 Tư tưởng

Bảng băm là một CTDL thường được sử dụng như một từ điển: mỗi phần tử trong bảng băm là một cặp (khóa, giá trị). Nếu so sánh với mảng, khóa được xem như chỉ số của

mảng, còn giá trị giống như giá trị mà ta lưu tại chỉ số tương ứng. Bảng băm không như các loại từ điển thông thường - ta có thể tìm được giá trị thông qua khóa của nó.

Không may, không phải tất cả các kiểu dữ liệu đều có thể sắp xếp vào một từ điển đơn giản. Đây chính là lúc mà quá trình băm (hash) ra đời. Hash là quá trình khởi tạo một giá trị khóa (thường là 32 bit hoặc 64 bit) từ một phần dữ liệu. Nó có thể là nn bit đầu tiên của dữ liệu, nn bit cuối cùng, giá trị mod cho một số nguyên tố nào đó. Dựa theo giá trị hash, dữ liệu được chia vào các bucket:



Hình 25: Hình minh họa

Giải thích hình minh họa:

- Ta cần lưu nghĩa của 3 từ tiếng Anh:
  - no: không …
  - hi: xin chào …
  - ok: đồng ý…
- Giá trị Hash của 3 từ này lần lượt là: 81, 122 và 125.
- Sau khi tính được giá trị Hash của 3 người, ta lưu vào các bucket tương ứng là 81, 122 và 125.

Nếu các kết quả của hàm hash được phân bố đều, các bucket sẽ có kích thước xấp xỉ nhau. Giả sử số bucket đủ lớn, mỗi buckets sẽ chỉ có một vài phần tử trong chúng. Điều này làm cho việc tìm kiếm rất hiệu quả.

5.2.2 Độ phức tạp

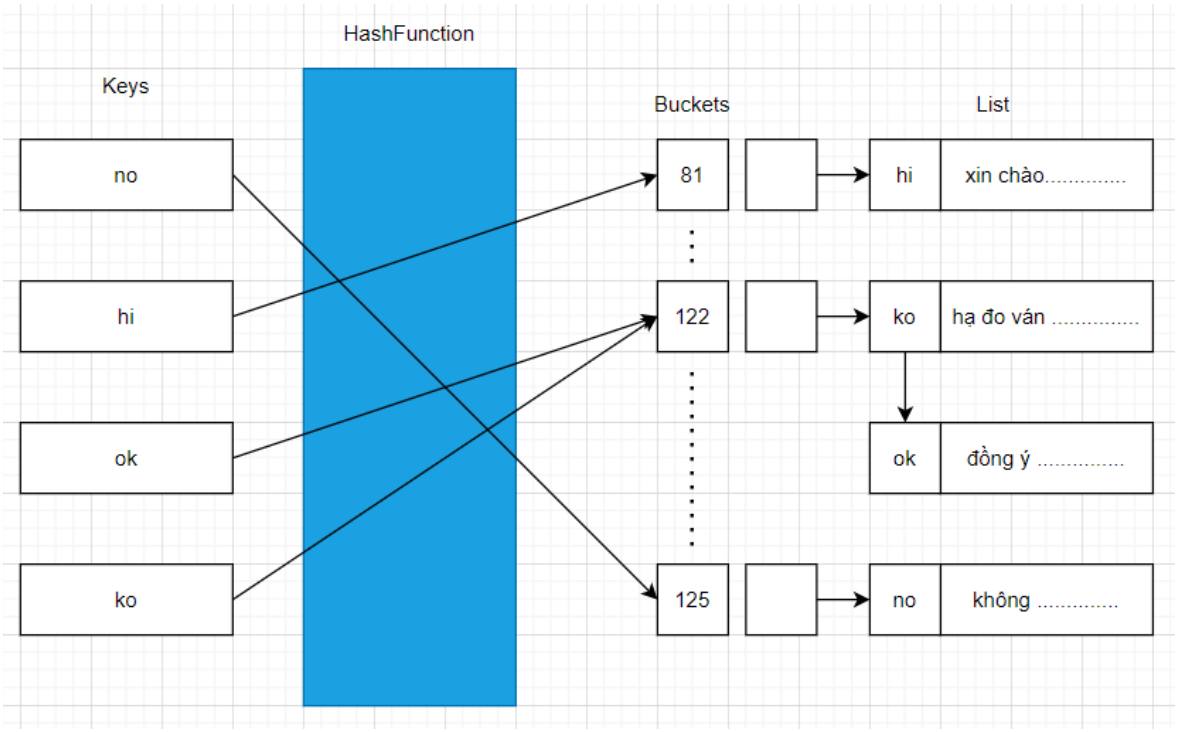
Gọi:

- n là số phần tử ta cần lưu trong Hash table
- k là số bucket

Giá trị n/k được gọi là load factor. Khi load factor nhỏ (xấp xỉ 1), và giá trị của hàm Hash phân bố đều, độ phức tạp của các thao tác trên Hash table là O(1).

5.2.3 Hash collision

Trường hợp một hash bucket chứa nhiều hơn một giá trị ta gọi đó là **Hash collision** (va chạm). Việc xử lý hash collision rất quan trọng đối với độ hiệu quả của bảng băm. Một trong những phương pháp đơn giản nhất là cài đặt các danh sách liên kết ở các bucket. Kĩ thuật này được gọi là **Separate chaining**:



Hình 26: Hình minh họa

Giải thích hình minh họa:

- Mỗi bucket là 1 danh sách liên kết
- ok và ko cùng có giá trị hàm hash là 122, nên ở bucket 122, ta có 1 danh sách liên kết chứa 2 phần tử.

5.2.4 Cài đặt

Dưới đây là HashTable đã được em cài đặt trong đồ án này:

- Class: MyHashTable

```
using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.Text;
using System.Threading.Tasks;
using System.IO;
using System.Windows.Forms;
namespace Dictionary.Business
{
    class MyHashTable
    {
        //Properties
        public static int N = Convert.ToInt32(SIZE.N);
        public LINKEDLIST[] HT = new LINKEDLIST[N];
        //Method
        //Hàm khởi tạo mảng băm
        public void InitHashTable()
        {
            for (int i = 0; i < HT.Count();i++ )
            {
                HT[i] = new LINKEDLIST();
                HT[i].InitList();
            }
        }
        //Hàm khởi tạo một từ điển
        public Node CreateNode(string key, string value)
        {
            Node p = new Node();
            p.key = key;
            p.value = value;
            p.pNext = null;
            return p;
        }
        //Hàm băm
        private int HashFunction(string key)
        {
            int k = 0;
            for (int i = 0; i < key.Length; i++)
                k = k + key[i] - 48;
            return k % N;
        }
        //Hàm thêm một key-value vào mảng băm
        public void AddKey(string key, string value)
        {
            int address = HashFunction(key);
            Node p = CreateNode(key, value);
            HT[address].AddLast(p);
        }
        //Hàm tìm một value dựa theo key
        public Node SearchKey(string key)
        {
            int address = HashFunction(key);
            return HT[address].SearchNode(key);
        }
        //Hàm remove key
        public void RemoveKey(string key)
        {
            int address = HashFunction(key);
            Node q = SearchKey(key);
            HT[address].RemoveNode(q);
        }
    }
}
```

- Class: LinkedList

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Dictionary.Business
{
    class LINKEDLIST
    {
        //Properties
        public Node pHead;
        public Node pTail;
        //Method
        public void InitList()
        {
            pHead = pTail = null;
        }
        public bool IsEmptyList()
        {
            if (pHead == null)
                return true;
            return false;
        }
        public void AddFirst(Node p)
        {
            if (pHead == pTail)
                pHead = pTail = p;
            else
            {
                p.pNext = pHead;
                pHead = p;
            }
        }
        public void AddLast(Node p)
        {
            if (pHead == null)
                pHead = pTail = p;
            else
            {
                pTail.pNext = p;
                pTail = p;
            }
        }
        public void RemoveFist()
        {
            if (!IsEmptyList())
            {
                Node p = pHead;
                pHead = pHead.pNext;

            }
        }
        public void RemoveLast()
        {
            if (!IsEmptyList())
            {
                if (pHead == pTail)
                    pHead = pTail = null;
```

```
                else
                {
                    Node p = pHead;
                    for (; p.pNext != pTail; p = p.pNext) ;
                    p.pNext = null;
                    pTail = p;
                }

            }
        }
        public void RemoveAfter(Node q)
        {
            Node temp = q.pNext;
            q.pNext = temp.pNext;
            if (q.pNext == null)
                pTail = q;
        }
        public void RemoveNode(Node q)
        {
            if (!IsEmptyList())
            {
                if (pHead == q)
                    RemoveFist();
                else if (pTail == q)
                    RemoveLast();
                else
                {
                    Node p = pHead;
                    for (; p.pNext != q; p = p.pNext) ;
                    RemoveAfter(p);
                }
            }
        }
        public Node SearchNode(string value)
        {
            for (Node p = pHead; p != null; p = p.pNext)
                if (Compare(p.key,value))
                    return p;
            return null;
        }
        private bool Compare(string x, string y)
        {
            x = x.Trim();
            y = y.Trim();
            return String.Compare(x, y, true) == 0;
        }
    }
}
```

- Class: Node

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Dictionary.Business
{
    class Node
    {
        public string key;
```

```
        public string value;
        public Node pNext;
    }
}
```

Chương 6 Kết luận và hướng phát triển

6.1 Kết luận

Sau quá trình nghiên cứu, tìm hiểu và thực hiện em đã xây dựng hành công ứng dụng từ điển Anh Việt hỗ trợ cho việc học tiếng Anh.

- ❖ Kết quả đạt được:
  - Ứng dụng đã hoàn thành các yêu cầu được đặt ra.
  - Chương trình không cần người dung phải có kết nối internet khi sử dụng.
  - Chương trình có giao diện đơn giản, dễ sử dụng.
- ❖ Hạn chế
  - Dữ liệu từ điển chưa đủ lớn.
  - Chưa có nhiều chức năng.
  - Chưa có thống kê kết quả trong quá trình sử dụng

6.2 Hướng phát triển

- Đề tài có thể phát triển trên nền tảng khác như Android, iOS, Windows Phone.
- Thêm các tính năng khác như: Dịch văn bản, gợi ý từ cần tra, lịch sử tra, các bài thi hỗ trợ học từ vựng, …
- Mở rộng và nâng cấp dữ liệu để hỗ trợ tra các cụm từ hoặc có thể là một câu.
- Thêm một số ngôn ngữ phổ biến khác như: tiếng Trung, tiếng Nhật, tiếng Hàn, …
- Xây dựng giao diện trực quan và đẹp mắt hơn.
- Áp dụng thêm nhiều thuật toán thông minh vào ứng dụng.
- Thống kê kết quả khi người dung sử dụng phần mềm.
- Tra từ qua clipboard: Đây là một tiện ích tra từ trong ứng dụng khác, chỉ việc bôi đen text và gõ Ctrl+C để tra đoạn text đó.

Tài liệu tham khảo

[1] Phạm Duy Anh, Ebook hướng dẫn lập trình từ điển, 02-01-2014, https://www.slideshare.net/duyanhphamkiller/ebook-huong-dan-lam-tu-dien-ti-123docvn

[2] ngotrung.se, Lập trình Windows Form cơ bản, 17-06-2020, https://codelearn.io/sharing/lap-trinh-windows-form-co-ban

[3] HashTable trong C#, cập nhật lần cuối 7-2020, https://www.howkteam.vn/course/khoa-hoc-lap-trinh-c-nang-cao/hashtable-trong-c-1560