**TRƯỜNG ĐẠI HỌC ĐỒNG THÁP**

**KHOA SƯ PHẠM TOÁN – TIN**



**NGÔN NGỮ LẬP TRÌNH PYTHON**

**CHƯƠNG TRÌNH QUẢN LÝ RAU CỦ**

**VÀ CHƯƠNG TRÌNH GAME CƠ BẢN BẰNG PYTHON**

NGHÀNH: KHOA HỌC MÁY TÍNH

LỚP: ĐHCNTT20A

**NGUYỄN TRƯỜNG 0020411284**

**NGUYỄN THIỆN CHÍ 0020410055**

**NGUYỄN HOÀNG NAM 0020410419**

**Đồng Tháp, tháng 1 năm 2023**

**TRƯỜNG ĐẠI HỌC ĐỒNG THÁP**

**KHOA SƯ PHẠM TOÁN – TIN**



**NGÔN NGỮ LẬP TRÌNH PYTHON**

**CHƯƠNG TRÌNH QUẢN LÝ RAU CỦ**

**VÀ CHƯƠNG TRÌNH GAME CƠ BẢN BẰNG PYTHON**

NGHÀNH: KHOA HỌC MÁY TÍNH

LỚP: ĐHCNTT20A

**GIẢNG VIÊN: TS ĐẶNG QUỐC BẢO**

# MỤC LỤC

[MỤC LỤC i](#_Toc124152481)

[DANH MỤC HÌNH 2](#_Toc124152482)

[PHẦN 1: GIỚI THIỆU 3](#_Toc124152483)

[1.1. Giới thiệu đề tài 3](#_Toc124152484)

[1.2. Nội dụng thực hiện 3](#_Toc124152485)

[1.3. Mục tiêu của đề tài 3](#_Toc124152486)

[1.4. Phân công công việc thực hiện 3](#_Toc124152487)

[PHẦN 2: CƠ SỞ LÝ THUYẾT VÀ DỮ LIỆU 4](#_Toc124152488)

[2.1. Ngôn ngữ lập trình python 4](#_Toc124152489)

[2.1.1. Giới thiệu về python 4](#_Toc124152490)

[2.1.2. Một số thư viện trong python 4](#_Toc124152491)

[2.2. Hệ quản trị cơ sở dữ liệu MySQL 5](#_Toc124152492)

[PHẦN 3: XÂY DỰNG ỨNG DỤNG 8](#_Toc124152493)

[3.1. Giao diện đăng nhập 8](#_Toc124152494)

[3.2. Giao diện khi nhấn nút giải trí 8](#_Toc124152495)

[3.3. Giao diện quản lý chương trình rau củ 11](#_Toc124152496)

[3.4. Giao diện kết nối cơ sở dữ liệu 12](#_Toc124152497)

[3.5. Giao diện khi thêm rau củ mới 14](#_Toc124152498)

[3.6. Giao diện tìm thêm rau củ 18](#_Toc124152499)

[3.7. Giao diện xóa 20](#_Toc124152500)

[3.8. Giao diện xem danh sách rau củ 21](#_Toc124152501)

[3.9. Giao diện xuất danh sách rau củ ra file exel 22](#_Toc124152502)

[3.10. Giao diện cập nhật danh sách rau củ 25](#_Toc124152503)

[PHẦN 4: KẾT LUẬN 27](#_Toc124152504)

[4.1. Kết quả đạt được 27](#_Toc124152505)

[4.2. Hạn chế 27](#_Toc124152506)

[Hướng phát triển 27](#_Toc124152507)

[TÀI LIỆU THAM KHẢO 28](#_Toc124152508)

# DANH MỤC HÌNH

[Hinh 1 Giao diện đăng nhập 10](file:///D:\HK%201%20NAM%20BA\PYTHON\NHOM_1_BTL_PYTHON_QUAN_LY_RAU_CU.docx#_Toc124119064)

[Hinh 2 Giao diện giải trí 11](file:///D:\HK%201%20NAM%20BA\PYTHON\NHOM_1_BTL_PYTHON_QUAN_LY_RAU_CU.docx#_Toc124119065)

[Hinh 3 Giao diện flappy bird 11](file:///D:\HK%201%20NAM%20BA\PYTHON\NHOM_1_BTL_PYTHON_QUAN_LY_RAU_CU.docx#_Toc124119066)

[Hinh 4 Giao diện game xếp hình 12](file:///D:\HK%201%20NAM%20BA\PYTHON\NHOM_1_BTL_PYTHON_QUAN_LY_RAU_CU.docx#_Toc124119067)

[Hinh 5 Giao diện quản lý chính 13](file:///D:\HK%201%20NAM%20BA\PYTHON\NHOM_1_BTL_PYTHON_QUAN_LY_RAU_CU.docx#_Toc124119068)

[Hinh 6 Giao diện kết nối cơ sở dữ liệu 14](file:///D:\HK%201%20NAM%20BA\PYTHON\NHOM_1_BTL_PYTHON_QUAN_LY_RAU_CU.docx#_Toc124119069)

[Hinh 7 Giao diện thêm mới 16](file:///D:\HK%201%20NAM%20BA\PYTHON\NHOM_1_BTL_PYTHON_QUAN_LY_RAU_CU.docx#_Toc124119070)

[Hinh 8 Giao diện thêm lỗi 16](file:///D:\HK%201%20NAM%20BA\PYTHON\NHOM_1_BTL_PYTHON_QUAN_LY_RAU_CU.docx#_Toc124119071)

[Hinh 9 Giao diện thêm thành công 17](#_Toc124119072)

[Hinh 10 Giao diện sau khi thêm sản phẩm thành công 17](file:///D:\HK%201%20NAM%20BA\PYTHON\NHOM_1_BTL_PYTHON_QUAN_LY_RAU_CU.docx#_Toc124119073)

[Hinh 11 Giao diện tìm kiếm 19](file:///D:\HK%201%20NAM%20BA\PYTHON\NHOM_1_BTL_PYTHON_QUAN_LY_RAU_CU.docx#_Toc124119074)

[Hinh 12 Giao diện sau khi xóa 21](file:///D:\HK%201%20NAM%20BA\PYTHON\NHOM_1_BTL_PYTHON_QUAN_LY_RAU_CU.docx#_Toc124119075)

[Hinh 13 Giao diện trước khi xóa 22](file:///D:\HK%201%20NAM%20BA\PYTHON\NHOM_1_BTL_PYTHON_QUAN_LY_RAU_CU.docx#_Toc124119076)

[Hinh 14 Giao diện xem danh sách rau củ 23](file:///D:\HK%201%20NAM%20BA\PYTHON\NHOM_1_BTL_PYTHON_QUAN_LY_RAU_CU.docx#_Toc124119077)

[Hinh 15 Giao diện xuất file excel 24](file:///D:\HK%201%20NAM%20BA\PYTHON\NHOM_1_BTL_PYTHON_QUAN_LY_RAU_CU.docx#_Toc124119078)

[Hinh 16 Giao diện xuất file excel 24](file:///D:\HK%201%20NAM%20BA\PYTHON\NHOM_1_BTL_PYTHON_QUAN_LY_RAU_CU.docx#_Toc124119079)

[Hinh 17 File excel đã xuất 25](file:///D:\HK%201%20NAM%20BA\PYTHON\NHOM_1_BTL_PYTHON_QUAN_LY_RAU_CU.docx#_Toc124119080)

[Hinh 18 Giao diện cập nhật danh sách rau củ 25](file:///D:\HK%201%20NAM%20BA\PYTHON\NHOM_1_BTL_PYTHON_QUAN_LY_RAU_CU.docx#_Toc124119081)

# PHẦN 1: GIỚI THIỆU

## 1.1. Giới thiệu đề tài

Rau củ là thực phẩm không thể thiếu trong bữa ăn hằng ngày của mỗi người. Chính vì thế việc buôn bán rau củ ngày càng phát triển, nhu cầu mua hàng của con người ngày càng cao mà sức làm việc của con người có hạn, do đó rất cần sự hỗ trợ của công nghệ. Mặt khác, công nghệ phần mềm ngày càng phát triển nên việc viết một phần mềm quản lý rau củ là việc rất có khả thi.

Ngoài ra, để giúp nhân viên bán rau củ có thể giải trí thì việc tích hợp vào chương trình các game đơn là rất cần thiết. Đó cũng là lý do nhóm chọn đề tài “chương trình quản lý rau củ và chương trình game cơ bản bằng python”.

## 1.2. Nội dụng thực hiện

Nội dụng thực hiện đề tài:

* Xây dựng cơ sở dữ liệu trên MySql.
* Xây dựng các game đơn giản.
* Tạo giao diện quản lý và xử lý các nút lệnh.

## 1.3. Mục tiêu của đề tài

Mục tiêu khi thực hiện đề tài:

* Xây dựng được ứng dụng quản lý rau củ.
* Tạo được hai game đơn giản.
* Giao diện thân thiện.
* Cơ sở dữ liệu phù hợp.

## 1.4. Phân công công việc thực hiện

|  |  |
| --- | --- |
| **Nội dụng thực hiện** | **Người thực hiện** |
| Xây dựng cơ sở dữ liệu | Nguyễn Thiện Chí  Nguyễn Trường An  Nguyễn Hoàng Nam |
| Xây dựng giao diện và các nút lệnh | Nguyễn Thiện Chí  Nguyễn Trường An  Nguyễn Hoàng Nam |
| Thực hiện bài báo cáo | Nguyễn Thiện Chí  Nguyễn Trường An  Nguyễn Hoàng Nam |

# PHẦN 2: CƠ SỞ LÝ THUYẾT VÀ DỮ LIỆU

## 2.1. Ngôn ngữ lập trình python

### 2.1.1. Giới thiệu về python

Python là một ngôn ngữ lập trình thông dịch do Guido van Rossum tạo ra năm 1990. Python hoàn toàn tạo kiểu động và dùng cơ chế cấp phát bộ nhớ tự động do vậy nó tương tự như Perl, Ruby, Scheme, Smalltalk,… Python được phát triển trong một dự án mã nguồn mở, do tổ chức phi lợi nhuận Python Software Foundation quản lý.

Theo đánh giá của Eric S. Raymond, Python là ngôn ngữ có hình thức rất sáng sủa, cấu trúc rõ ràng, thuận tiện cho người mới học lập trình. Cấu trúc của Python còn cho phép người sử dụng viết mã lệnh với số lần gõ phím tối thiểu. Ban đầu, Python được phát triển để chạy trên nền Unix. Nhưng rồi theo thời gian, nó đã phát triển sang mọi hệ điều hành từ MS-DOS đến Mac OS, OS/2, Windows, Linux và các hệ điều hành khác thuộc họ Unix. Mặc dù sự phát triển của Python có sự đóng góp của rất nhiều cá nhân, nhưng Guido van Rossum vẫn được xem là tác giả chính của Python. Ông giữ vai trò chủ chốt trong việc định hướng phát triển của Python. [1]

### 2.1.2. Một số thư viện trong python

#### 2.1.2.1. Thư viện time

Python có một mô-đun được đặt tên là time để xử lý các tác vụ liên quan đến thời gian. Để sử dụng các chức năng được xác định trong mô-đun, trước tiên chúng ta cần import mô-đun.[3]

#### 2.1.2.2. Thư viện time tkinter

Tkinter là thư viện GUI tiêu chuẩn cho Python. Tkinter trong Python cung cấp một cách nhanh chóng và dễ dàng để tạo các ứng dụng GUI. Tkinter cung cấp giao diện hướng đối tượng cho bộ công cụ Tk GUI.[2]

Sau đây là các bước để tạo một ứng dụng Tkinter:

* Import Tkinter module.
* Tạo cửa sổ chính của ứng dụng GUI.
* Thêm một hoặc nhiều widget nói trên vào ứng dụng GUI.
* Gọi vòng lặp sự kiện chính để các hành động có thể diễn ra trên màn hình máy tính của người dùng.

#### 2.1.2.3. Thư viện time pymysql

Một trong số những ứng dụng của ngôn ngữ Python trong thực tế đó là việc kết nối với cơ sở dữ liệu và thao tác xử lý với dữ liệu được lưu trữ trong một số hệ quản trị dữ liệu. Trong một số hệ quản trị cơ sở dữ liệu đó, phải kể đến MySQL – một trong những hệ quản trị cơ sở dữ liệu phổ biến và dễ dàng thao tác với Python.

MySQL là hệ quản trị cơ sở dữ liệu mã nguồn mở. Hiện tại MySQL đang là hệ quản trị cơ sở dữ liệu phổ biến nhất thế giới và được các lập trình viên ưa chuộng trong quá trình phát triển ứng dụng. MySQL có nhiều ưu điểm như: tốc độ cao, ổn định và dễ sử dụng, có tính khả chuyển, hoạt động trên nhiều hệ điều hành cung cấp một hệ thống lớn các hàm tiện ích rất mạnh.

#### 2.1.2.4. Thư viện time pandas

Thư viện pandas trong python là một thư viện mã nguồn mở, hỗ trợ đắc lực trong thao tác dữ liệu. Đây cũng là bộ công cụ phân tích và xử lý dữ liệu mạnh mẽ của ngôn ngữ lập trình python. Thư viện này được sử dụng rộng rãi trong cả nghiên cứu lẫn phát triển các ứng dụng về khoa học dữ liệu. Thư viện này sử dụng một cấu trúc dữ liệu riêng là Dataframe. Pandas cung cấp rất nhiều chức năng xử lý và làm việc trên cấu trúc dữ liệu này. Chính sự linh hoạt và hiệu quả đã khiến cho pandas được sử dụng rộng rãi.[4]

#### 2.1.2.5. Thư viện time pygame

Pygame là một bộ mô-đun Python đa nền tảng được thiết kế để viết trò chơi điện tử. Nó bao gồm đồ họa máy tính và thư viện âm thanh được thiết kế để sử dụng với ngôn ngữ lập trình Python.

Pygame sử dụng thư viện Simple DirectMedia Layer (SDL), với mục đích cho phép phát triển trò chơi máy tính trong thời gian thực mà không cần cơ chế bậc thấp của ngôn ngữ lập trình C và các dẫn xuất của nó. Điều này dựa trên giả định rằng các chức năng đắt tiền nhất bên trong trò chơi có thể được trừu tượng hóa khỏi logic trò chơi, do đó có thể sử dụng ngôn ngữ lập trình bậc cao, chẳng hạn như Python, để cấu trúc trò chơi.

Các tính năng khác mà SDL không có bao gồm toán học vectơ, phát hiện va chạm, quản lý độ họa 2d, hỗ trợ MIDI, camera, thao tác mảng pixel, chuyển đổi, lọc, hỗ trợ phông chữ freetype nâng cao và vẽ.

Các ứng dụng sử dụng pygame có thể chạy trên điện thoại và máy tính bảng Android với việc sử dụng Bộ phụ pygame cho Android (pgs4a). Âm thanh, rung, bàn phím và gia tốc kế được hỗ trợ trên Android.

## 2.2. Hệ quản trị cơ sở dữ liệu MySQL

MySQL là một hệ thống quản lý cơ sở dữ liệu quan hệ mã nguồn mở (RDBMS) dựa trên ngôn ngữ truy vấn có cấu trúc ( SQL) được phát triển, phân phối và hỗ trợ bởi tập đoàn Oracle. MySQL chạy trên hầu hết tất cả các nền tảng, bao gồm cả Linux , UNIX và Windows. MySQL thường được kết hợp với các ứng dụng web.

SQL là ngôn ngữ phổ biến nhất để thêm, truy cập và quản lý nội dung trong cơ sở dữ liệu. Nó được chú ý nhất vì khả năng xử lý nhanh, độ tin cậy đã được chứng minh, dễ sử dụng và linh hoạt. MySQL là một phần thiết yếu của hầu hết mọi ứng dụng PHP mã nguồn mở. Các ví dụ điển hình cho các tập lệnh dựa trên PHP và MySQL là WordPress, Joomla, Magento và Drupal. MySQL đang trở nên phổ biến vì nhiều lý do tốt:

+ MySQL được phát hành theo giấy phép nguồn mở. Vì vậy, bạn không phải trả tiền để sử dụng nó.

+ MySQL là một chương trình rất mạnh theo đúng nghĩa của nó. Nó xử lý một tập hợp lớn các chức năng của các gói cơ sở dữ liệu mạnh mẽ và đắt tiền nhất.

+ MySQL sử dụng một dạng chuẩn của ngôn ngữ dữ liệu SQL nổi tiếng.

+ MySQL hoạt động trên nhiều hệ điều hành và với nhiều ngôn ngữ bao gồm PHP, PERL, C, C ++, JAVA, v.v.

+ MySQL hoạt động rất nhanh và hoạt động tốt ngay cả với các tập dữ liệu lớn.

+ MySQL rất thân thiện với PHP, ngôn ngữ được đánh giá cao nhất để phát triển web.

+ MySQL hỗ trợ cơ sở dữ liệu lớn, lên tới 50 triệu hàng hoặc nhiều hơn trong một bảng. Giới hạn kích thước tệp mặc định cho một bảng là 4GB, nhưng bạn có thể tăng mức này (nếu hệ điều hành của bạn có thể xử lý nó) đến giới hạn lý thuyết là 8 triệu terabyte (TB).

+ MySQL là tùy biến. Giấy phép GPL mã nguồn mở cho phép các lập trình viên sửa đổi phần mềm MySQL để phù hợp với môi trường cụ thể của riêng họ.

**Cách thức hoạt động của MySQL**

MySQL dựa trên mô hình client-server. Cốt lõi của MySQL là máy chủ MySQL, xử lý tất cả các hướng dẫn cơ sở dữ liệu (hoặc các lệnh). Máy chủ MySQL có sẵn như là một chương trình riêng biệt để sử dụng trong môi trường mạng client-server và như một thư viện có thể được nhúng (hoặc liên kết) vào các ứng dụng riêng biệt.

MySQL hoạt động cùng với một số chương trình tiện ích hỗ trợ quản trị cơ sở dữ liệu MySQL. Các lệnh được gửi đến MySQLServer thông qua máy khách MySQL, được cài đặt trên máy tính.

MySQL ban đầu được phát triển để xử lý cơ sở dữ liệu lớn một cách nhanh chóng. Mặc dù MySQL thường chỉ được cài đặt trên một máy, nhưng nó có thể gửi cơ sở dữ liệu đến nhiều vị trí, vì người dùng có thể truy cập thông qua các giao diện máy khách MySQL khác nhau. Các giao diện này gửi các câu lệnh SQL đến máy chủ và sau đó hiển thị kết quả.

**Các tính năng cốt lõi của MySQL**

MySQL cho phép dữ liệu được lưu trữ và truy cập trên nhiều công cụ lưu trữ, bao gồm InnoDB, CSV và NDB. MySQL cũng có khả năng sao chép dữ liệu và phân vùng bảng để có hiệu suất và độ bền tốt hơn. Người dùng MySQL không bắt buộc phải học các lệnh mới; họ có thể truy cập dữ liệu của mình bằng các lệnh SQL tiêu chuẩn.

MySQL được viết bằng C và C++ và có thể truy cập và có sẵn trên hơn 20 nền tảng, bao gồm Mac, Windows, Linux và Unix. RDBMS hỗ trợ cơ sở dữ liệu lớn với hàng triệu bản ghi và hỗ trợ nhiều loại dữ liệu bao gồm các số nguyên có chữ ký hoặc không dấu có độ dài 1, 2, 3, 4 và 8 byte(s); FLOAT; DOUBLE; CHAR; VARCHAR; BINARY; VARBINARY; TEXT; BLOB; DATE; TIME; DATETIME; TIMESTAMP; YEAR; SET; ENUM; và các kiểu OpenGIS. Các loại chuỗi có độ dài cố định và biến đổi cũng được hỗ trợ.

Để bảo mật, MySQL sử dụng một đặc quyền truy cập và hệ thống mật khẩu được mã hóa cho phép xác minh dựa trên máy chủ. Các máy khách MySQL có thể kết nối với Máy chủ MySQL bằng một số giao thức, bao gồm cả giao thức TCP/IP trên bất kỳ nền tảng nào. MySQL cũng hỗ trợ một số chương trình máy khách và tiện ích, chương trình dòng lệnh và công cụ quản trị như MySQL Workbench.

# PHẦN 3: XÂY DỰNG ỨNG DỤNG

## 3.1. Giao diện đăng nhập

Hình 1 là giao diện đăng nhập khi nhập tài khoản python và mật khẩu là python chương trình sẽ đưa đến giao diện chính nếu nhấn vào nút đăng nhập. Nếu nhấn vào nút giải trí sẽ chuyển đến màn hình lựa chọn game.

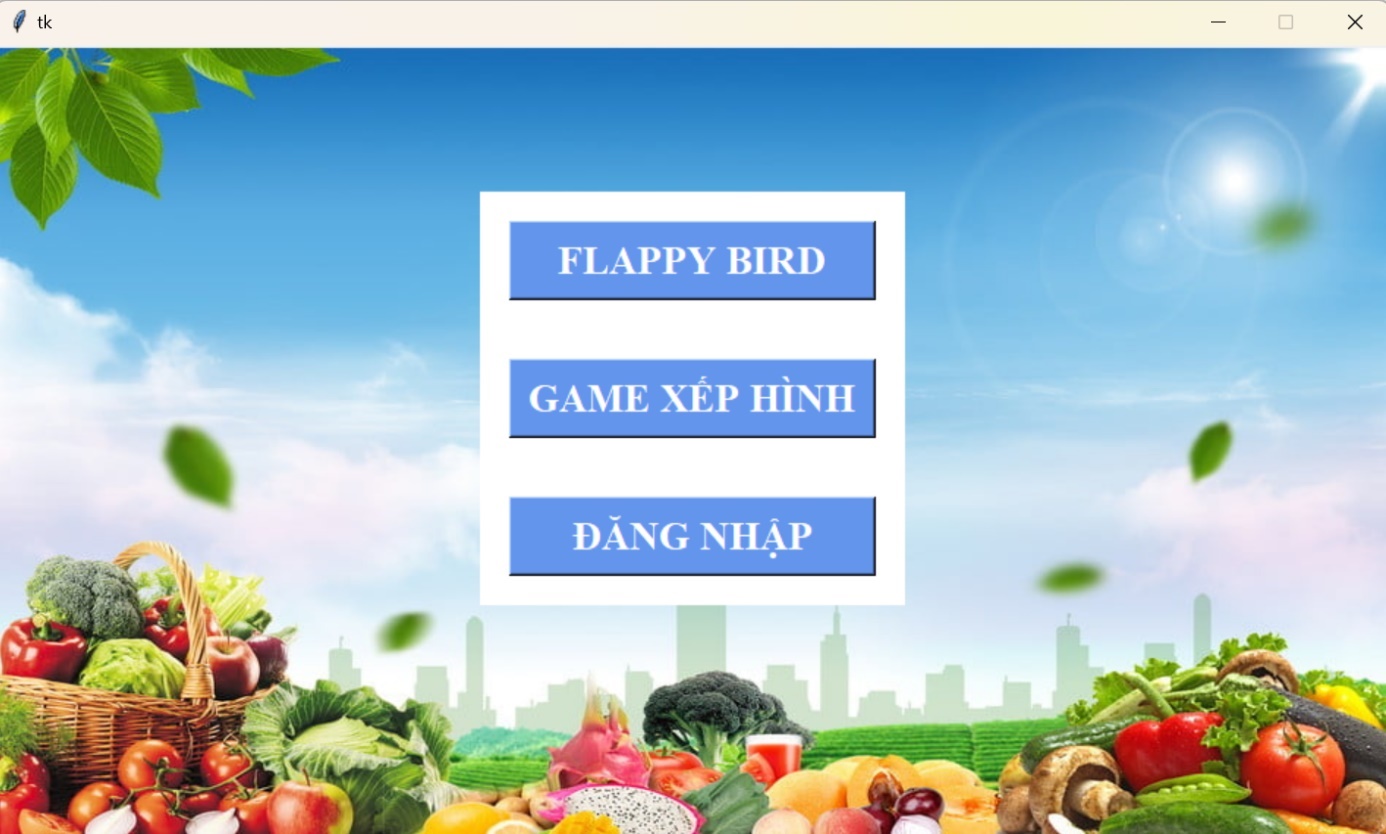
Hinh 1 Giao diện đăng nhập

Dưới đây là code cho nút đăng nhập:

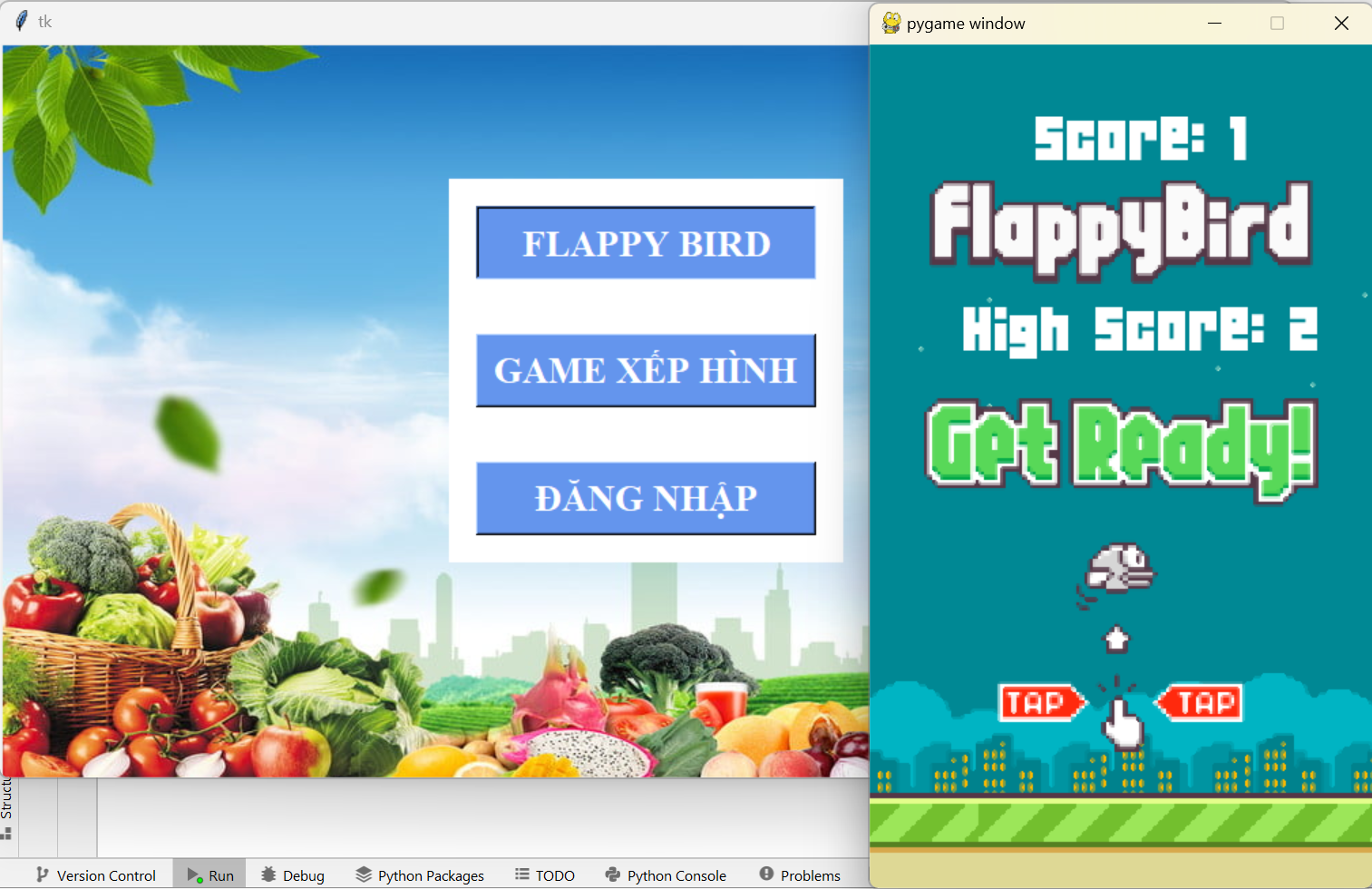
|  |
| --- |
| **def** login():  *#Tài khoản là python, mật khẩu python* **if** usernameEntry.get() == **'' and** passwordEntry.get() ==**''**:  messagebox.showerror(**'Lỗi đăng nhập'**,  **'Tài khoản hoặc mật khẩu sai chưa được nhập. Vui lòng kiểm tra lại!'**)  **elif** usernameEntry.get() == **'python' and** passwordEntry.get() == **'python'**:  messagebox.showinfo(**'Thông báo'**, **'Đăng nhập thành công!'**)  window.destroy()  **import** quanly\_raucu  **else**: messagebox.showerror(**'Lỗi đăng nhập'**,  **'Tài khoản hoặc mật khẩu sai. Vui lòng kiểm tra lại!'**) |

## 3.2. Giao diện khi nhấn nút giải trí

Hình 2 là giao diện đăng nhập khi nhấn vào nút giải trí. Ở giao diện này người dùng có thể lựa chọn game để giải trí hoặc trở về giao diện đăng nhập.

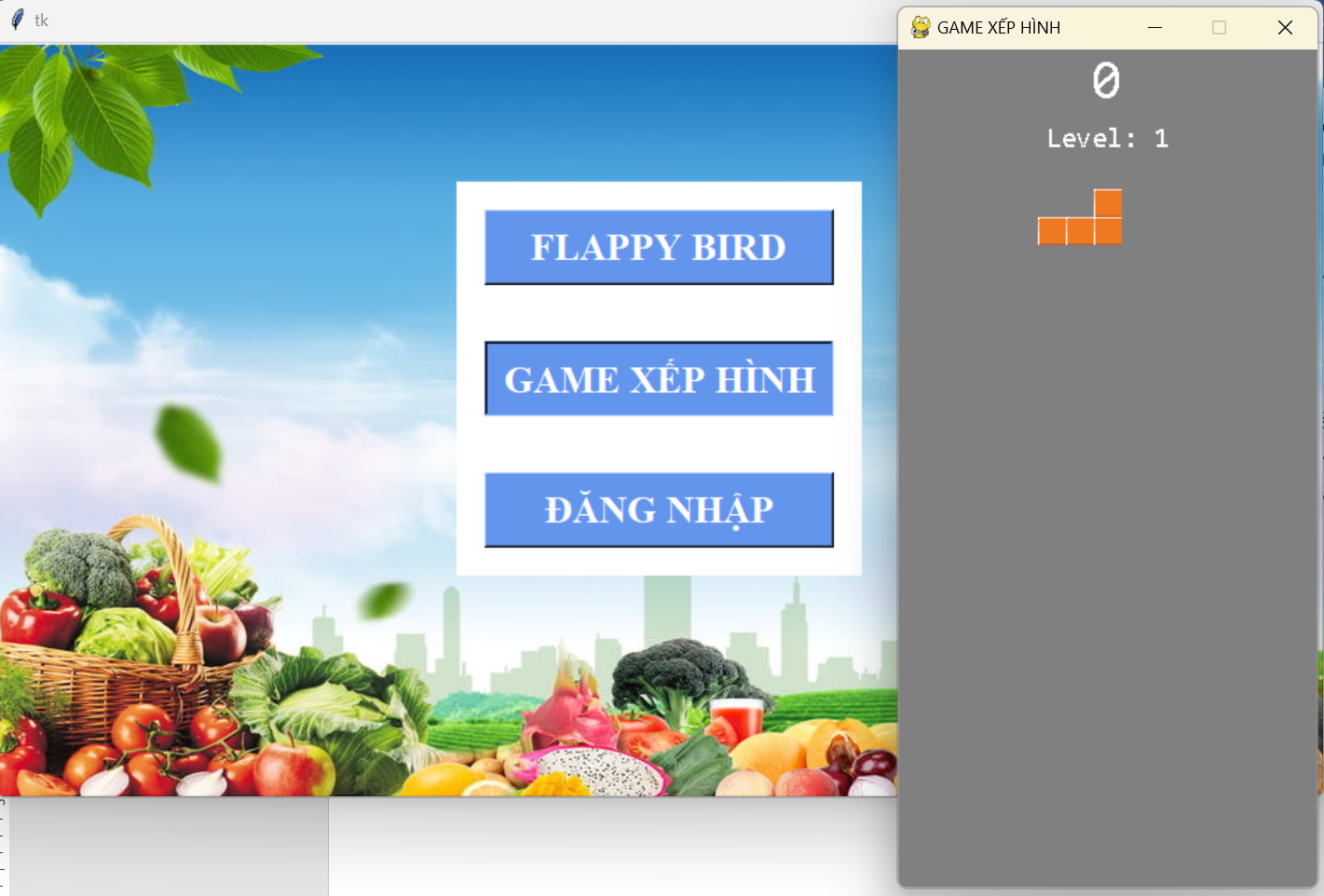


Hinh 2 Giao diện giải trí

Khi người dùng lựa chọn vào nút flappy bird chương trình sẽ hiện giao diện game cho người chơi như hình 3:

Hinh 3 Giao diện flappy bird

Ở giao diện game flappy bird người dùng nút space để bắt đầu trò chơi và cũng sử dụng nút space trên bằng phím để điều khiển chim tránh va chạm vào các ống trên đường đi. Nếu chim va chạm vào ống sẽ kết thúc trò chơi. Ngoài ra khi người dùng đều khiển chim rơi xuống đất trò chơi cũng kết thúc. Điểm số sẽ được cộng sau mỗi 1s và chương trình sẽ lưu điểm số cao nhất và hiển thị màn hình gameover cho người dùng lựa chọn.

Còn nếu người dùng lựa chọn vào nút game xếp hình chương trình sẽ chuyển người dùng đến giao diện game xếp hình như hình 4 dưới đây:

Hinh 4 Giao diện game xếp hình

Game sẽ ngẫu nhiên sinh ra các khối hình khác nhau. Nhiệm vụ của người chơi là sắp xếp chúng lại để lắp đầy một dòng, nếu đầy một dòng thì dòng đó sẽ biến mất và cộng điểm cho người chơi. Trong quá trình rơi xuống người dùng có thể dùng các phím mũi tên để điều khiển khối hình rơi theo ý của người chơi. Phím mũi tên xuống sẽ giúp cho khối hình rơi xuống nhanh hơn. Phím mũi tên sang trái sẽ đưa khối hình sang trái. Phím mũi tên sang phải sẽ đưa khối hình sang phải. Phím space sẽ giúp người chơi xoay khối hình theo ý muốn.

Nếu người dùng nhấn nút đăng nhập thì chương trình sẽ đưa người dùng trở về giao diện đăng nhập như trên.

## 3.3. Giao diện quản lý chương trình rau củ

Hình 5 là giao diện quản lý chính của chuong trình khi nhập tài khoản python và mật khẩu là python chương trình sẽ đưa đến giao diện chính nếu nhấn vào nút đăng nhập.



Hinh 5 Giao diện quản lý chính

Phần Date sẽ lấy ngày hiện tại trên máy tính và phần Time sẽ lấy giờ hiện tại trên máy tính của người dùng có chức năng như một chiếc đồng hồ. Dưới đây là code của chiếc đồng hồ thông minh này:

|  |
| --- |
| **def** clock():  date = time.strftime(**'%d/%m/%Y'**)  currenttime = time.strftime(**'%H:%M:%S'**)  dateTimeLabel.config(text=**f' Date: {**date**}\nTime: {**currenttime**}'**)  dateTimeLabel.after(1000, clock) |

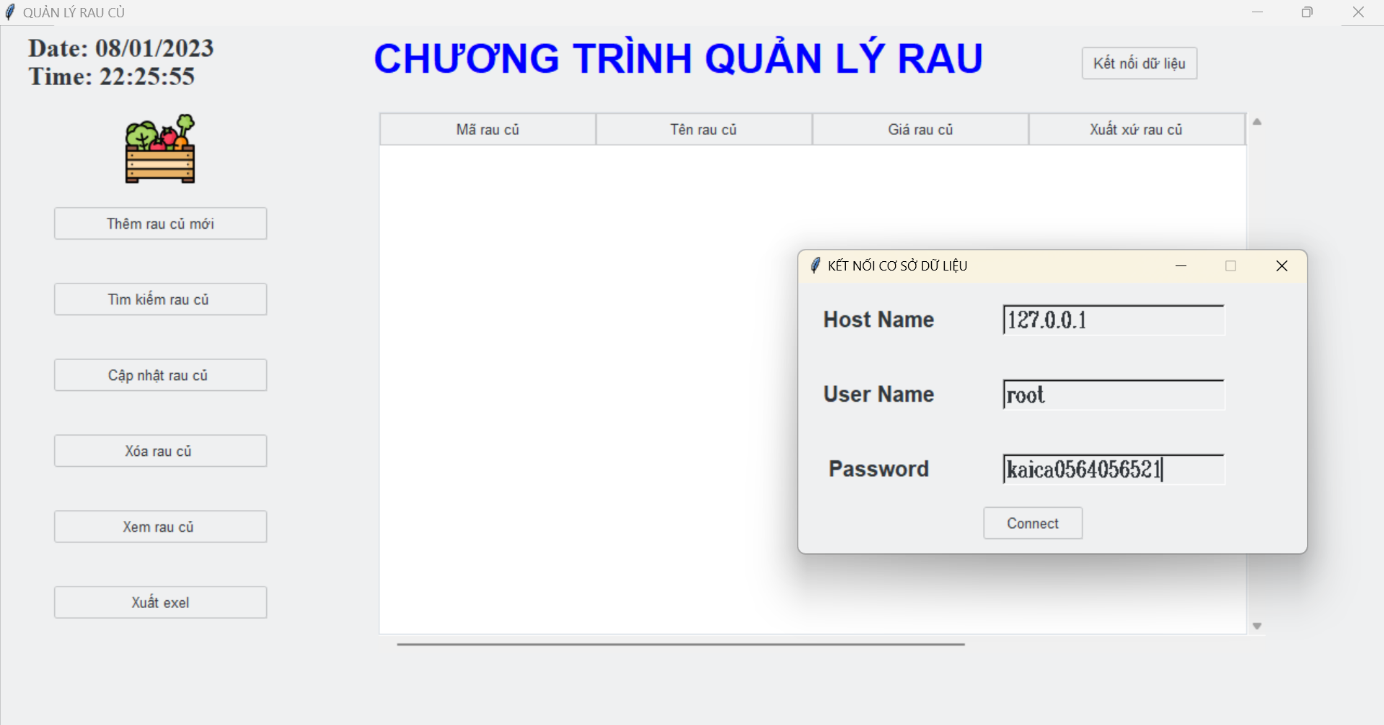
Phần chương trình quản lý rau củ là slide chuyển động liên tục. Dưới đây là code của phần này:

|  |
| --- |
| count = 0 text = **''** *#Hàm chuyển slide* **def** slider():  **global** text, count  **if** count == len(s):  count = 0  text = **''** text = text + s[count]  sliderLabel.config(text=text)  count += 1  sliderLabel.after(300, slider) |

Phần giao diện hiển thị cột và hàng là Treeview để hiện thị danh sách rau củ. Cần import thư viện ttkthemes của tkinter để có giao diện đẹp hơn. Có thể sử dụng giao diện khác tùy theo ý thích dươi đây là code:

|  |
| --- |
| *#Tạo khung chương trình* root = ttkthemes.ThemedTk() root.get\_themes() root.set\_theme(**'breeze'**)#theme đã dùng là breeze root.geometry(**'1174x500+50+20'**) root.title(**'QUẢN LÝ RAU CỦ'**) |

## 3.4. Giao diện kết nối cơ sở dữ liệu

Hình 6 là giao diện kết nối với cơ sở dữ liệu MySQL ở đây sẽ sử dụng hostname, username, password trên máy của người dùng. Trên máy tôi hostname là 127.0.0.1, username là root và password là kaica0564056521.

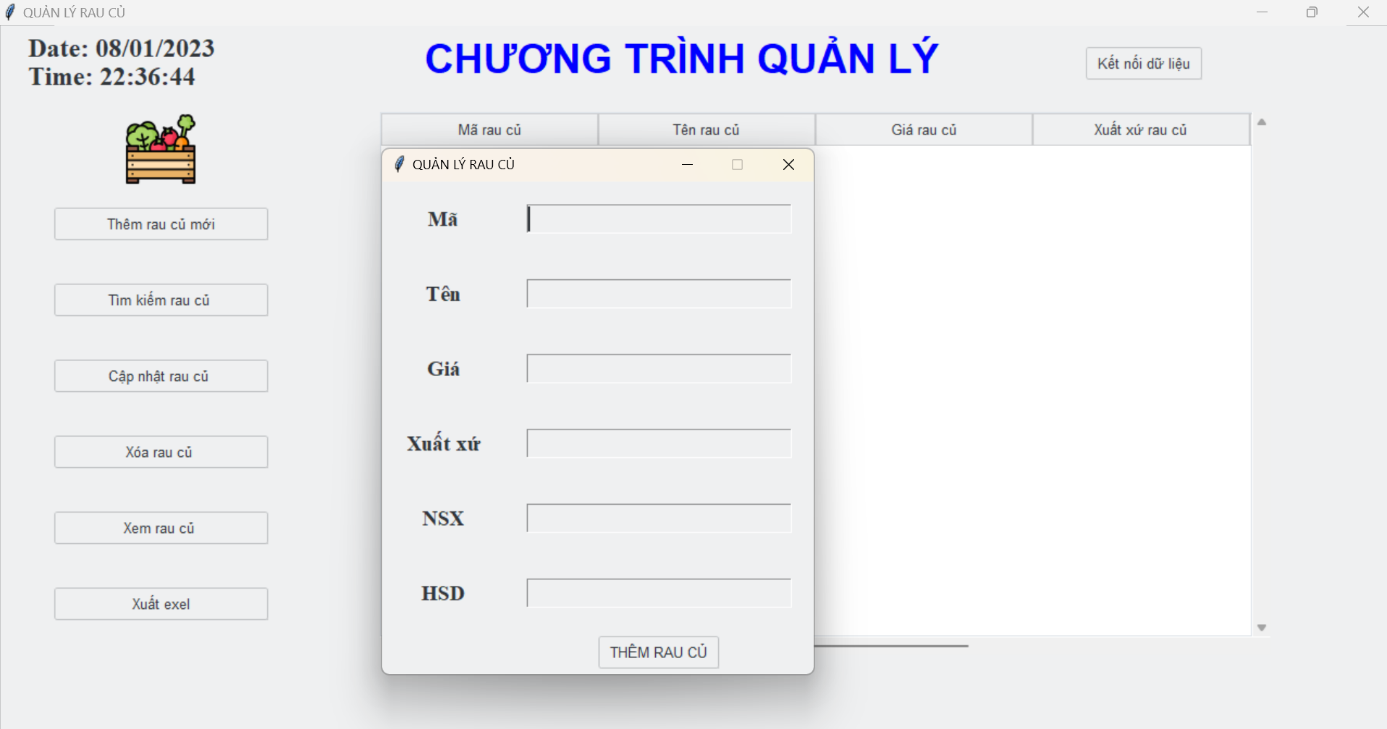
Hinh 6 Giao diện kết nối cơ sở dữ liệu

Sau đó nhấn connect chương trình sẽ kết nối với cơ sở dữ liệu. Trường hợp cơ sở dữ liệu người dùng chưa tạo chương trình sẽ tạo tự động giúp cho người dùng như code bên dưới:

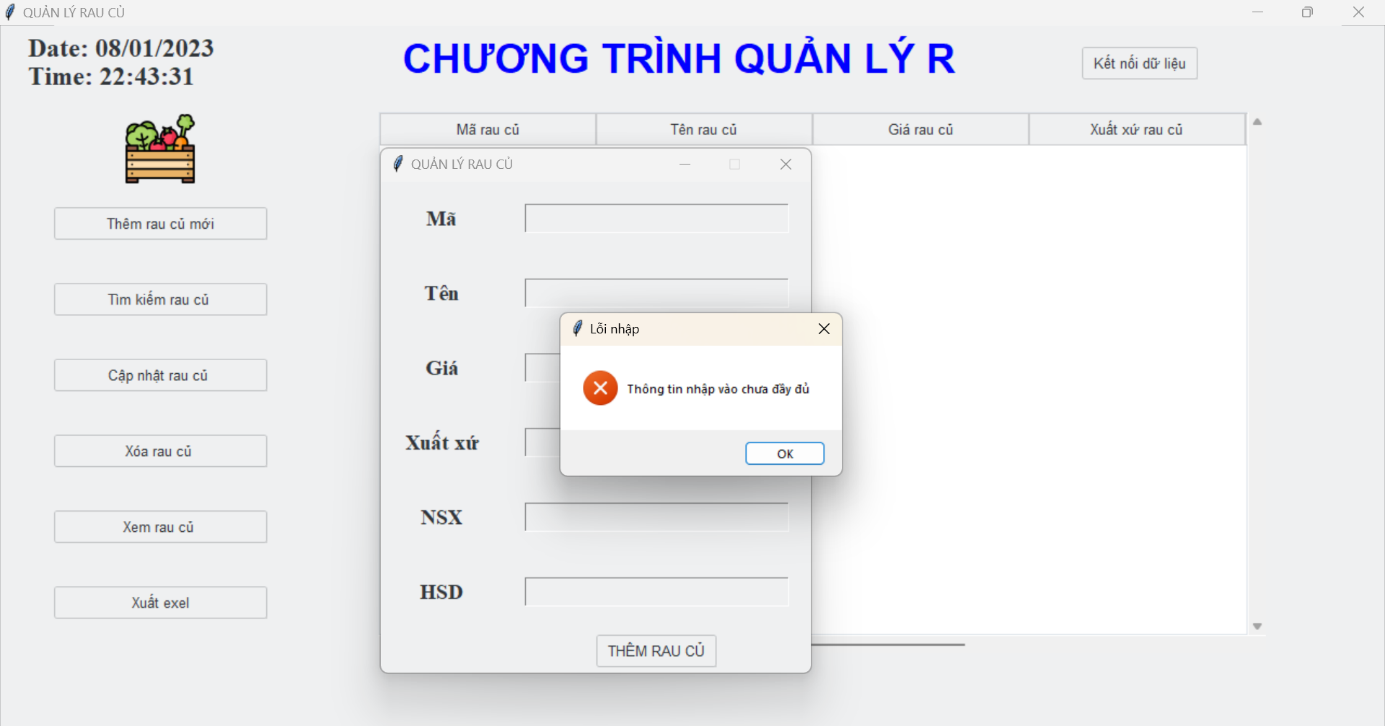
|  |
| --- |
| **def** connect\_database():  *#Hàm keets nối với mysql* **def** connect():  **global** mycusor, con  **try**:  con = pymysql.connect(host=hostnameEntry.get(),  user=usernameEntry.get(),  password=passwordEntry.get())  mycusor = con.cursor()   **except**:  messagebox.showerror(**'Lỗi kết nối'**,  **'Không thể kết nối với cơ sở dữ liệu'**,  parent=connectWindow)  **return  try**:  *#Tạo database nếu chưa có* query = **'create database ql\_raucu'** mycusor.execute(query)  query = **'use ql\_raucu'** mycusor.execute(query)  query = **'create table raucu(ma varchar(50) not null primary key,'** \  **'ten varchar(50) not null, gia varchar(50) not null,'** \  **'xuatxu varchar(50), nxs varchar(50) not null,'** \  **'hsd varchar(50) not null)'** mycusor.execute(query)   **except**:  *#Sử dụng database đã có* query = **'use ql\_raucu'** mycusor.execute(query)  messagebox.showinfo(**'Thông báo'**,  **'Kết nối thành công.'**,parent=connectWindow)  connectWindow.destroy()  addButton.config(state=NORMAL)  deleteButton.config(state=NORMAL)  updateButton.config(state=NORMAL)  exportButton.config(state=NORMAL)  showButton.config(state=NORMAL)  searchButton.config(state=NORMAL)    connectWindow=Toplevel()  connectWindow.geometry(**'470x250+730+230'**)  connectWindow.title(**'KẾT NỐI CƠ SỞ DỮ LIỆU'**)  connectWindow.resizable(0,0)   *#Tạo các sự kiện* hostnameLabel = Label(connectWindow,  text=**'Host Name'**,  font=(**'arial'**, 15, **'bold'**))  hostnameLabel.grid(row=0, column=0, padx=20)   hostnameEntry = Entry(connectWindow,  font=(**'roman'**, 15, **'bold'**), bd=2)  hostnameEntry.grid(row=0, column=1, pady= 20, padx= 40)   usernameLabel = Label(connectWindow,  text=**'User Name'**,  font=(**'arial'**, 15, **'bold'**))  usernameLabel.grid(row=1, column=0, padx=20)   usernameEntry = Entry(connectWindow,  font=(**'roman'**, 15, **'bold'**), bd=2)  usernameEntry.grid(row=1, column=1, pady=20, padx=40)   passwordLabel = Label(connectWindow,  text=**'Password'**,  font=(**'arial'**, 15, **'bold'**))  passwordLabel.grid(row=2, column=0, padx=20)  passwordEntry = Entry(connectWindow,  font=(**'roman'**, 15, **'bold'**), bd=2)  passwordEntry.grid(row=2, column=1, pady=20, padx=40)   connectButton = ttk.Button(connectWindow, text=**'Connect'**, command=connect)  connectButton.grid(row=3, columnspan=2) |

## 3.5. Giao diện khi thêm rau củ mới

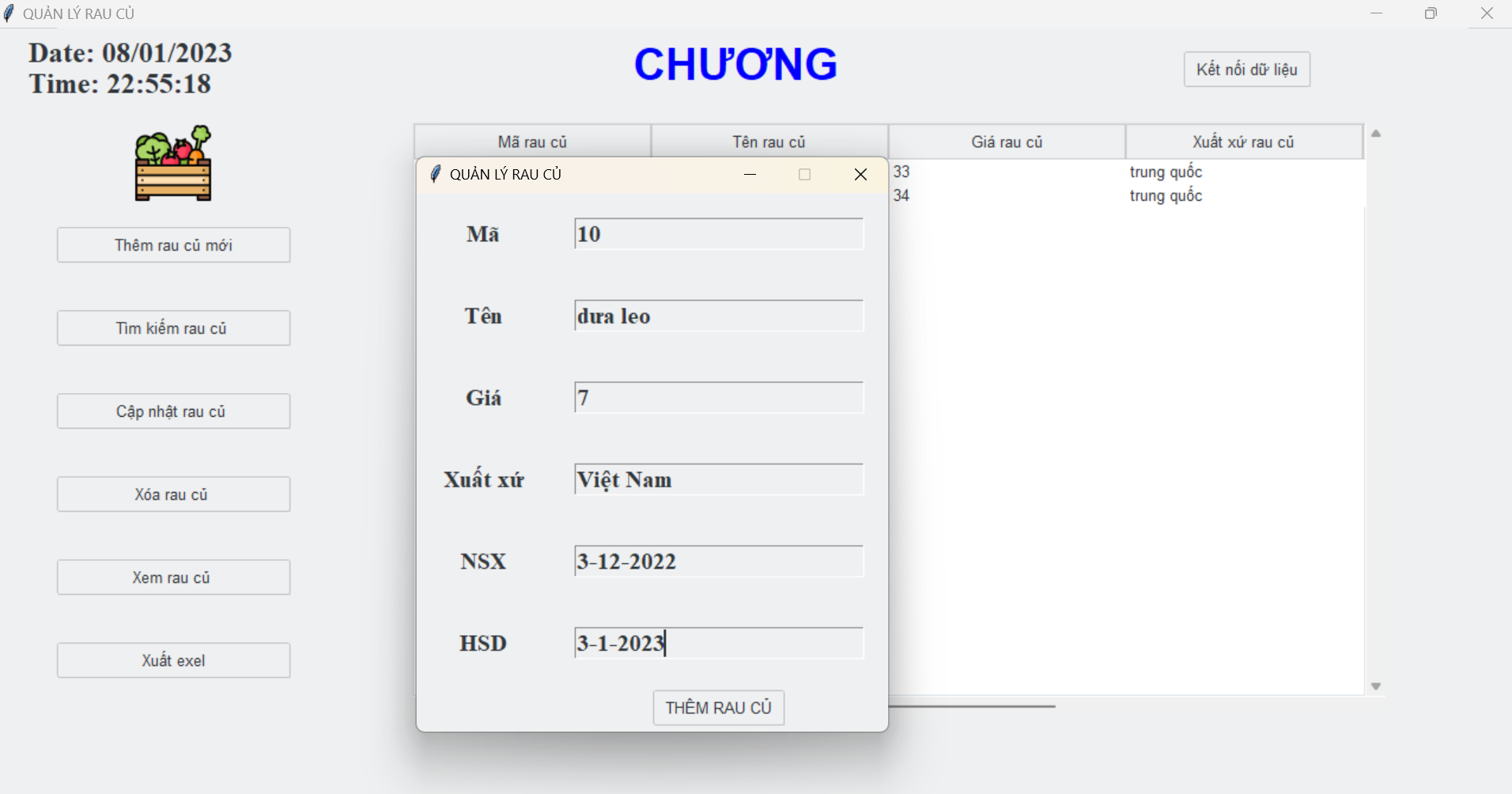
Hình 7 là giao diện khi người dùng nhấn vào thêm rau củ mới.

 Khi người dùng đã nhập thông tin vào các Entry và nhấn thêm rau củ. Trường hợp người dùng nhập đúng các thông tin thì dữ liệu mới sẽ được thêm vào cơ sở dữ liệu. Trường hợp người dùng không nhập gì hoặc nhập mã đã có trong cơ sở dữ liệu thì dữ liệu nhập của người dùng sẽ báo lỗi và không được thêm vào cơ sở dữ liệu như hình giao diện lỗi khi nhập bên dưới:

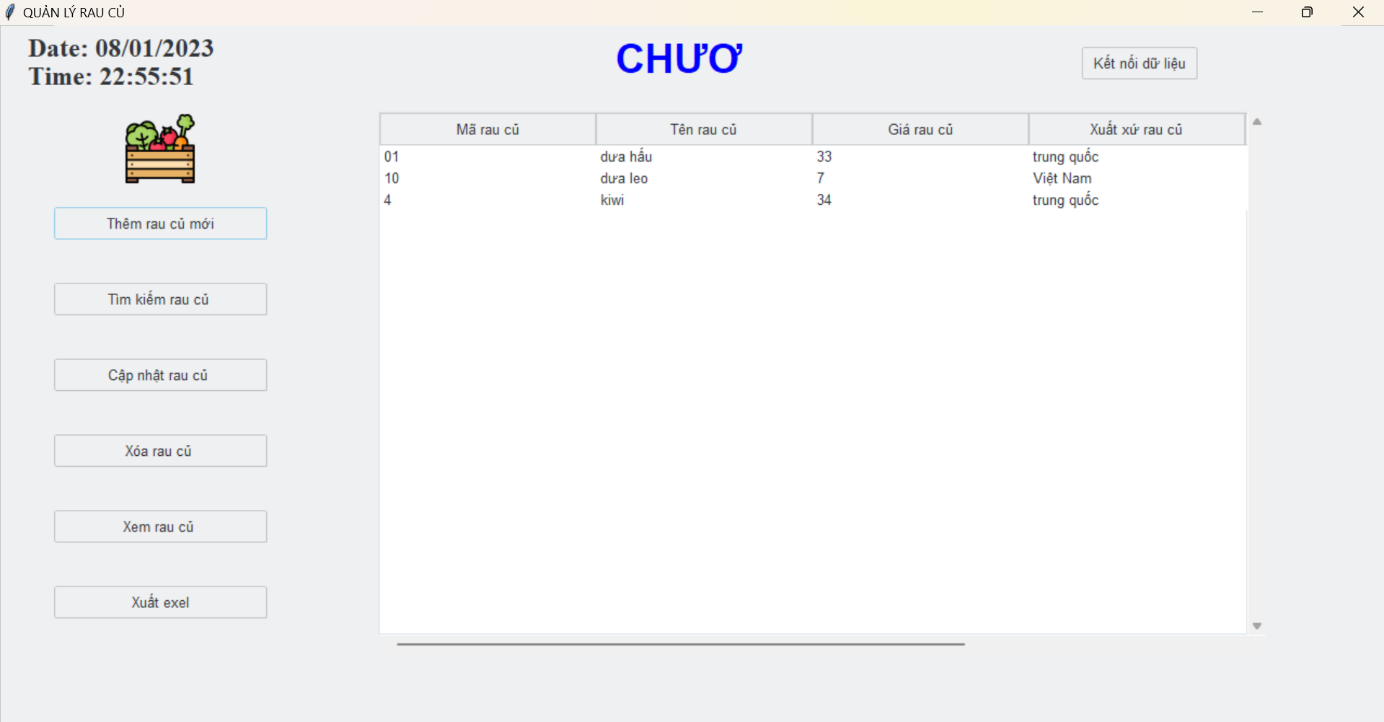
Hinh 7 Giao diện thêm mới

 Dưới đây là hình 9 ảnh đã nhập đầy đủ và chính xác thông tin.

Hinh 8 Giao diện thêm lỗi



Hinh 9 Giao diện thêm thành công

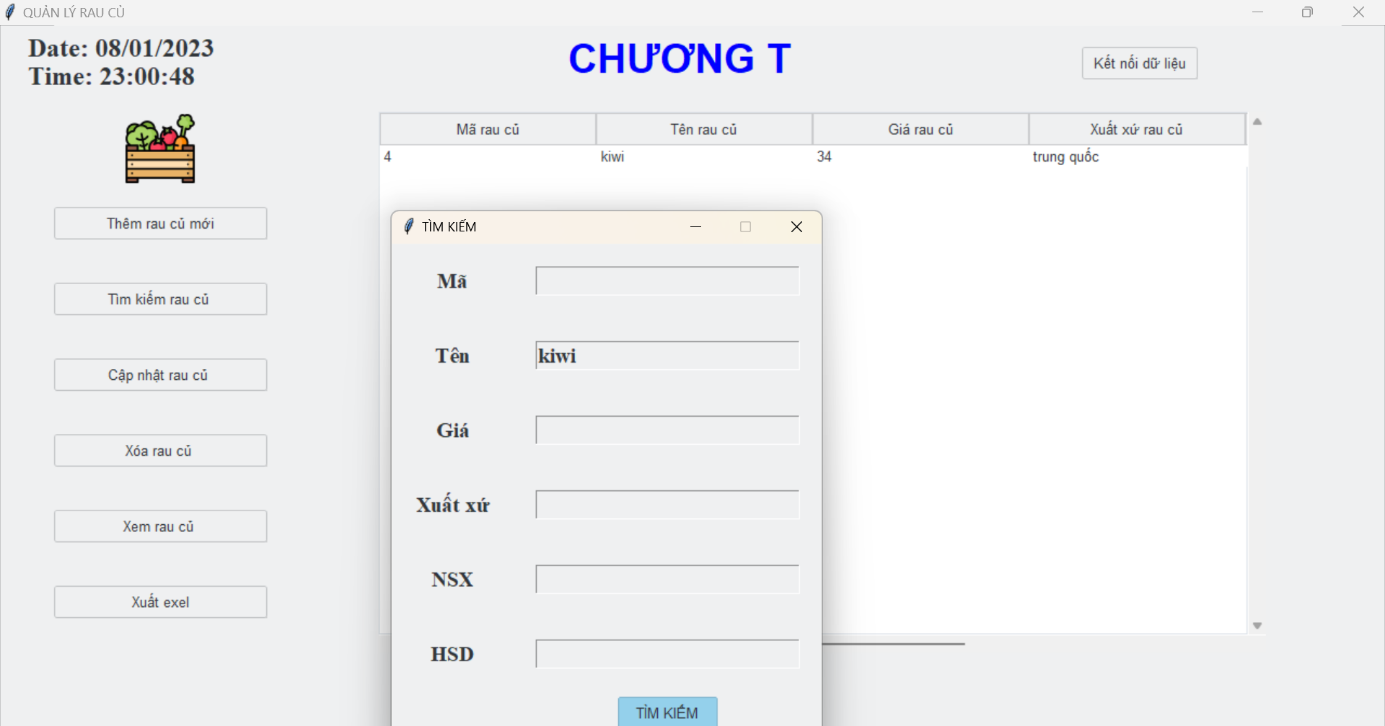
Dưới đây là hình 10 sản phẩm sau khi thêm vào cơ sở dữ liệu.

Hinh 10 Giao diện sau khi thêm sản phẩm thành công

Dưới đây là code thêm rau củ:

|  |
| --- |
| *#Hàm thêm rau củ mới* **def** add\_raucu():  *# Hàm lấy thêm data vào database* **def** add\_data():  **if** idEntry.get() == **'' or** nameEntry.get()==**'' or** priceEntry.get()==**'' or** makeinEntry.get()==**'' or** nsxEntry.get()==**''or** hsdEntry.get()==**''**:  messagebox.showerror(**'Lỗi nhập'**,  **'Thông tin nhập vào chưa đầy đủ'**,  parent=add\_window)  **else**:  **try**:  query = **'insert into raucu values(%s,%s,%s,%s,%s,%s)'** mycusor.execute(query,(idEntry.get(), nameEntry.get(),  priceEntry.get(), makeinEntry.get(),  nsxEntry.get(), hsdEntry.get()))  con.commit()  result=messagebox.askyesno(**'Xác nhận'**,**'Dữ liệu đã được thêm vào. '  'Bạn có muốn xóa form?'**,  parent= add\_window)  **if** result:  idEntry.delete(0, END)  nameEntry.delete(0, END)  priceEntry.delete(0, END)  makeinEntry.delete(0, END)  nsxEntry.delete(0, END)  hsdEntry.delete(0, END)  **else**:  **pass  except**:  messagebox.showerror(**'Lỗi thêm dữ liệu'**,  **'Mã rau củ bị trùng'**,  parent=add\_window)  **return** query = **'select \* from raucu'** mycusor.execute(query)  fetched\_data = mycusor.fetchall()  vg\_table.delete(\*vg\_table.get\_children())   **for** data **in** fetched\_data:  datalist = list(data)  vg\_table.insert(**''**,END, values=datalist)    add\_window = Toplevel()  add\_window.grab\_set()  add\_window.resizable(**False**, **False**)   *#Mã* idLabel = Label(add\_window, text=**'Mã'**,  font=(**'times new roman'**, 15, **'bold'**))  idLabel.grid(row=0, column=0, padx=20,pady=20)  idEntry = Entry(add\_window,  font=(**'times new roman'**, 15, **'bold'**),  width=24)  idEntry.grid(row=0, column=1, padx=20,pady=20)   *# tên* nameLabel = Label(add\_window, text=**'Tên'**,  font=(**'times new roman'**, 15, **'bold'**))  nameLabel.grid(row=1, column=0, padx=20, pady=20)  nameEntry = Entry(add\_window,  font=(**'times new roman'**, 15, **'bold'**),  width=24)  nameEntry.grid(row=1, column=1, padx=20, pady=20)   *# Giá* priceLabel = Label(add\_window, text=**'Giá'**,  font=(**'times new roman'**, 15, **'bold'**))  priceLabel.grid(row=2, column=0, padx=20, pady=20)  priceEntry = Entry(add\_window,  font=(**'times new roman'**, 15, **'bold'**),  width=24)  priceEntry.grid(row=2, column=1, padx=20, pady=20)   *# Xuâất xứ* makeinLabel = Label(add\_window, text=**'Xuất xứ'**,  font=(**'times new roman'**, 15, **'bold'**))  makeinLabel.grid(row=3, column=0, padx=20, pady=20)  makeinEntry = Entry(add\_window,  font=(**'times new roman'**, 15, **'bold'**),  width=24)  makeinEntry.grid(row=3, column=1, padx=20, pady=20)   *# Ngày sản xuất* nsxLabel = Label(add\_window, text=**'NSX'**,  font=(**'times new roman'**, 15, **'bold'**))  nsxLabel.grid(row=4, column=0, padx=20, pady=20)  nsxEntry = Entry(add\_window,  font=(**'times new roman'**, 15, **'bold'**),  width=24)  nsxEntry.grid(row=4, column=1, padx=20, pady=20)   *# hạn sử dụng* hsdLabel = Label(add\_window, text=**'HSD'**,  font=(**'times new roman'**, 15, **'bold'**))  hsdLabel.grid(row=5, column=0, padx=20, pady=20)  hsdEntry = Entry(add\_window,  font=(**'times new roman'**, 15, **'bold'**),  width=24)  hsdEntry.grid(row=5, column=1, padx=20, pady=20)   *#nút thêm* add\_vg\_button=ttk.Button(add\_window, text = **'THÊM RAU CỦ'**,  command=add\_data)  add\_vg\_button.grid(row=6, column=1, pady=5, padx=5) |

## 3.6. Giao diện tìm thêm rau củ

Hình 11 là giao diện tìm kiếm rau củ theo tất cả các tiêu chí như: mã, tên, giá, xuất xứ, ngày sản xuất, hạn sử dụng. Ở đây tôi sẽ tìm kiếm theo tên sản phẩm.

Hinh 11 Giao diện tìm kiếm

Người dùng sẽ nhập vào thông tin cần tìm chương trình đó lấy dữ liệu đó sao đó dùng câu truy vấn để dò trong cơ sở dữ liệu nếu có sẽ lấy lên và hiện trên treeview cho người dùng xem. Dưới đây là code:

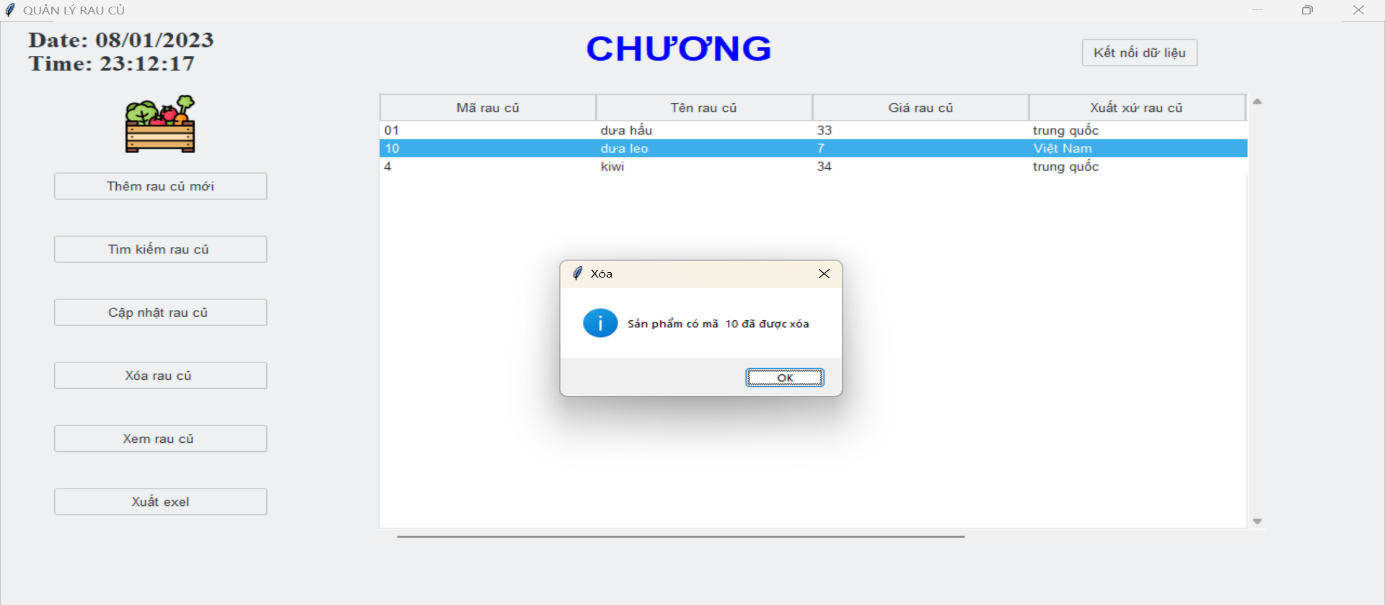
|  |
| --- |
| *#Hàm tìm kiếm* **def** search\_vg():  **def** search\_data():  query = **'select \* from raucu where ma = %s or ten=%s or gia=%s or xuatxu=%s or nxs=%s or hsd=%s'** mycusor.execute(query, (idEntry.get(), nameEntry.get(),  priceEntry.get(), makeinEntry.get(),  nsxEntry.get(), hsdEntry.get()))  vg\_table.delete(\*vg\_table.get\_children())  fetched\_data= mycusor.fetchall()  **for** data **in** fetched\_data:  vg\_table.insert(**''**, END, values=data)   search\_window = Toplevel()  search\_window.grab\_set()  search\_window.resizable(**False**, **False**)  search\_window.title(**'TÌM KIẾM'**)   *# Mã* idLabel = Label(search\_window, text=**'Mã'**,  font=(**'times new roman'**, 15, **'bold'**))  idLabel.grid(row=0, column=0, padx=20, pady=20)  idEntry = Entry(search\_window,  font=(**'times new roman'**, 15, **'bold'**),  width=24)  idEntry.grid(row=0, column=1, padx=20, pady=20)   *# tên* nameLabel = Label(search\_window, text=**'Tên'**,  font=(**'times new roman'**, 15, **'bold'**))  nameLabel.grid(row=1, column=0, padx=20, pady=20)  nameEntry = Entry(search\_window,  font=(**'times new roman'**, 15, **'bold'**),  width=24)  nameEntry.grid(row=1, column=1, padx=20, pady=20)   *# Giá* priceLabel = Label(search\_window, text=**'Giá'**,  font=(**'times new roman'**, 15, **'bold'**))  priceLabel.grid(row=2, column=0, padx=20, pady=20)  priceEntry = Entry(search\_window,  font=(**'times new roman'**, 15, **'bold'**),  width=24)  priceEntry.grid(row=2, column=1, padx=20, pady=20)   *# Xuâất xứ* makeinLabel = Label(search\_window, text=**'Xuất xứ'**,  font=(**'times new roman'**, 15, **'bold'**))  makeinLabel.grid(row=3, column=0, padx=20, pady=20)  makeinEntry = Entry(search\_window,  font=(**'times new roman'**, 15, **'bold'**),  width=24)  makeinEntry.grid(row=3, column=1, padx=20, pady=20)   *# Ngày sản xuất* nsxLabel = Label(search\_window, text=**'NSX'**,  font=(**'times new roman'**, 15, **'bold'**))  nsxLabel.grid(row=4, column=0, padx=20, pady=20)  nsxEntry = Entry(search\_window,  font=(**'times new roman'**, 15, **'bold'**),  width=24)  nsxEntry.grid(row=4, column=1, padx=20, pady=20)   *# hạn sử dụng* hsdLabel = Label(search\_window, text=**'HSD'**,  font=(**'times new roman'**, 15, **'bold'**))  hsdLabel.grid(row=5, column=0, padx=20, pady=20)  hsdEntry = Entry(search\_window,  font=(**'times new roman'**, 15, **'bold'**),  width=24)  hsdEntry.grid(row=5, column=1, padx=20, pady=20)   *# nút thêm* search\_vg\_button = ttk.Button(search\_window, text=**'TÌM KIẾM'**,  command=search\_data)  search\_vg\_button.grid(row=6, column=1, pady=5, padx=5) |

## 3.7. Giao diện xóa

Người dụng chọn dòng rau củ cần xóa sau khi đó nhấn vào nút xóa. Khi này chương sẽ tiến hành truy vấn đến cơ sở dữ liệu và lọc ra mã được chọn trên dòng này nếu mã có trong cơ sở dữ liệu sẽ tiến hành xóa dữ liệu về dòng rau củ đó như hình 12 và hình 13.



Hinh 12 Giao diện sau khi xóa

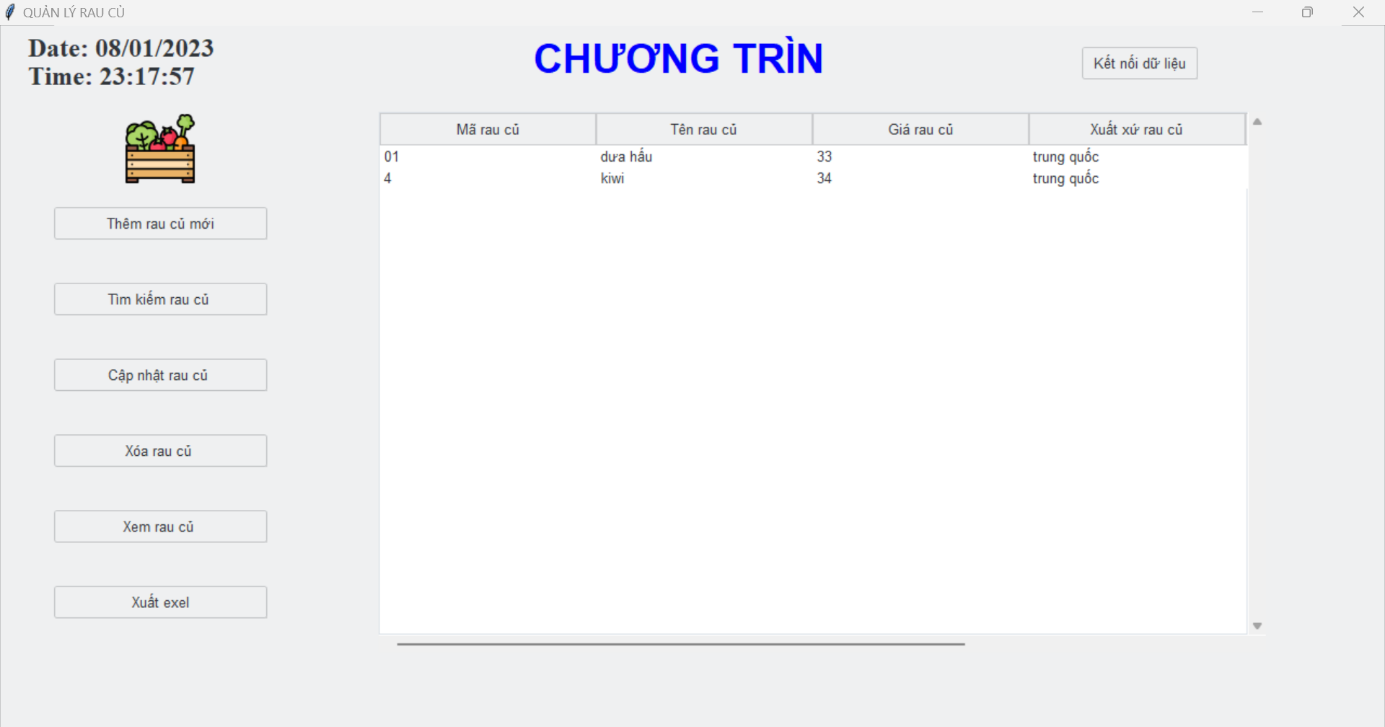
Dưới đây là code của hàm xóa rau củ:

Hinh 13 Giao diện trước khi xóa

|  |
| --- |
| *#Hàm xóa* **def** delete\_button():  indexing = vg\_table.focus()  content = vg\_table.item(indexing)  content\_id=content[**'values'**][0]  query = **'delete from raucu where ma = %s'** mycusor.execute(query, (content\_id))  con.commit()  messagebox.showinfo(**'Xóa'**,  **f'Sản phẩm có mã {**content\_id**} đã được xóa'**)  query = **'select \* from raucu'** mycusor.execute(query)  fetched\_data = mycusor.fetchall()  vg\_table.delete(\*vg\_table.get\_children())  **for** data **in** fetched\_data:  vg\_table.insert(**''**,END, values=data) |

## 3.8. Giao diện xem danh sách rau củ

Hình 14 là giao diện xem danh sách rau củ

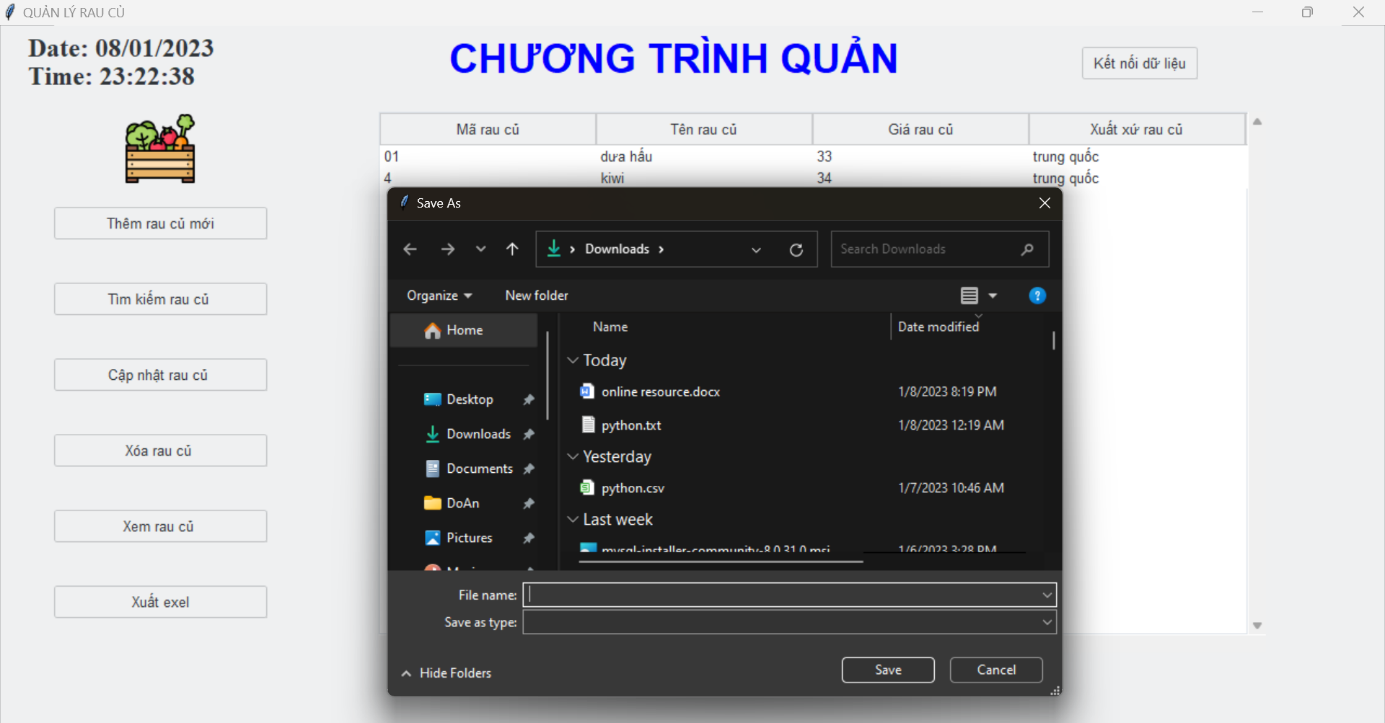
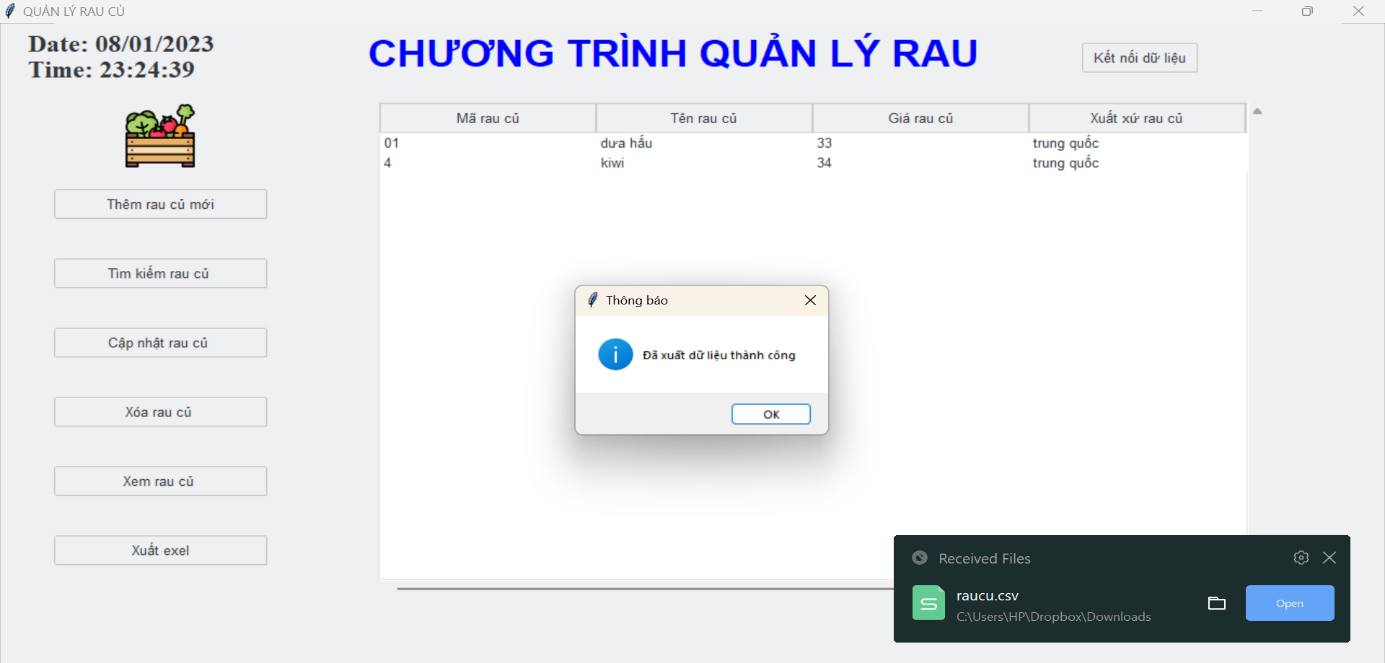
Khi người dùng nhấn vào nút xem rau củ thì chương trình sẽ truy vấn lấy toàn bộ dữ liệu từ bảng raucu trong cơ sở dữ liệu đem lên treeview cho người dùng xem. Dưới đây là code của nút xem rau củ.

Hinh 14 Giao diện xem danh sách rau củ

|  |
| --- |
| *#hàm hiện bảng* **def** show\_button():  query = **'select \* from raucu'** mycusor.execute(query)  fetched\_data = mycusor.fetchall()  vg\_table.delete(\*vg\_table.get\_children())  **for** data **in** fetched\_data:  vg\_table.insert(**''**, END, values=data) |

## 3.9. Giao diện xuất danh sách rau củ ra file exel

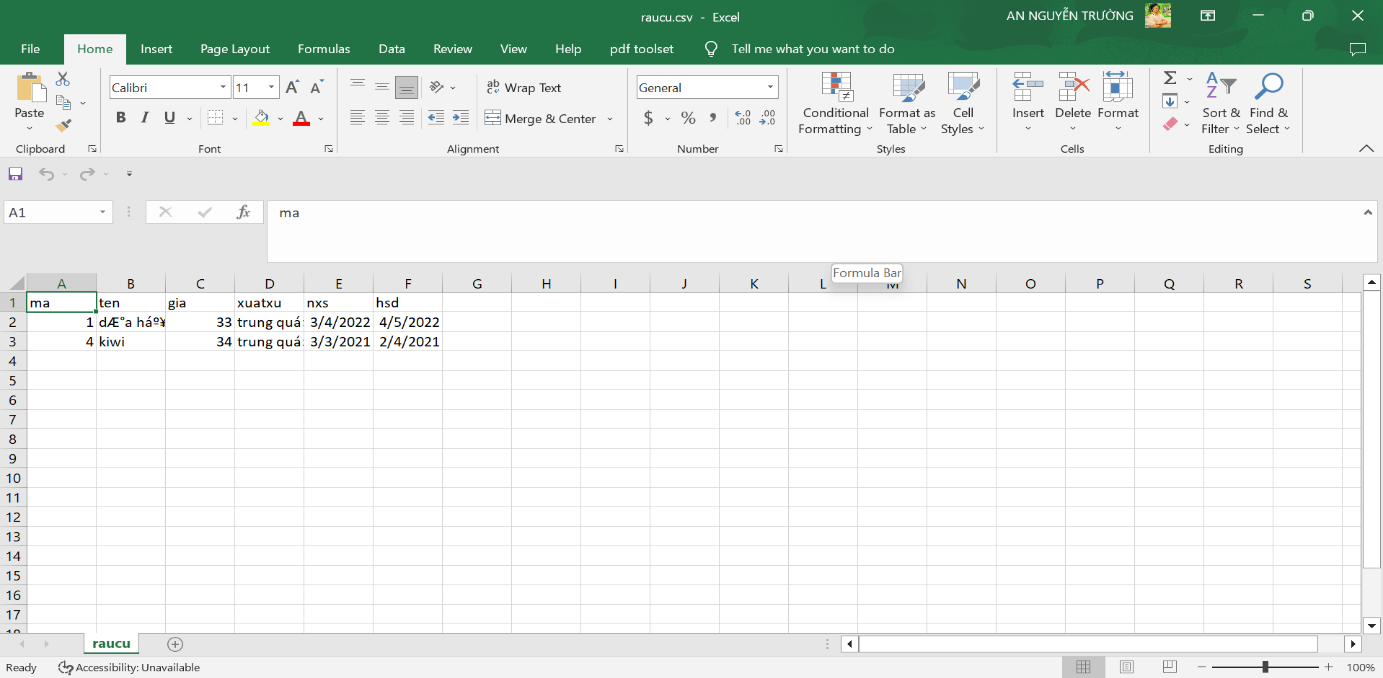
Hình 15 là giao diện khi người dùng nhấn vào nút xuất exel.

Khi này người dùng sẽ chọn nơi lưu file exel, sau đó nhấn save khi đó chương trình sẽ đem toàn bộ dữ liệu trên treeview đổ vào file exel như hình 16 bên dưới.

Hinh 15 Giao diện xuất file excel

Hinh 16 Giao diện xuất file excel

Dưới đây là hình file exel khi đã xuất ra.

Dưới đây là code của nút xuất file excel.

Hinh 17 File excel đã xuất

|  |
| --- |
| *#Các hàm #Hàm xuất dữ liệu ra exel* **def** export\_data():  url = filedialog.asksaveasfilename(defaultextension=**'.csv'**)  indexing = vg\_table.get\_children()  newlist=[]  **for** index **in** indexing:  content=vg\_table.item(index)  datalist=content[**'values'**]  newlist.append(datalist)  table = pandas.DataFrame(newlist, columns=[**'ma'**, **'ten'**,  **'gia'**,**'xuatxu'**,  **'nxs'**, **'hsd'**])  table.to\_csv(url,index=**False**)  messagebox.showinfo(**'Thông báo'**,**'Đã xuất dữ liệu thành công'**) |

## 3.10. Giao diện cập nhật danh sách rau củ

Hinh 18 Giao diện cập nhật danh sách rau củ

Khi người dùng muốn cập nhật lại rau củ nào đó chỉ cần chọn vào dòng cần cập nhật sau đó nhấn vào nút cập nhật sẽ có giao diện như hình 18:

Khi người dùng nhấn cập nhật chương trình sẽ lấy mã dựa trên dòng dữ liệu mà người dùng chọn sau đó truy vấn đến cơ sở dữ liệu nếu và cập nhật lại dữ liệu cho dòng đó. Dưới đây là code cho nút cập nhật.

|  |
| --- |
| *#hàm cập nhật* **def** update\_button():  **def** update\_data():  query = **'update raucu set ten=%s, gia=%s, xuatxu=%s,'** \  **'nxs=%s, hsd=%s where ma=%s'** mycusor.execute(query,(nameEntry.get(),priceEntry.get(),  makeinEntry.get(),nsxEntry.get(),  hsdEntry.get(), idEntry.get()))  con.commit()  messagebox.showinfo(**'Thông báo'**,**f'Mã {**idEntry.get()**}'  f' đã được cập nhật'**)  update\_window.destroy()  show\_button()   update\_window = Toplevel()  update\_window.grab\_set()  update\_window.resizable(**False**, **False**)  update\_window.title(**'CẬP NHẬT'**)   *# Mã* idLabel = Label(update\_window, text=**'Mã'**,  font=(**'times new roman'**, 15, **'bold'**))  idLabel.grid(row=0, column=0, padx=20, pady=20)  idEntry = Entry(update\_window,  font=(**'times new roman'**, 15, **'bold'**),  width=24)  idEntry.grid(row=0, column=1, padx=20, pady=20)   *# tên* nameLabel = Label(update\_window, text=**'Tên'**,  font=(**'times new roman'**, 15, **'bold'**))  nameLabel.grid(row=1, column=0, padx=20, pady=20)  nameEntry = Entry(update\_window,  font=(**'times new roman'**, 15, **'bold'**),  width=24)  nameEntry.grid(row=1, column=1, padx=20, pady=20)   *# Giá* priceLabel = Label(update\_window, text=**'Giá'**,  font=(**'times new roman'**, 15, **'bold'**))  priceLabel.grid(row=2, column=0, padx=20, pady=20)  priceEntry = Entry(update\_window,  font=(**'times new roman'**, 15, **'bold'**),  width=24)  priceEntry.grid(row=2, column=1, padx=20, pady=20)   *# Xuâất xứ* makeinLabel = Label(update\_window, text=**'Xuất xứ'**,  font=(**'times new roman'**, 15, **'bold'**))  makeinLabel.grid(row=3, column=0, padx=20, pady=20)  makeinEntry = Entry(update\_window,  font=(**'times new roman'**, 15, **'bold'**),  width=24)  makeinEntry.grid(row=3, column=1, padx=20, pady=20)   *# Ngày sản xuất* nsxLabel = Label(update\_window, text=**'NSX'**,  font=(**'times new roman'**, 15, **'bold'**))  nsxLabel.grid(row=4, column=0, padx=20, pady=20)  nsxEntry = Entry(update\_window,  font=(**'times new roman'**, 15, **'bold'**),  width=24)  nsxEntry.grid(row=4, column=1, padx=20, pady=20)   *# hạn sử dụng* hsdLabel = Label(update\_window, text=**'HSD'**,  font=(**'times new roman'**, 15, **'bold'**))  hsdLabel.grid(row=5, column=0, padx=20, pady=20)  hsdEntry = Entry(update\_window,  font=(**'times new roman'**, 15, **'bold'**),  width=24)  hsdEntry.grid(row=5, column=1, padx=20, pady=20)   *# nút thêm* update\_vg\_button = ttk.Button(update\_window, text=**'CẬP NHẬT'**,  command=update\_data)  update\_vg\_button.grid(row=6, column=1, pady=5, padx=5)   indexing=vg\_table.focus()  content=vg\_table.item(indexing)  listdata=content[**'values'**]  idEntry.insert(0, listdata[0])  nameEntry.insert(0,listdata[1])  priceEntry.insert(0,listdata[2])  makeinEntry.insert(0,listdata[3])  nsxEntry.insert(0,listdata[4])  hsdEntry.insert(0,listdata[5]) |

# PHẦN 4: KẾT LUẬN

## 4.1. Kết quả đạt được

* Xây dựng thành công ứng dụng quản lý rau củ.
* Xây dựng thành công chương trình có kết nối cơ sở dữ liệu với sql.
* Xây dựng được hai game giải trí đơn giản.

## 4.2. Hạn chế

* Cơ sở dữ liệu chưa được đa dạng.
* Chưa bẩy lỗi hết các trường hợp cũng như phàn giao diện chưa được tối ưu.

## Hướng phát triển

* Cải thiện giao diện tối ưu cho người dùng.
* Xây dựng cơ sở dữ liệu đa dạng, phong phú.

# TÀI LIỆU THAM KHẢO

[1]. Lê Minh Thư, Nguyễn Quốc Anh, Nguyễn Thị Mỹ Dung, 2022, Đồng Tháp, Giáo trình “Lập trình python và ứng dụng”.

[2] Viettuts, Ngày đăng 2016, <https://viettuts.vn/python-tkinter> Truy cập: 1/1/2023

[3] Nguyễn Mạnh Thắng, Ngày đăng 12/11/2020, <https://viblo.asia/p/module-time-trong-python-07LKXeBkZV4> Truy cập: 13/1/2023

[4] laptrinhtudau, Ngày đăng 17/3/2022 <https://laptrinhtudau.com/ket-noi-mysql-voi-python/> Truy cập: 2/1/2022

[5] Nguyễn Văn Hiếu, Ngày đăng 02/12/2018 <https://blog.luyencode.net/thu-vien-pandas-python/> Truy cập: 2/1/2022