**TRƯỜNG ĐẠI HỌC AN GIANG  
KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN CHUYÊN ĐỀ PYTHON**

**Đồ Án**

**Ứng Dụng Bán Hàng Pay and Claim**

**SINH VIÊN:**

DTH225650 – Huỳnh Quốc Huy

**An Giang, tháng 9 năm 2023**

**MỤC LỤC**

[**I.** **GIỚI THIỆU:** 4](#_Toc202781168)

[**II.** **THỰC HIỆN ĐỒ ÁN** 4](#_Toc202781169)

[**1.** **Thời gian thực hiện:** 28/10/2024 – 9/12/2024 4](#_Toc202781170)

[**2.** **Thư Viện và Biến Toàn Cục** 5](#_Toc202781171)

[**a)** **Thư viện:** 5](#_Toc202781172)

[**b)** **Biến toàn cục** 6](#_Toc202781173)

[**3.** **Kết nối Cơ Sở Dữ Liệu** 7](#_Toc202781174)

[**a)** **Database MySQL** 7](#_Toc202781175)

[**b)** **Hàm kết nối với Database** 8](#_Toc202781176)

[**c)** **Hàm thực thi truy vấn** 9](#_Toc202781177)

[**4.** **Đăng Nhập** 10](#_Toc202781178)

[**5.** **Đăng Ký** 13](#_Toc202781179)

[**6.** **Đổi Mật Khẩu** 16](#_Toc202781180)

[**7.** **Hàm Giao Diện Người Dùng** 18](#_Toc202781181)

[**a)** **Hàm Làm xuất hiện chính giữa màn hình** 18](#_Toc202781182)

[**b)** **Hàm Lấy Dữ liệu vào TreeView** 19](#_Toc202781183)

[**c)** **Hàm Làm mới dữ liệu** 20](#_Toc202781184)

[**d)** **Giao diện chính** 20](#_Toc202781185)

[**e)** **Tab Order** 21](#_Toc202781186)

[**f)** **Tab Sơ đồ:** 24](#_Toc202781187)

[**g)** **Tab Tiện Ích** 25](#_Toc202781188)

[**8.** **Binding Event** 26](#_Toc202781189)

[**9.** **Chức Năng:** 27](#_Toc202781190)

[**a)** **Thêm:** 27](#_Toc202781191)

[**b)** **Xóa** 34](#_Toc202781192)

[**c)** **Hủy** 37](#_Toc202781193)

[**d)** **Order** 38](#_Toc202781194)

[**e)** **Thanh Toán** 42](#_Toc202781195)

[**f)** **Thêm, Xóa Sửa Danh Sách Đồ Uống** 46](#_Toc202781196)

[**g)** **Tìm Kiếm** 49](#_Toc202781197)

[**h)** **Lưu Hóa Đơn** 50](#_Toc202781198)

[**i)** **Lưu Thống Kê Doanh Thu** 51](#_Toc202781199)

[**j)** **Xem Hóa Đơn, Doanh Thu** 52](#_Toc202781200)

[**10.** **Giao Diện** 53](#_Toc202781201)

[**III.** **TỰ ĐÁNH GIÁ** 58](#_Toc202781202)

[**IV.** **TÀI LIỆU THAM KHẢO** 59](#_Toc202781203)

1. **GIỚI THIỆU:**

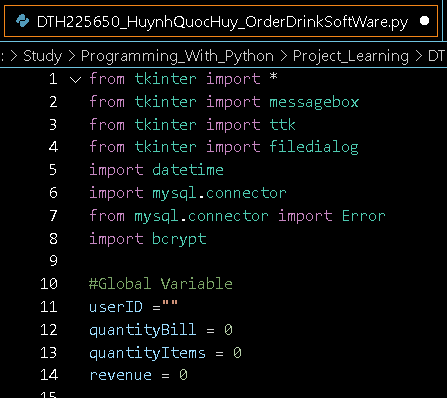
Việt Nam một trong những nước xuất khẩu cà phê hàng đầu thế giới, song song với đó thị trường Việt Nam cũng tiêu thụ số lượng lớn cà phê này và các loại đôg uống khác. Trước bối cảnh đó, trong những năm gần đây, càng ngày càng có nhiều quán cà phê, cửa hàng nước giải khát xuất hiện với nhiều món nước đa dạng, với các mô hình Pay and Claim hoặc kinh doanh truyền thống. Việc bán hàng thủ công trở nên phức tạp với nhiều thủ tục và quản lý dễ sai sót do đó cần có một phần mềm bán hàng để thực hiện những công việc này trở nên tối ưu hơn.

Phần mềm bán nước giải khát trong đồ án này, được phát triển với các chức năng cơ bản của một phần mềm bán nước giải khát cần có. Trong đó các công việc gồm thêm, xóa, hủy món trong Order, đặt hàng cho một Order và thanh toán trong Order đó. Ngoài ra còn có các chức năng như xuất hóa đơn, thống kê doanh thu, lưu và xem hóa đơn, chọn bàn…Phần mềm còn thêm tính năng đăng nhập để đảm bảo tính bảo mật. Bên cạnh đó còn liên kết với cơ sơ dữ liệu để lưu trữ và truy xuất hiệu quả hơn.

1. **THỰC HIỆN ĐỒ ÁN**
2. **Thời gian thực hiện:** 28/10/2024 – 9/12/2024

* **28/10 – 3/11:** Tham khảo tài liệu, lên ý tưởng
* **4/11 – 10/11:** Thiết kế giao diện
* **11/11 – 17/11:** Cải tiến giao diện, các module cơ bản
* **18/11 – 24/11:** Viết các chức năng cơ bản
* **25/11 – 1/12:** Phát triển phần mềm kết nối cơ sở dữ liệu, đăng nhập…
* **2/12 -10/12:** Kiểm thử,chạy demo, hoàn thiện tài liệu, báo cáo đồ án

1. **Thư Viện và Biến Toàn Cục**

****

1. **Thư viện:**

* Dòng: from tkinter import \*

Import tất cả các lớp và hàm từ thư viện tkinter, một thư viện chuẩn của Python để tạo giao diện đồ họa (GUI). Dấu \* có nghĩa là import toàn bộ nội dung từ thư viện, bao gồm các widget như Tk, Label, Button, Entry, Text…

* Dòng from tkinter import messagebox

Import messagebox, một module con của tkinter, được dùng để hiển thị các hộp thoại thông báo như thông báo lỗi, cảnh báo, hay xác nhận (ví dụ: messagebox.showinfo, messagebox.showwarning).

* Dòng from tkinter import ttk

Import ttk (Themed Tkinter Widgets), một module cung cấp các widget nâng cao hơn từ tkinter với giao diện hiện đại và hỗ trợ tốt hơn về chủ đề (theme). Các widget phổ biến của ttk gồm: ttk.Combobox, ttk.Treeview, ttk.Progressbar, v.v.

* Dòng from tkinter import filedialog.

Import filedialog, một module con của tkinter, cung cấp các hộp thoại để mở hoặc lưu tệp tin.Các hàm chính bao gồm askopenfilename, asksaveasfilename, và askdirectory.

* Dòng import datetime.

Import module datetime, một thư viện chuẩn của Python để làm việc với ngày và giờ. Thường được sử dụng để lấy thời gian hiện tại (datetime.datetime.now()) hoặc xử lý các giá trị ngày giờ.

* Dòng import mysql.connector.

Import module mysql.connector, một thư viện Python dùng để kết nối và làm việc với cơ sở dữ liệu MySQL.Cung cấp các phương pháp để thực thi truy vấn SQL, quản lý kết nối, và xử lý kết quả.

* Dòng from mysql.connector import Error và Import lớp Error từ module mysql.connector.

Lớp này được dùng để xử lý các lỗi liên quan đến kết nối và truy vấn MySQL (ví dụ: mysql.connector.Error).

* Dòng import bcrypt

Import module bcrypt, một thư viện mã hóa (hashing) mật khẩu phổ biến. Bcrypt sử dụng thuật toán mã hóa mạnh để lưu trữ mật khẩu an toàn.Các chức năng chính:

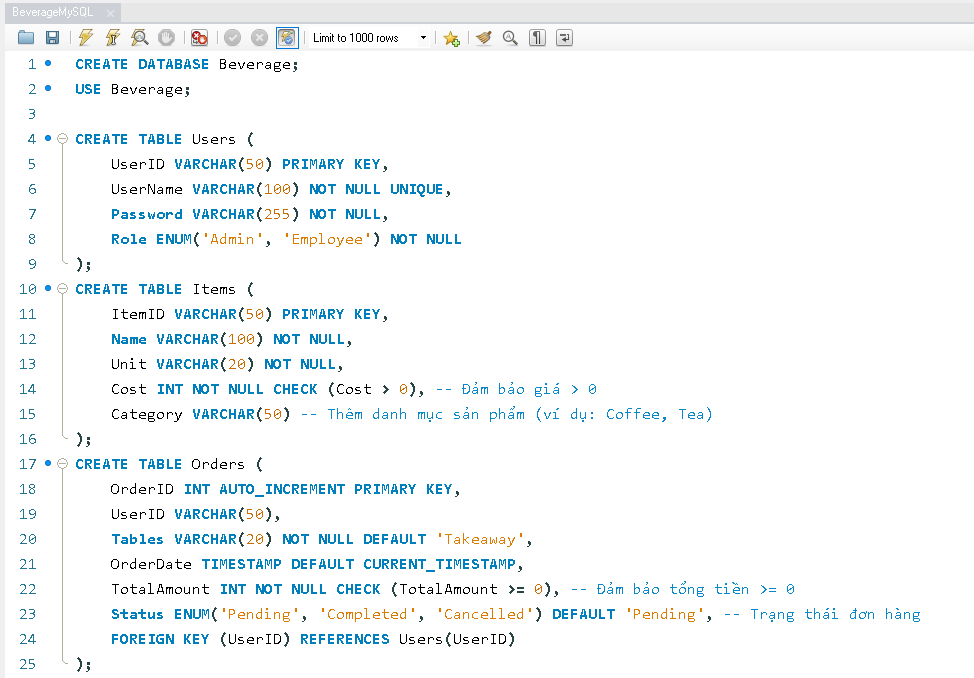
* + bcrypt.hashpw(password, salt): Mã hóa mật khẩu.
  + bcrypt.checkpw(password, hashed\_password): Kiểm tra mật khẩu đầu vào với mật khẩu đã mã hóa.
  + bcrypt.gensalt(): Tạo một salt ngẫu nhiên.

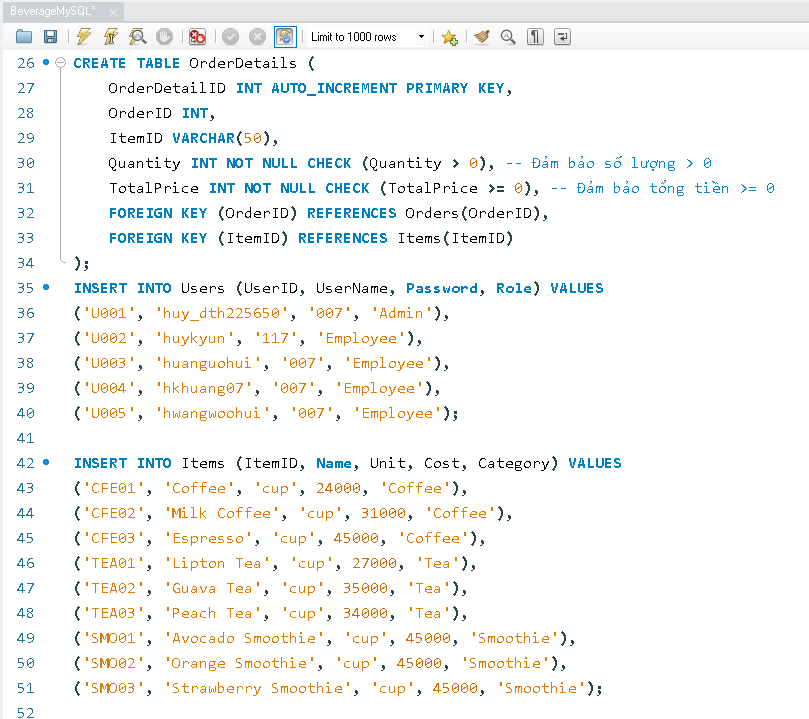
1. **Biến toàn cục**

userID = “” : biến userID rỗng

quantityBill=0, quantityItems=0, revenue=0 : khởi tạo 2 biến số lượng Bill = 0, số lượng Items = 0, giá trị doanh thu = 0

1. **Kết nối Cơ Sở Dữ Liệu**
2. **Database MySQL**

****

****

1. **Hàm kết nối với Database**

Hàm connectDataBase() thực hiện kết nối với cơ sở dữ liệu MySQL và trả về đối tượng kết nối nếu kết nối thành công. Nếu kết nối thất bại, hàm hiển thị thông báo lỗi và trả về None.

* Quy trình

Thiết lập kết nối:

Dùng mysql.connector.connect() để kết nối tới cơ sở dữ liệu MySQL với các thông tin:

Host: "localhost".

User: "root".

Password: "" (không mật khẩu).

Database: "Beverage".

Kiểm tra trạng thái kết nối:

Nếu kết nối thành công (is\_connected()), in thông báo thành công trong console và trả về đối tượng kết nối (connection).

Xử lý lỗi khi kết nối thất bại:

Trong trường hợp có lỗi, hàm:

Hiển thị thông báo lỗi cụ thể qua messagebox.showerror().

Trả về None.

****

1. **Hàm thực thi truy vấn**

Hàm ExecuteQuery(query, params=None) thực hiện việc tương tác với cơ sở dữ liệu MySQL bằng cách kết nối với cơ sở dữ liệu thông qua hàm connectDataBase().Chạy truy vấn SQL (SELECT, INSERT, UPDATE, DELETE, v.v.) với các tham số tùy chọn. Trả về kết quả truy vấn (nếu là SELECT) hoặc thực hiện cam kết thay đổi trong cơ sở dữ liệu (nếu là các truy vấn khác).Đảm bảo đóng kết nối sau khi truy vấn được thực hiện.

* Quy trình:

Thiết lập kết nối:Gọi hàm connectDataBase() để thiết lập kết nối tới cơ sở dữ liệu. Nếu kết nối thất bại (trả về None), hàm dừng và trả về None.

Thực hiện truy vấn:

Tạo một con trỏ (cursor) từ đối tượng kết nối. Sử dụng cursor.execute() để thực hiện câu lệnh SQL với tham số (params) nếu có.

Phân loại xử lý dựa trên loại truy vấn:

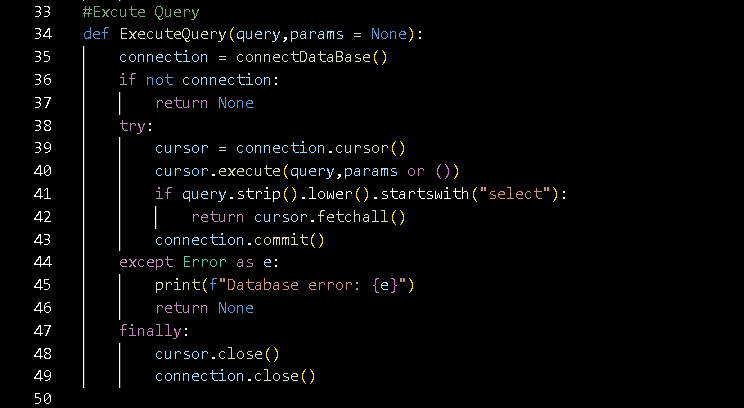
Nếu truy vấn bắt đầu bằng SELECT:Trả về tất cả kết quả truy vấn qua cursor.fetchall().

Các truy vấn khác (INSERT, UPDATE, DELETE): Sử dụng connection.commit() để lưu các thay đổi.

Xử lý lỗi:

Bắt lỗi thông qua khối try-except.In thông báo lỗi cụ thể khi có lỗi truy vấn.Đóng tài nguyên:

Đảm bảo con trỏ (cursor) và kết nối (connection) được đóng sau khi xử lý xong bằng cách sử dụng khối finally.

****

1. **Đăng Nhập**

Hàm LogInWindow tạo giao diện đăng nhập và các chức năng chính sau:

* Tạo giao diện đăng nhập

Cửa sổ chính (loginWin):

Tiêu đề: LOGIN.

Kích thước: 320x200.

Màu nền: #AFEEEE.

Vị trí được căn giữa bằng hàm makecenter(loginWin).

Các thành phần giao diện:

Nhãn (Label):

"LOG IN" (tiêu đề chính).

"User name:" và "Password:" (hướng dẫn nhập liệu).

Ô nhập liệu (Entry):

txtUserName: Nhập tên người dùng.

txtUserPass: Nhập mật khẩu (ẩn bằng ký tự \*).

Con trỏ tự động: Đặt ở ô nhập tên ban đầu (txtUserName.focus()).

Nút bấm (Button):

Login: Xác thực thông tin người dùng.

Exit: Thoát ứng dụng.

Sign up: Mở cửa sổ đăng ký tài khoản mới.

Change Password: Mở cửa sổ đổi mật khẩu.

* Xác thực thông tin đăng nhập

Hàm HandleLogin():

Lấy thông tin từ ô nhập liệu (username, password).

Kiểm tra xem các trường nhập có bị bỏ trống.

Truy vấn cơ sở dữ liệu để lấy thông tin người dùng từ bảng Users:

SELECT UserID, Password, Role theo tên người dùng.

So sánh mật khẩu nhập vào với mật khẩu đã mã hóa trong cơ sở dữ liệu bằng bcrypt.checkpw().

Nếu thông tin hợp lệ:

Hiển thị thông báo chào mừng.

Mở cửa sổ chính (MainWindow()).

Lưu lại UserID của người dùng vào biến toàn cục userID.

Nếu thông tin không hợp lệ:

Hiển thị thông báo lỗi.

* Thoát ứng dụng

Hàm ApplicationExit():

Hiển thị hộp thoại xác nhận thoát ứng dụng.

Nếu người dùng đồng ý, cửa sổ sẽ đóng.

* Tích hợp sự kiện

Khi nhấn Enter:

Trong ô tên người dùng: Chuyển sang ô mật khẩu.

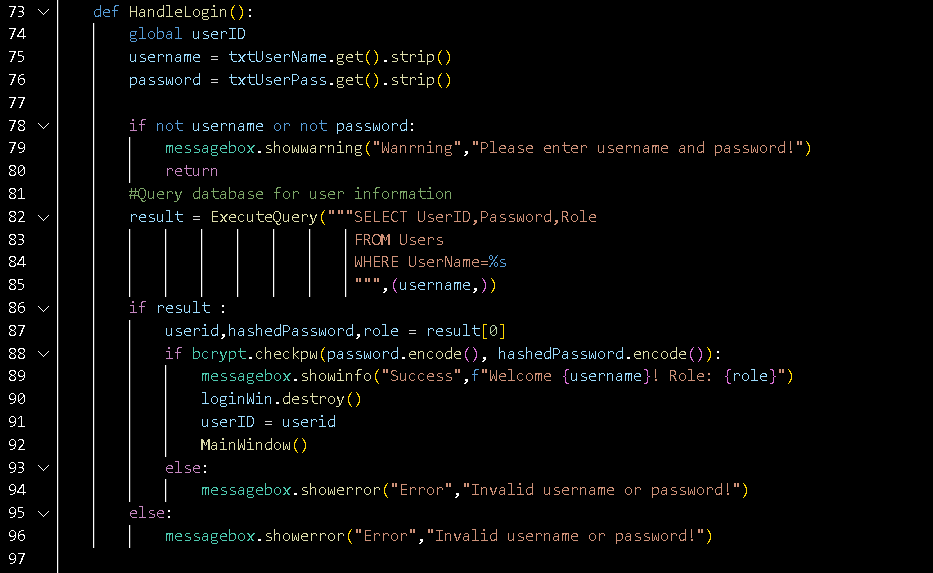
Trong ô mật khẩu: Thực hiện kiểm tra đăng nhập.

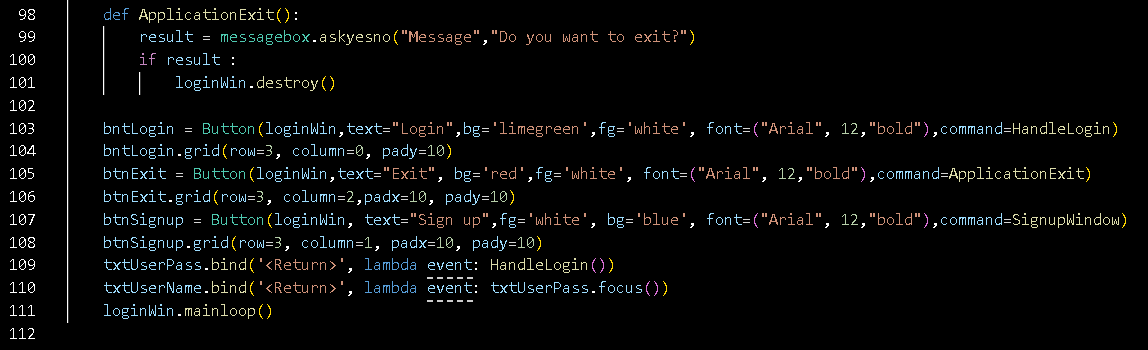
* Các nút hỗ trợ

Sign up:Mở cửa sổ đăng ký tài khoản (SignupWindow()).

Change Password:Mở cửa sổ đổi mật khẩu (ChangePassword()).

****

****

****

1. **Đăng Ký**

Hàm SignupWindow() tạo giao diện đăng ký tài khoản mới với các chức năng chính sau:

1.Tạo giao diện đăng ký

Cửa sổ chính (signupWin):

Tiêu đề: SIGN UP ACCOUNT.

Kích thước: 320x200.

Màu nền: #AFEEEE.

Cho phép thay đổi kích thước.

Đặt cửa sổ luôn ở phía trên cùng (attributes("-topmost", 1)) và căn giữa qua hàm makecenter().

Các thành phần giao diện:

Nhãn (Label):

Hiển thị hướng dẫn nhập User ID, User name, Password, và chọn Role.

Ô nhập liệu (Entry):

txtIDSignUp: Nhập User ID.

txtUserSignUp: Nhập User name.

txtPasSignUp: Nhập Password.

Nút chọn (Checkbox):

chkAdmin: Chọn vai trò Admin.

chkEmployee: Chọn vai trò Employee.

Nút bấm (Button):

Create Account: Xử lý tạo tài khoản mới.

Exit: Thoát cửa sổ đăng ký.

2. Chức năng chính

Tạo tài khoản mới (CreateAccount)

Thu thập thông tin người dùng:

User ID, User name, Password, và vai trò (Role).

Kiểm tra dữ liệu đầu vào:

Tất cả các trường đều phải được điền.

Người dùng phải chọn một vai trò duy nhất.

Xác thực:

Kiểm tra tên người dùng có tồn tại hay chưa trong cơ sở dữ liệu.

Nếu đã tồn tại, hiển thị thông báo lỗi.

Mã hóa mật khẩu:

Mã hóa mật khẩu bằng bcrypt với hàm HashPassword():

Tạo Salt và mã hóa mật khẩu.

Lưu thông tin vào cơ sở dữ liệu:

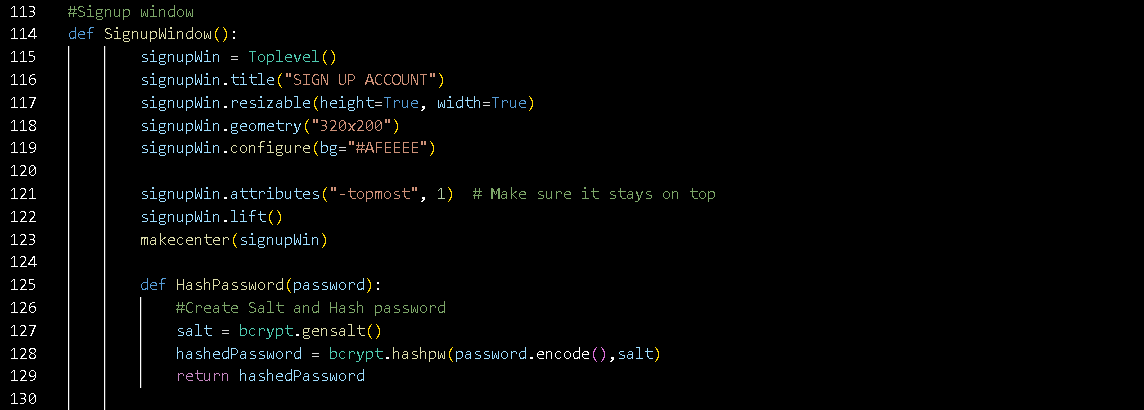
Chèn thông tin người dùng mới vào bảng Users:

INSERT INTO Users (UserID, UserName, Password, Role).

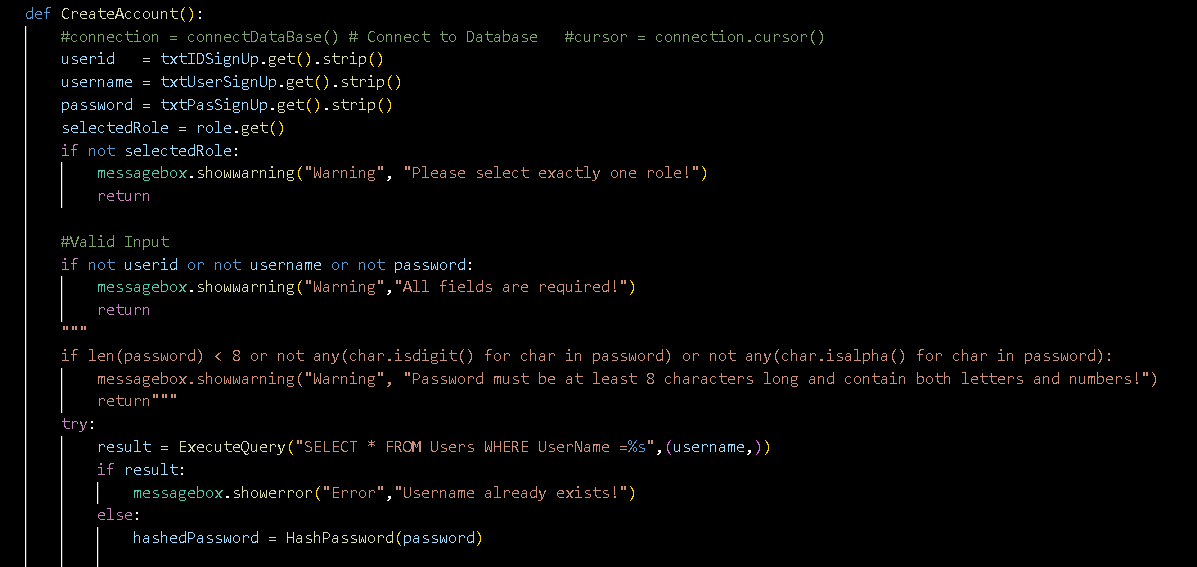
Thông báo:

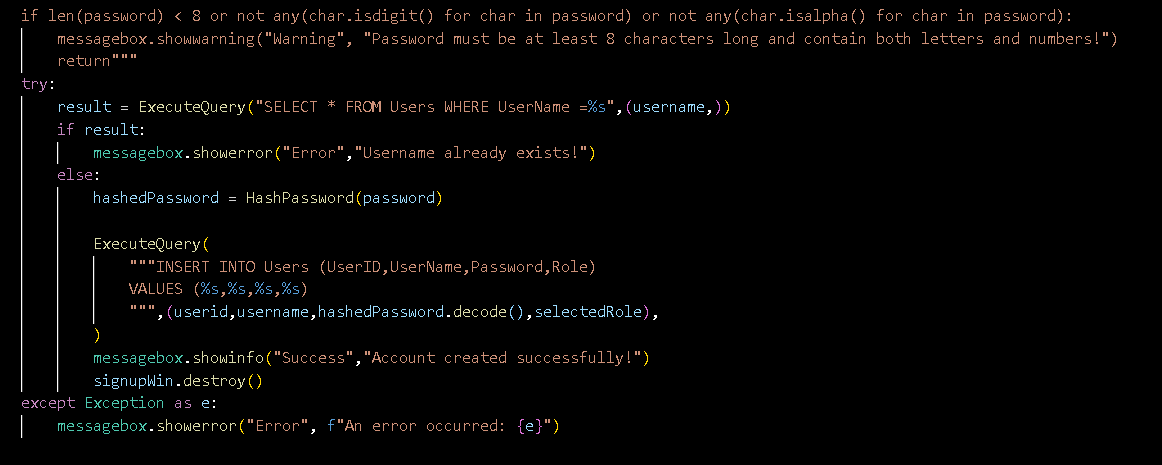
Nếu thành công: Hiển thị thông báo tài khoản được tạo và đóng cửa sổ.

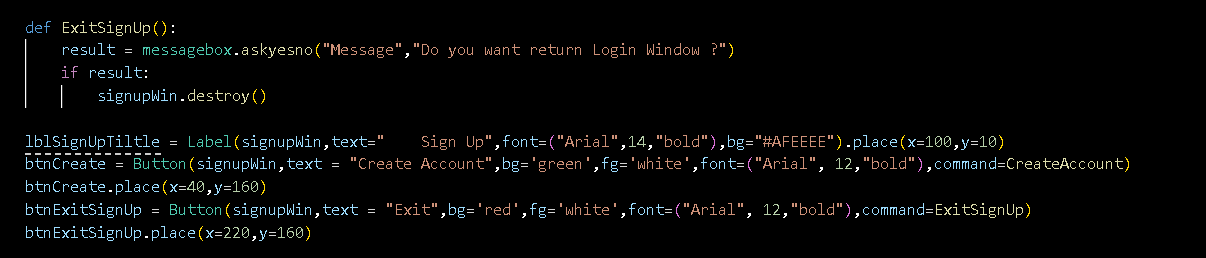
Nếu thất bại: Hiển thị thông báo lỗi.

****

****

****

****

****

1. **Đổi Mật Khẩu**

Hàm ChangePassword() tạo giao diện và xử lý việc thay đổi mật khẩu của người dùng với các bước cụ thể như sau:

1. Tạo giao diện

Cửa sổ chính (changePassWin):

Tiêu đề: Change Password.

Kích thước: 320x240.

Màu nền: #AFEEEE.

Sử dụng hàm makecenter() để căn giữa cửa sổ.

Các thành phần giao diện:

Nhãn (Label):

Hiển thị hướng dẫn nhập các thông tin cần thiết: User Name, Current Password, New Password, và Confirm Password.

Ô nhập liệu (Entry):

txtUserName: Nhập User Name.

txtCurrentPassword: Nhập mật khẩu hiện tại (dạng ẩn ký tự \*).

txtNewPassword: Nhập mật khẩu mới.

txtConfirmPassword: Nhập lại mật khẩu mới để xác nhận.

Nút bấm (Button):

Change Password: Gọi hàm xử lý thay đổi mật khẩu.

Exit: Đóng cửa sổ thay đổi mật khẩu.

2. Xử lý chính

Thay đổi mật khẩu (HandleChangePassword)

Thu thập thông tin người dùng:

Lấy thông tin từ các ô nhập liệu: userName, currentPassword, newPassword, và confirmPassword.

Kiểm tra hợp lệ dữ liệu:

Tất cả các trường phải được điền.

Mật khẩu mới và mật khẩu xác nhận phải khớp nhau.

Xác thực mật khẩu hiện tại:

Truy vấn cơ sở dữ liệu để lấy mật khẩu mã hóa của người dùng.

Dùng bcrypt để kiểm tra mật khẩu hiện tại có đúng không.

Cập nhật mật khẩu mới:

Nếu mật khẩu hiện tại đúng:

Mã hóa mật khẩu mới bằng bcrypt.

Cập nhật mật khẩu mới vào cơ sở dữ liệu với câu lệnh UPDATE Users.

Hiển thị thông báo thành công và đóng cửa sổ.

Nếu sai:

Hiển thị thông báo lỗi.

Xử lý lỗi:

Sử dụng try-except để bắt và thông báo lỗi trong quá trình truy vấn hoặc cập nhật cơ sở dữ liệu.



****

1. **Hàm Giao Diện Người Dùng**
2. **Hàm Làm xuất hiện chính giữa màn hình**

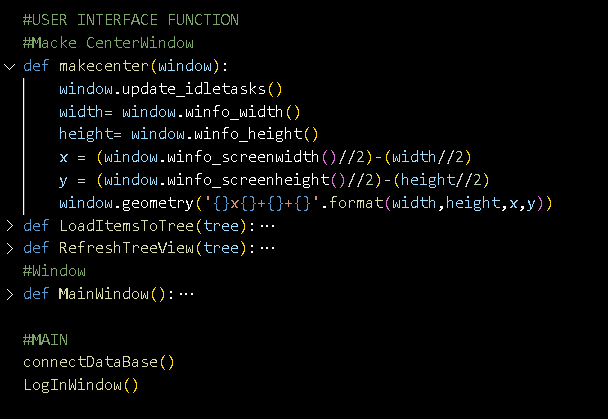
Chức năng:

Căn giữa cửa sổ window trên màn hình.

Chi tiết hoạt động:

Tính toán vị trí x, y dựa trên kích thước cửa sổ (width, height) và độ phân giải màn hình (screenwidth, screenheight).

Dùng phương thức geometry() để đặt lại tọa độ của cửa sổ.

****

1. **Hàm Lấy Dữ liệu vào TreeView**

Chức năng:

Kết nối cơ sở dữ liệu để tải dữ liệu từ bảng Items vào TreeView.

Chi tiết hoạt động:

Kết nối cơ sở dữ liệu:

Gọi hàm connectDataBase().

Hiển thị thông báo lỗi nếu không kết nối được.

Truy vấn dữ liệu:

Truy vấn các trường ItemID, Name, Unit, Cost, Category từ bảng Items với điều kiện Cost > 0.

Xóa dữ liệu cũ trong TreeView:

Xóa các dòng hiện tại trong tree bằng vòng lặp for.

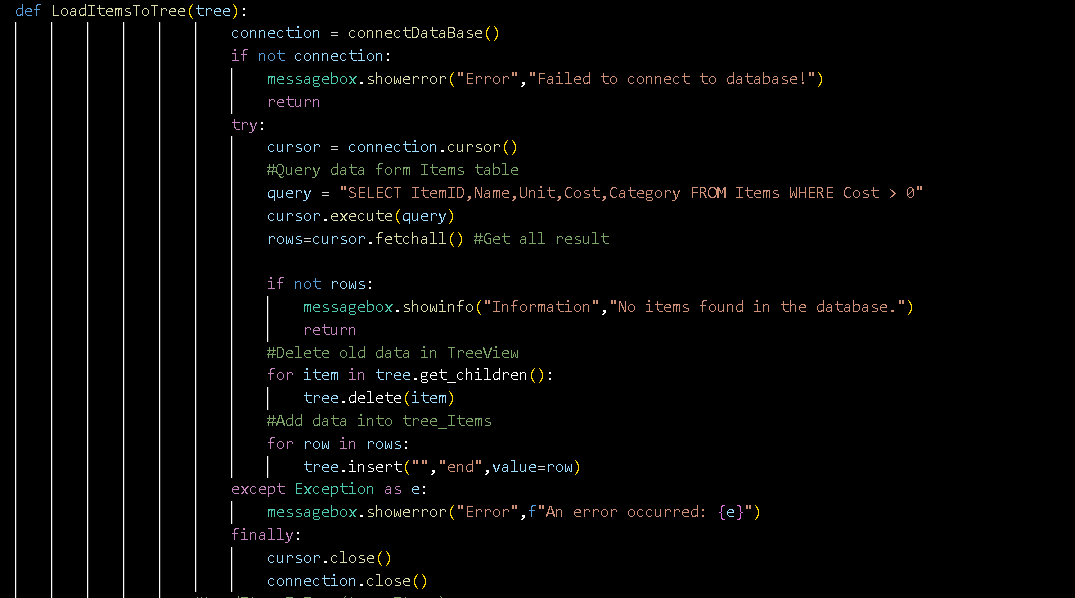
Thêm dữ liệu mới vào TreeView:

Chèn từng dòng dữ liệu lấy từ cơ sở dữ liệu vào TreeView qua phương thức insert().

Xử lý lỗi và đóng kết nối:

Hiển thị thông báo lỗi nếu có ngoại lệ xảy ra.

Đảm bảo đóng kết nối cơ sở dữ liệu bằng khối finally.

****

1. **Hàm Làm mới dữ liệu**

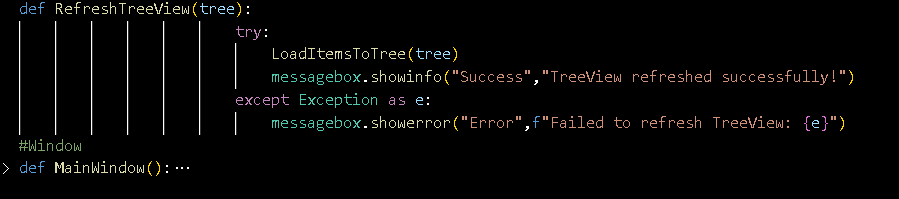
Chức năng:

Làm mới giao diện TreeView.

Hành vi mong đợi:

Gọi lại hàm LoadItemsToTree(tree) để tải lại dữ liệu từ cơ sở dữ liệu vào TreeView.

Xóa mọi lựa chọn hiện tại trong TreeView.

****

1. **Giao diện chính**

Hàm MainWindow() tạo ra giao diện chính của phần mềm order nước giải khát, bao gồm cửa sổ chính, các tab chức năng, và một số thiết lập ban đầu.

* Cửa sổ chính:tạo cửa sổ chính với màu nền sáng, kích thước cố định, và được căn giữa.
* Giao diện TreeView:tùy chỉnh màu sắc, chiều cao dòng, và hành vi khi chọn.
* Tab Menu: tạo 3 tab chính: Layout, Order, và Utilities.

****

1. **Tab Order**

* Xây dựng giao diện tab Order trong ứng dụng phần mềm bán nước giải khát. Giao diện này bao gồm các vùng hiển thị dữ liệu, các ô nhập liệu, và các nút chức năng liên quan đến quản lý đặt hàng.
* Giao diện tab Order được thiết kế trực quan với 3 khu vực chính:

Menu sản phẩm.

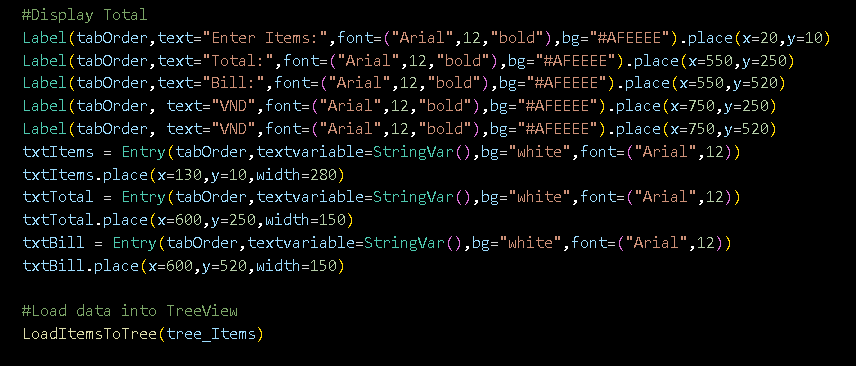
Chi tiết các mặt hàng đã thêm vào đơn hàng.

Thông tin tổng quan về đơn hàng.

* Cung cấp các nút chức năng đầy đủ như: thêm, tìm kiếm, xóa, đặt hàng, thanh toán, và làm mới danh sách.
* Bố cục hợp lý và các nhãn giải thích rõ ràng giúp người dùng dễ sử dụng.

****

****

****

****

****

1. **Tab Sơ đồ:**

* Xây dựng giao diện Tab Layout trong ứng dụng, nơi người dùng có thể chọn bàn hoặc chế độ Takeaway khi đặt hàng.
* Tab Layout cung cấp giao diện chọn bàn dễ sử dụng:

Hiển thị danh sách các bàn bằng các nút bấm.

Chọn bàn sẽ cập nhật biến trạng thái selectedTable và hiển thị thông báo xác nhận.

* Thiết kế giao diện:

Gọn gàng với bố cục dạng lưới (tối đa 9 nút mỗi hàng).

Màu sắc thân thiện, dễ nhìn.

****

1. **Tab Tiện Ích**

* Xây dựng giao diện cho Tab Utilities trong phần mềm, với ba chức năng chính: thống kê, quản lý hóa đơn, và tiện ích khác. Tab này giúp người dùng kiểm soát và quản lý các hoạt động liên quan đến hóa đơn và cấu hình hệ thống.
* Tab Utilities là nơi cung cấp các chức năng quản lý và tiện ích hỗ trợ:

Thống kê: Hiển thị số liệu tổng quan như số lượng hóa đơn, sản phẩm, và doanh thu.

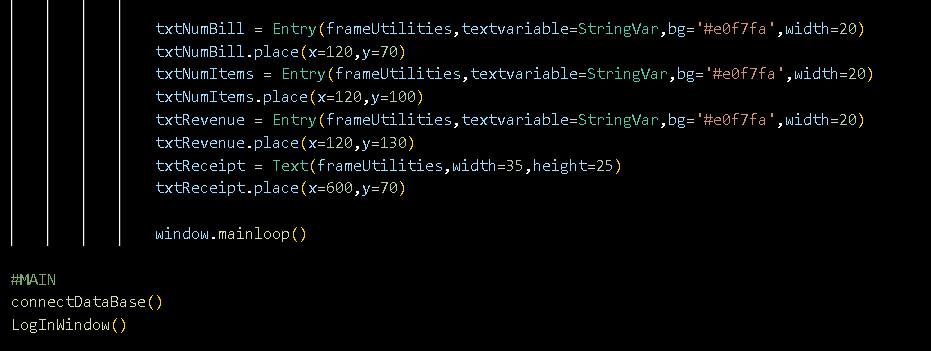
Quản lý hóa đơn: Cho phép xem và lưu trữ hóa đơn dễ dàng.

Tiện ích khác: Bao gồm cấu hình sản phẩm và thoát chương trình.

* Giao diện:

Màu sắc hài hòa, bố cục rõ ràng với các chức năng được phân chia hợp lý.

****

****

1. **Binding Event**

Hàm TreeOrderSelectedIndexChange được sử dụng để xử lý sự thay đổi khi người dùng chọn một mục trong bảng tree\_Order, đồng thời cập nhật thông tin hiển thị trong txtBill với tổng giá trị của đơn hàng tương ứng. Các sự kiện <<TreeviewSelect>> và các sự kiện <Return> trên các đối tượng khác cũng được gắn kết để xử lý hành động của người dùng.

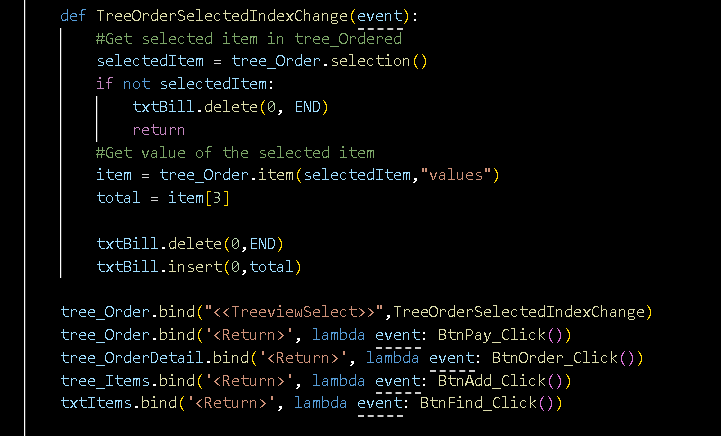
* Quy trình

Hàm TreeOrderSelectedIndexChange:

Khi người dùng chọn một mục trong tree\_Order, hàm này sẽ được gọi để lấy tổng giá trị của đơn hàng và hiển thị trong txtBill.

Các sự kiện <<TreeviewSelect>> và <Return>:

Các sự kiện được gắn kết với các đối tượng tree\_Order, tree\_OrderDetail, tree\_Items, và txtItems để xử lý hành động của người dùng khi nhấn Enter hoặc chọn mục trong các bảng.

****

1. **Chức Năng:**
2. **Thêm:**

Hàm BtnAdd\_Click thực hiện chức năng thêm sản phẩm vào đơn hàng hiện tại (hoặc tạo một đơn hàng mới nếu chưa có) và cập nhật các chi tiết đơn hàng vào cơ sở dữ liệu và giao diện người dùng.

* Quy trình:

Kết nối cơ sở dữ liệu.

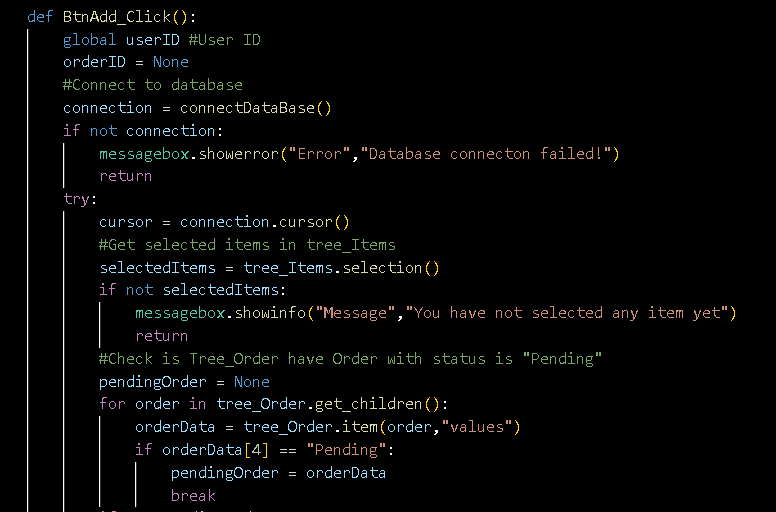
Kiểm tra và tạo đơn hàng mới nếu không có đơn hàng với trạng thái "Pending".

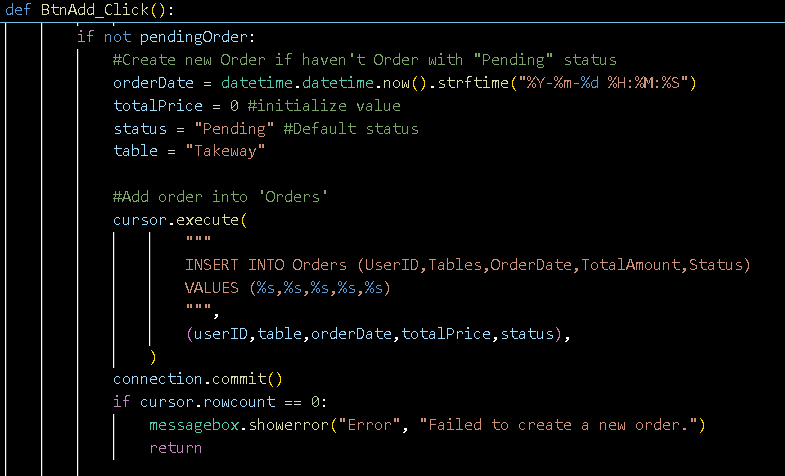
Thêm hoặc cập nhật sản phẩm vào chi tiết đơn hàng.

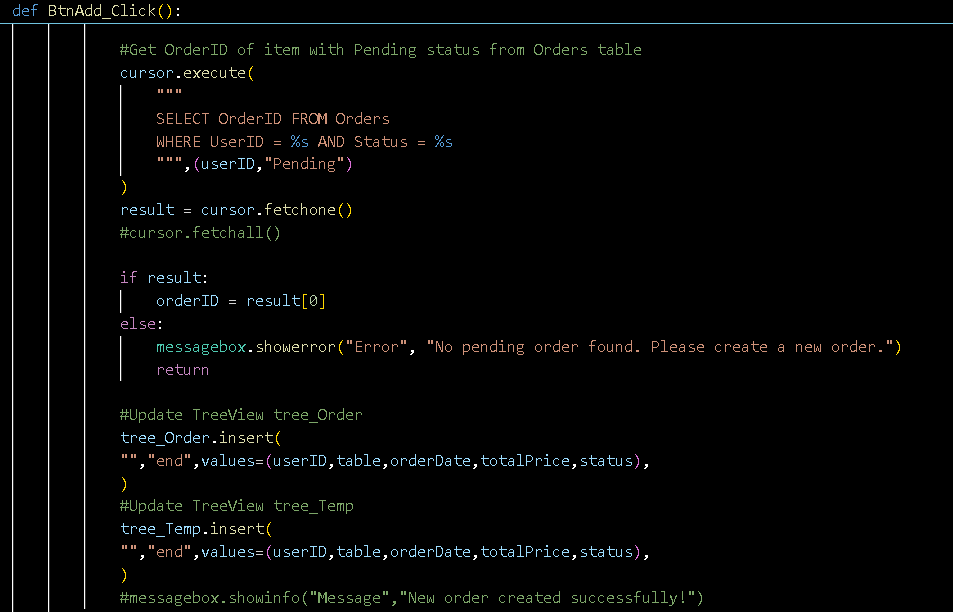
Tính toán lại tổng giá trị của đơn hàng và cập nhật vào cơ sở dữ liệu.

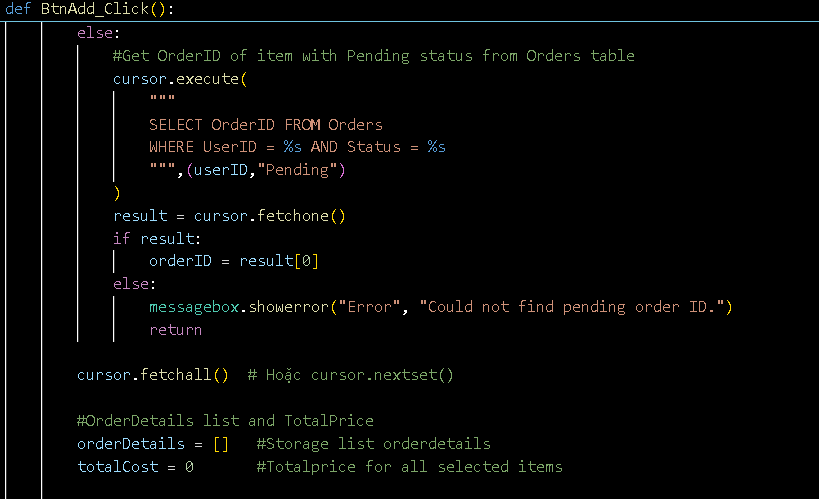
Cập nhật giao diện người dùng với thông tin mới.

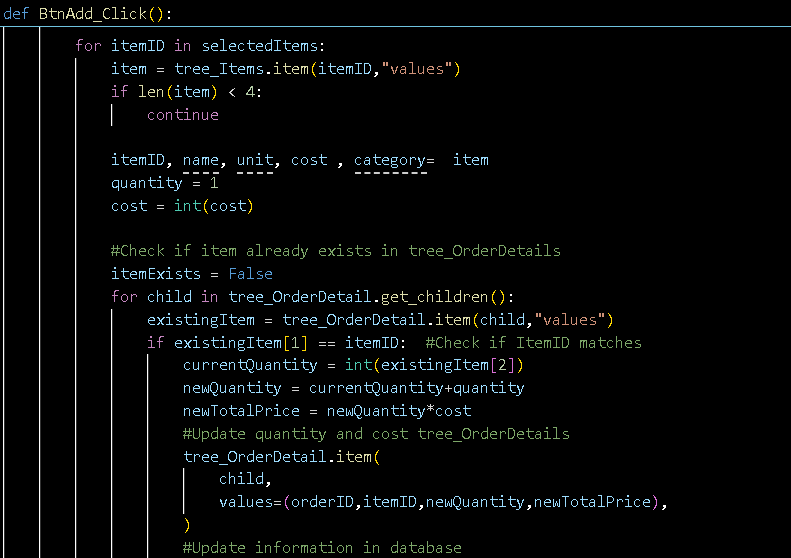
Xử lý lỗi và đóng kết nối.

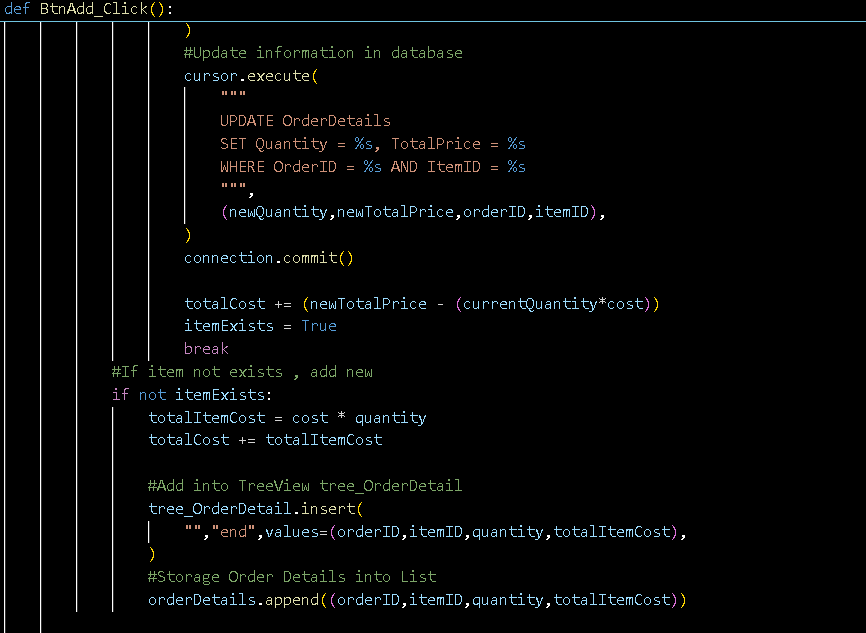
****

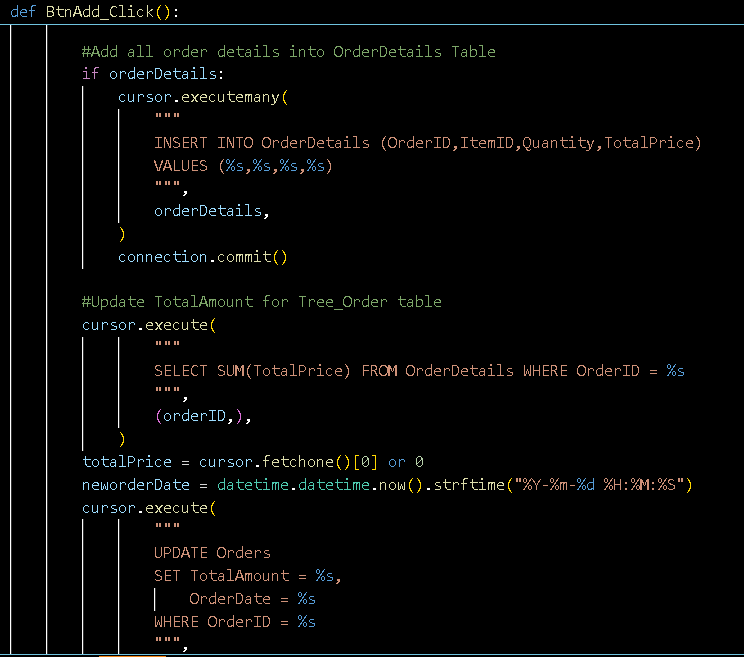
****

****

****

****

****

****

****

****

1. **Xóa**

Hàm BtnDelete\_Click thực hiện chức năng xóa các mục (sản phẩm) đã chọn trong chi tiết đơn hàng (OrderDetails) và cập nhật cơ sở dữ liệu cùng với giao diện người dùng.

* Quy trình:

Kiểm tra mục được chọn trong tree\_OrderDetail.

Xác nhận xóa với người dùng.

Chuẩn bị và kiểm tra dữ liệu cần xóa.

Thực hiện xóa trong cơ sở dữ liệu.

Cập nhật giao diện người dùng (xóa mục trong tree\_OrderDetail).

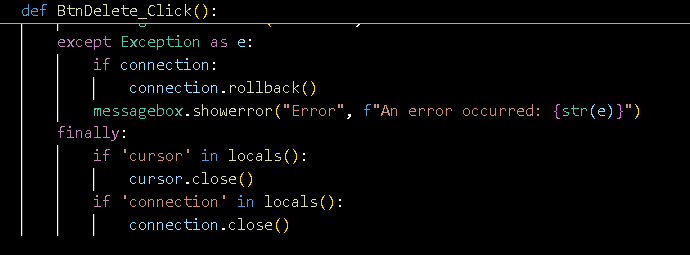
Cập nhật tổng giá trị đơn hàng.

Cập nhật các biến toàn cục (giảm quantityItems và quantityBill).

Xử lý lỗi và đóng kết nối cơ sở dữ liệu.

****

****

****

1. **Hủy**

Hàm BtnCancel\_Click thực hiện chức năng hủy tất cả các đơn hàng (Order) có trạng thái "Pending", xóa chúng khỏi cơ sở dữ liệu và cập nhật giao diện người dùng.

* Quy trình:

Kiểm tra mục được chọn trong tree\_OrderDetail.

Xác nhận hủy đơn hàng với người dùng.

Kết nối cơ sở dữ liệu và lấy OrderID của đơn hàng có trạng thái "Pending".

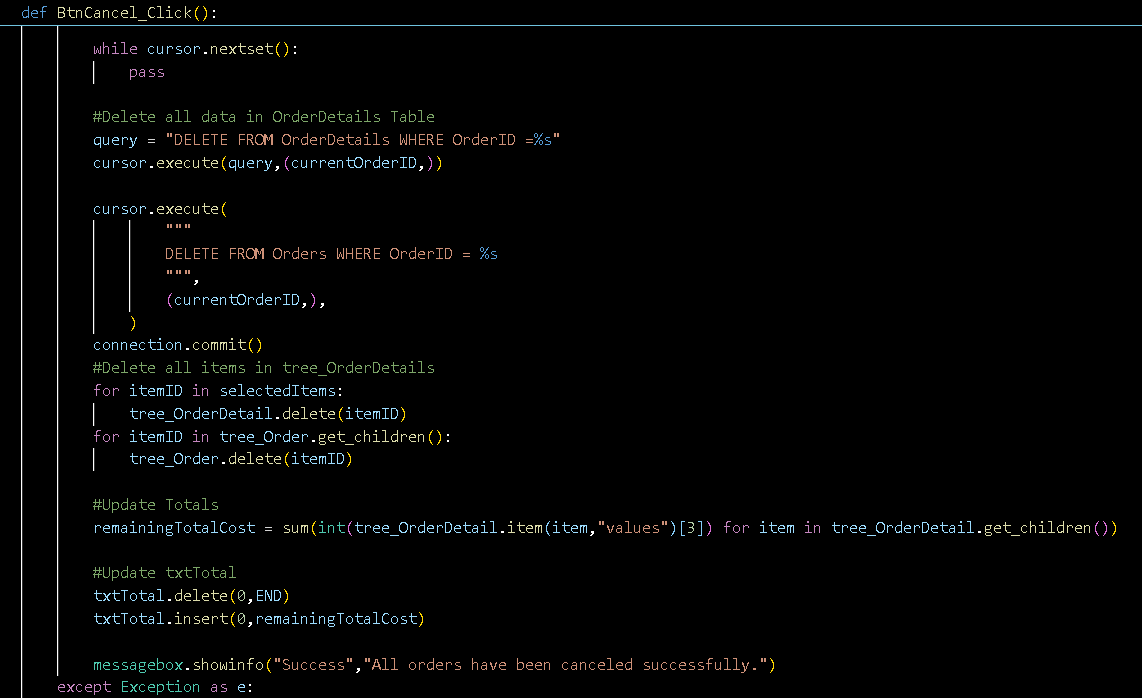
Xóa dữ liệu khỏi cơ sở dữ liệu: xóa chi tiết đơn hàng và đơn hàng trong bảng OrderDetails và Orders.

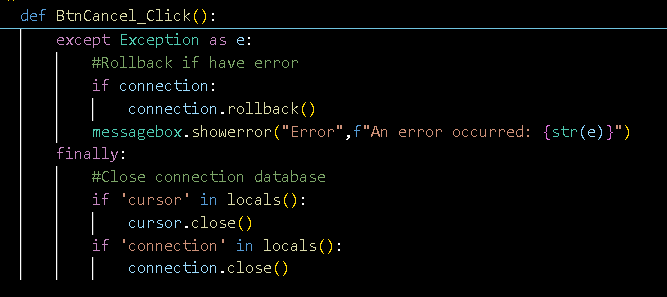
Cập nhật giao diện người dùng (xóa các mục trong tree\_OrderDetail và tree\_Order).

Cập nhật tổng giá trị và hiển thị lại vào txtTotal.

Xử lý lỗi và đóng kết nối cơ sở dữ liệu.

****

****

****

1. **Order**

Hàm BtnOrder\_Click xử lý việc hoàn tất một đơn hàng, bao gồm việc xác nhận và cập nhật đơn hàng từ "Pending" sang "Completed", cập nhật cơ sở dữ liệu và giao diện người dùng, và xóa các mục đã chọn từ tree\_OrderDetail và tree\_Items.

* Quy trình:

Kiểm tra thông tin bàn đã chọn và xóa hóa đơn cũ nếu có.

Kết nối với cơ sở dữ liệu và kiểm tra xem có đơn hàng nào cần xử lý không.

Lọc đơn hàng "Pending", lấy thời gian hiện tại và cập nhật TotalAmount, Tables, và OrderDate.

Cập nhật trạng thái từ "Pending" thành "Completed" cho các đơn hàng.

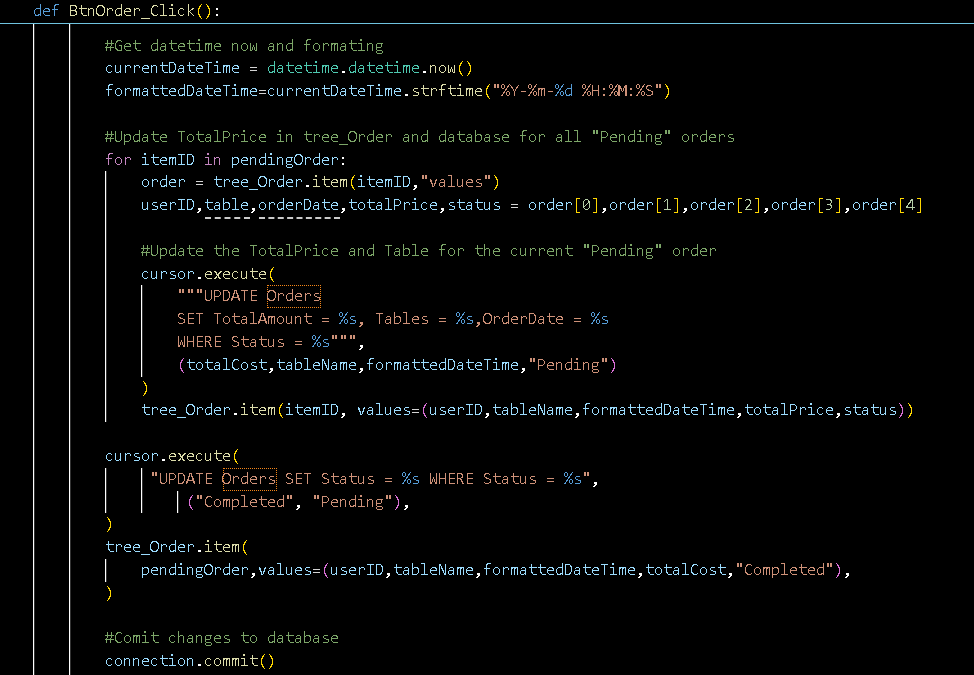
Cập nhật thông tin giao diện người dùng (số lượng hóa đơn, tổng số mục, xóa các mục trong tree\_OrderDetail).

Xóa lựa chọn trong tree\_OrderDetail và tree\_Items.

Commit cơ sở dữ liệu và đóng kết nối.

Xử lý lỗi và thực hiện rollback nếu có lỗi.

****

****

****

1. **Thanh Toán**

Hàm BtnFind\_Click được sử dụng để tìm kiếm các mục (items) trong bảng tree\_Items dựa trên tên, mã ID, đơn vị, giá hoặc loại của mục. Nó thực hiện tìm kiếm dựa trên từ khóa được nhập vào trong txtItems và hiển thị kết quả tìm kiếm cho người dùng.

* Quy trình:

Lấy từ khóa tìm kiếm từ txtItems và kiểm tra xem có nhập hay không.

Tìm kiếm trong bảng tree\_Items bằng cách kiểm tra xem từ khóa có xuất hiện trong bất kỳ trường thông tin nào (ID, tên, đơn vị, giá, hoặc loại).

Nếu có mục khớp:

Xóa lựa chọn cũ trong tree\_Items.

Chọn và hiển thị các mục khớp.

Cập nhật thông báo cho người dùng về số lượng mục tìm thấy.

Nếu không có mục khớp, thông báo lỗi cho người dùng.

****

****

****

****

1. **Thêm, Xóa Sửa Danh Sách Đồ Uống**

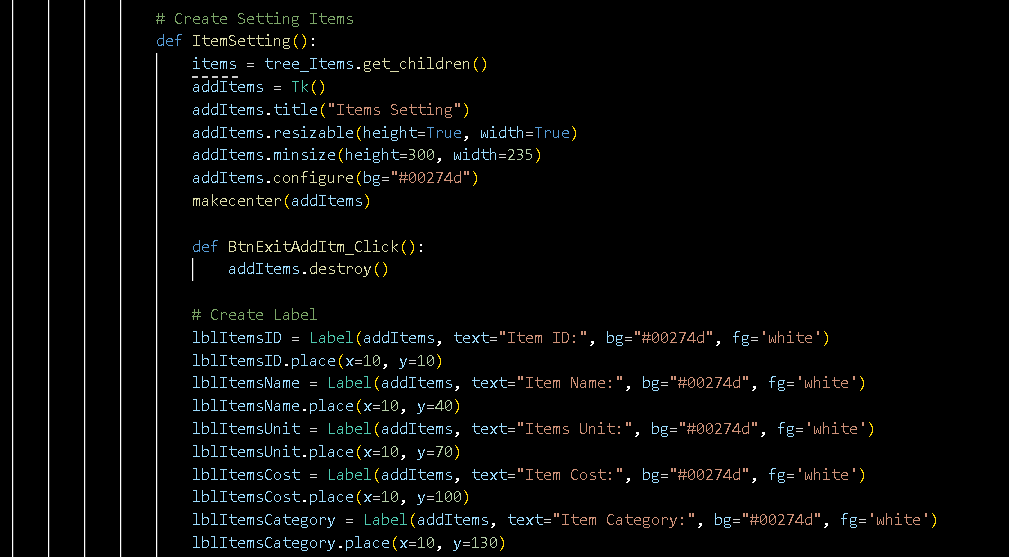
Hàm ItemSetting mở một cửa sổ Toplevel mới cho phép người dùng thực hiện các thao tác như thêm, xóa, chỉnh sửa thông tin mặt hàng trong hệ thống. Cửa sổ này bao gồm các trường nhập liệu để nhập thông tin cho mặt hàng, đồng thời cung cấp các nút để thực hiện các thao tác trên các mặt hàng trong cơ sở dữ liệu.

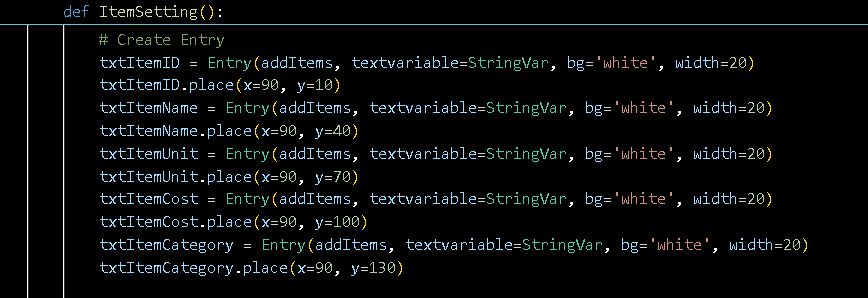
* Quy trình:

Khi người dùng muốn thêm, chỉnh sửa, hoặc xóa một mặt hàng, cửa sổ ItemSetting sẽ mở ra, cung cấp các trường nhập liệu và các nút thao tác.

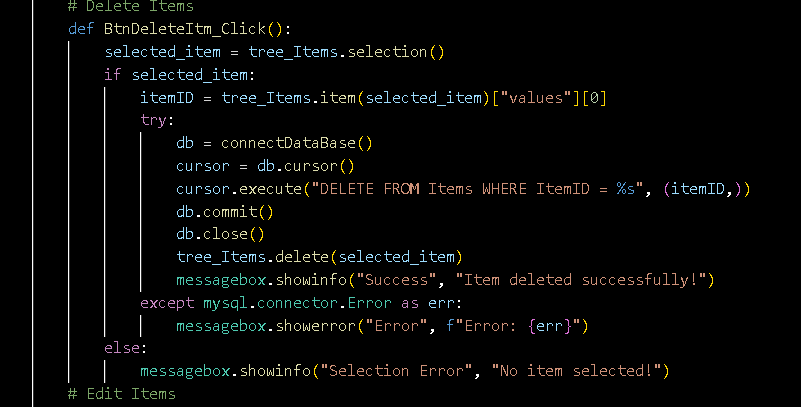
Các thao tác sẽ được thực hiện trên cơ sở dữ liệu thông qua các câu lệnh SQL, và bảng tree\_Items sẽ được cập nhật để phản ánh các thay đổi.

Người dùng có thể thực hiện các thay đổi liên quan đến mặt hàng và sẽ nhận được thông báo về kết quả của thao tác.

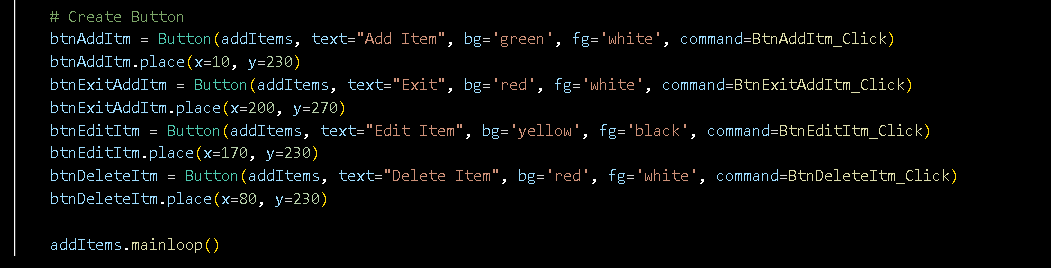
****

****

****

****

****

****

1. **Tìm Kiếm**

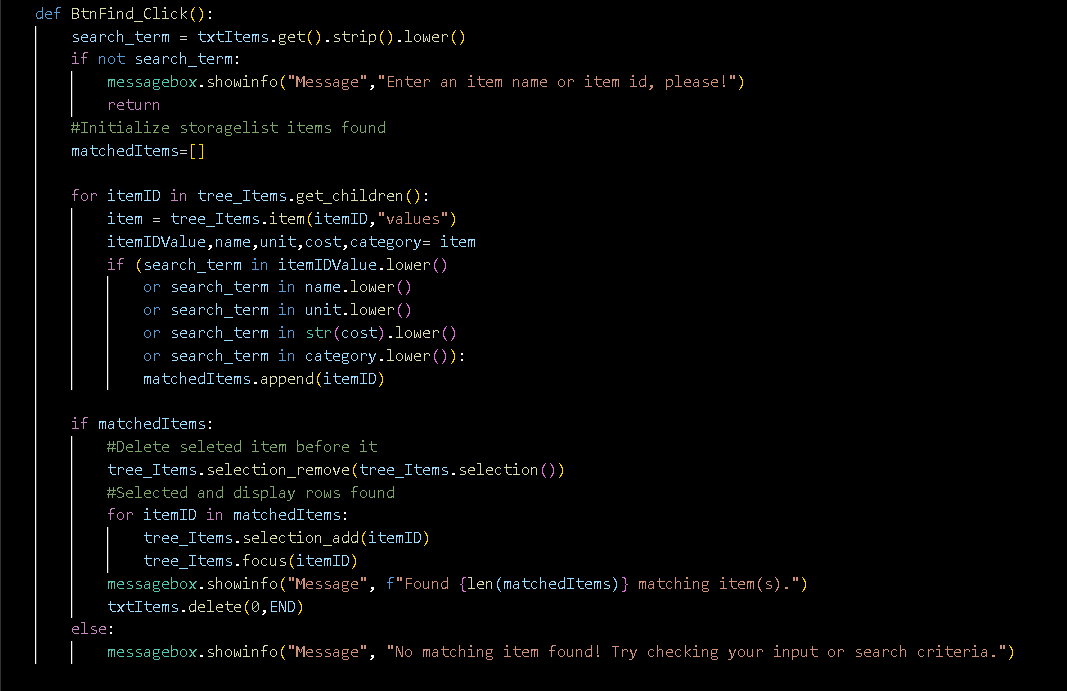
Hàm BtnFind\_Click thực hiện chức năng tìm kiếm một mặt hàng trong bảng tree\_Items dựa trên từ khóa người dùng nhập vào. Từ khóa có thể là Item ID, Tên mặt hàng, Đơn vị, Giá trị hoặc Danh mục của mặt hàng.

* Quy trình

Người dùng nhập từ khóa vào trường tìm kiếm.

Hệ thống sẽ kiểm tra và tìm kiếm các mặt hàng trong bảng tree\_Items có thông tin khớp với từ khóa.

Nếu tìm thấy, các mặt hàng phù hợp sẽ được chọn và hiển thị trong bảng. Nếu không, thông báo sẽ được hiển thị để người dùng biết kết quả tìm kiếm

****

1. **Lưu Hóa Đơn**

Hàm SaveBill() thực hiện chức năng lưu hóa đơn thành file văn bản (.txt), cung cấp cho người dùng tùy chọn vị trí lưu file qua hộp thoại.

* Quy trình:

Tạo tên file hóa đơn tự động:

Lấy thời gian hiện tại bằng: datetime.datetime.now().

Tạo tên file theo định dạng: "Bill\_YYYYMMDD\_HHMMSS.txt", trong đó YYYYMMDD là ngày và HHMMSS là giờ phút giây hiện tại.

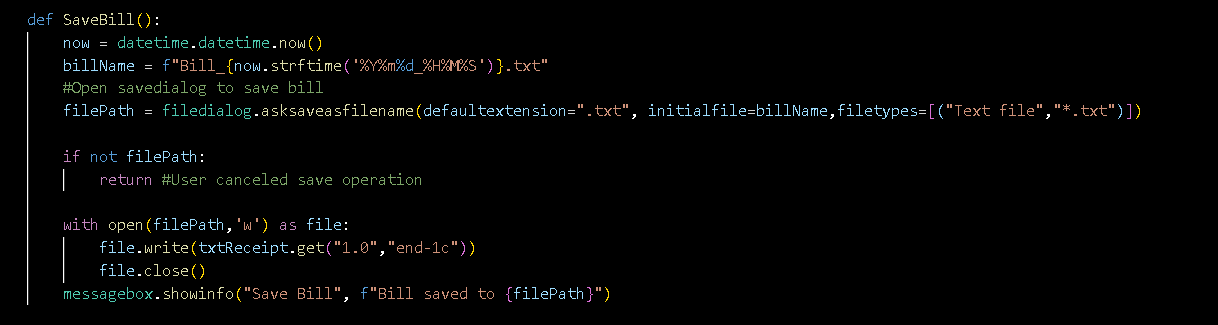
Hiển thị hộp thoại lưu file:Sử dụng filedialog.asksaveasfilename() để hiển thị hộp thoại, cho phép người dùng chọn vị trí và tên file.

Mặc định:Định dạng file: .txt.

Tên file ban đầu: tên hóa đơn được tạo tự động. Nếu người dùng hủy thao tác (không chọn vị trí hoặc tên file), hàm sẽ thoát mà không thực hiện thêm hành động nào.

Ghi nội dung hóa đơn vào file:Mở file ở chế độ ghi ('w') tại đường dẫn mà người dùng chọn (filePath). Lấy nội dung từ txtReceipt (một widget kiểu Text) và ghi vào file. Đóng file sau khi ghi xong.

Thông báo kết quả:Nếu lưu thành công, hiển thị một hộp thoại thông báo rằng hóa đơn đã được lưu, kèm theo đường dẫn nơi file được lưu.

****

1. **Lưu Thống Kê Doanh Thu**

Hàm SaveReceipt() thực hiện chức năng lưu biên lai dưới dạng file văn bản (.txt) với nội dung được lấy từ danh sách các mục trong tree\_Temp, đồng thời ghi nhận tổng doanh thu.

* Quy trình

Tạo tên file biên lai tự động:Sử dụng datetime.datetime.now() để lấy thời gian hiện tại. Đặt tên file theo định dạng "Receipt\_YYYYMMDD\_HHMMSS.txt", trong đó YYYYMMDD là ngày và HHMMSS là giờ phút giây.

Hiển thị hộp thoại lưu file:Sử dụng filedialog.asksaveasfilename() để hiển thị hộp thoại, cho phép người dùng chọn vị trí lưu file.

Mặc định:Định dạng file: .txt.Tên file ban đầu: tên biên lai tự động.Nếu người dùng hủy thao tác (không chọn vị trí), hàm sẽ thoát.

Ghi nội dung biên lai vào file:

Mở file ở chế độ ghi ('w') tại đường dẫn được chọn (filePath).

Ghi nội dung bao gồm:

Tiêu đề: "Drink Order Receipt".

Ngày giờ: Lấy từ thời gian hiện tại.

Danh sách chi tiết: Lấy thông tin từ tree\_Temp:

Gồm các trường như UserID, Table, Date, TotalPrice, Status.

Tổng doanh thu: Lấy từ giá trị trong txtRevenue.

Kết thúc và thông báo:

Sau khi ghi xong, đóng file.

Hiển thị thông báo rằng biên lai đã được lưu thành công, kèm theo đường dẫn.

****

1. **Xem Hóa Đơn, Doanh Thu**

Hàm OpenReceipt() thực hiện chức năng mở và đọc nội dung từ một file biên lai có định dạng .txt, sau đó hiển thị nội dung đó trong widget văn bản txtReceipt.

* Quy trình:

Yêu cầu người dùng chọn file:

Hiển thị một thông báo gợi ý rằng chỉ chọn file bắt đầu bằng Receipt\_\*.

Sử dụng filedialog.askopenfilename() để hiển thị hộp thoại chọn file, giới hạn:

Định dạng file: .txt.

Tất cả file: \*.\*.

Kiểm tra file được chọn:

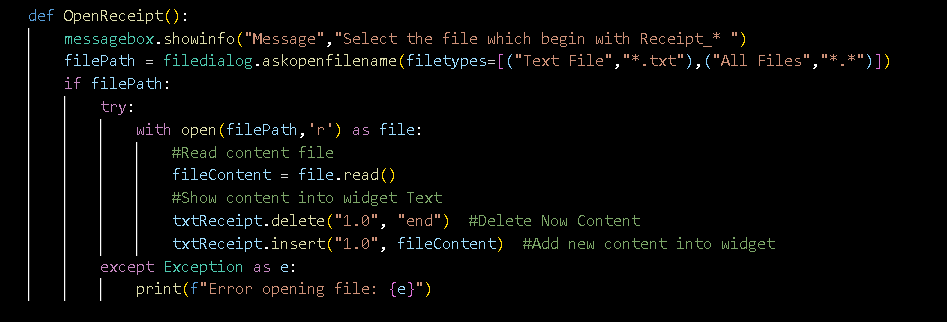
Nếu người dùng không chọn file (filePath là None hoặc rỗng), hàm sẽ kết thúc ngay lập tức.

Mở và đọc nội dung file:Mở file ở chế độ đọc ('r').Đọc toàn bộ nội dung file bằng file.read().Hiển thị nội dung trong widget:

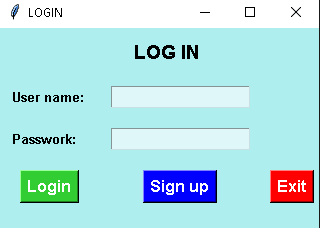
Xóa nội dung hiện tại của txtReceipt (nếu có) bằng txtReceipt.delete().

Chèn nội dung mới đọc từ file vào vị trí đầu tiên của widget bằng txtReceipt.insert().

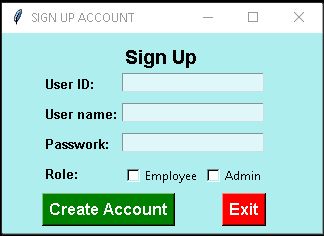
Xử lý lỗi: Nếu xảy ra lỗi trong quá trình mở hoặc đọc file (ví dụ: file bị hỏng hoặc không thể truy cập), thông báo lỗi sẽ được ghi ra console.

****

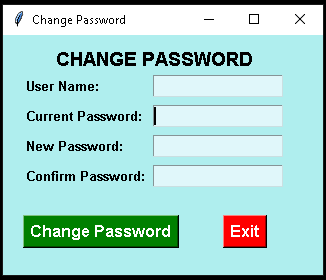
1. **Giao Diện**

****

**Hình 1: Giao diện đăng nhập**

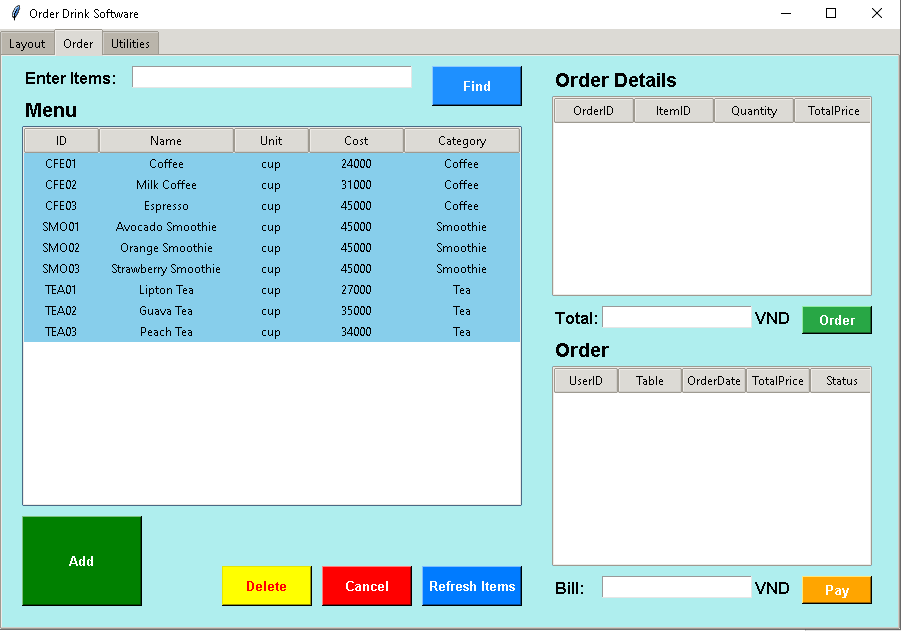
****

**Hình 2: Giao diện đăng ký**

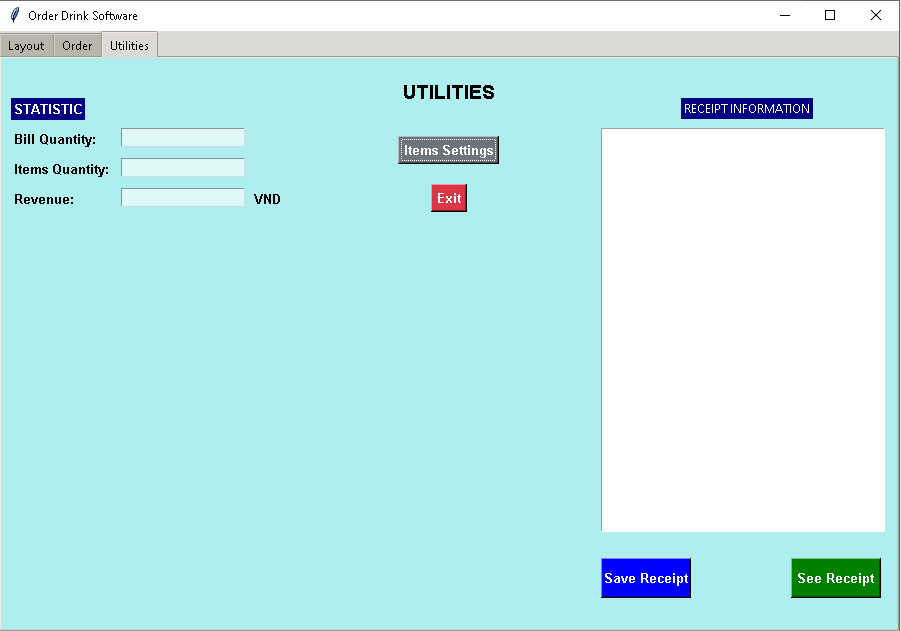
****

**Hình 3: Giao diện đổi mật khẩu**

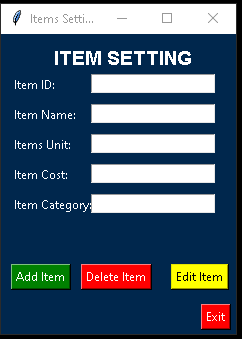
**Hình 4: Giao diện Sơ đồ**

****

**Hình 5: Giao diện Order**

****

**Hình 6: Giao diện tiện ích**

****

**Hình 7: Giao diện cài đặt danh mục đồ uống**

1. **TỰ ĐÁNH GIÁ**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tiêu chí** | **Tỷ lệ** | **Mô tả** | **Chi tiết** | **Điểm** |
| Quy trình phát triển PM | 30 | Hiểu được từng module và thể hiện được nội dung cơ bản lên báo cáo. | Quy trình phát triển phần mềm | 0.5 |
| Phần mềm hoàn thiện các tính năng, chất lượng tốt: viết câu từ dễ hiểu, ít chính tả, cô động, súc tích | 2.5 |
| Quá trình thực hiện | 40 | Đúng tiến độ yêu cầu, có cố gắng học hỏi sửa chữa và phát triển, chủ động khám phá và hoàn thiện sản phẩm. | Quá trình làm việc tốt (các buổi thực hành đáp ứng yêu cầu) | 1.0 |
| Sáng tạo phát triển thêm các tính năng khác | 2.0 |
| Có cố gắng quyết tâm, không bỏ cuộc, chủ động khám phá, tìm kiếm tài liệu, tham khảo có chọn lọc đạt hiệu quả. | 1.0 |
| Chất lượng báo cáo | 30 | Nội dung và hình thức báo cáo đạt chất lượng tốt, đáp ứng các yêu cầu đặt ra | Hình thức báo cáo đạt yêu cầu | 1.0 |
| Sản phẩm đáp ứng tốt yêu cầu thực tế. Chịu khó tự học code, Demo tốt. | 2.0 |

Tổng điểm : 6.0/10.0

1. **TÀI LIỆU THAM KHẢO**
2. **Màu sắc:** [1]
3. **Bcrypt:** [2]
4. **Python widget:** [3]
5. **Kết nối cơ sở dữ liệu Python:** [4]
6. **Lập trình giao diện Pyhton:** [5]
7. **So sánh attributes(”-topmost”,1) và lift():**[6]
8. **Hướng dẫn dùng Notebook, Frame, TreeView:** [7]
9. **Phần mềm Order nước MISA CukCuk:** phần mềm download trên CHPlay : CukCuk Sale

[1] “Mã màu xanh lam - màu xanh RGB.” Accessed: Dec. 09, 2024. [Online]. Available: https://www.rapidtables.org/vi/web/color/blue-color.html

[2] “Salted passwords and password hashing: How do they work? | NordVPN.” Accessed: Dec. 09, 2024. [Online]. Available: https://nordvpn.com/blog/what-are-salted-passwords-and-password-hashing/

[3] “Cách sử dụng về widget Treeview trong Tkinter.” Accessed: Dec. 09, 2024. [Online]. Available: https://freetuts.net/cach-su-dung-ve-widget-treeview-trong-tkinter-6792.html

[4] “Hướng dẫn lập trình cơ sở dữ liệu với Python,” ICANTECH. Accessed: Dec. 09, 2024. [Online]. Available: https://www.icantech.vn/kham-pha/lap-trinh-co-so-du-lieu-voi-python

[5] “Hướng dẫn lập trình giao diện GUI bằng Tkinter trong ngôn ngữ Python - VniTeach - Giáo viên 4.0.” Accessed: Dec. 09, 2024. [Online]. Available: https://www.vniteach.com/2022/02/05/huong-dan-lap-trinh-giao-dien-gui-bang-tkinter-trong-ngon-ngu-python/

[6] “python - What does the one in ’tk.wm\_attributes(‘-topmost’, 1) mean? - Stack Overflow.” Accessed: Dec. 09, 2024. [Online]. Available: https://stackoverflow.com/questions/65046290/what-does-the-one-in-tk-wm-attributes-topmost-1-mean

[7] “Python Tkinter placing frame and treeview within a notebook - Stack Overflow.” Accessed: Dec. 09, 2024. [Online]. Available: https://stackoverflow.com/questions/66109101/python-tkinter-placing-frame-and-treeview-within-a-notebook