def adicionar\_item\_saida():

nome\_produto = entrada\_nome\_qtde\_saida.get().strip()

qtde\_retirada = entrada\_qtde\_retirada.get().strip()

if not nome\_produto or not qtde\_retirada:

showinfo("Aviso", "Preencha todos os campos!")

return

try:

qtde = int(qtde\_retirada)

if qtde <= 0:

showinfo("Aviso", "A quantidade deve ser maior que zero!")

return

except ValueError:

showinfo("Aviso", "Quantidade inválida!")

return

# Adiciona o item à lista de saída

item\_texto = f"{nome\_produto} - -{qtde} unidades"

# Verifica se já existe na lista para atualizar a quantidade

for child in frame\_scroll4.winfo\_children():

if isinstance(child, customtkinter.CTkLabel) and nome\_produto in child.cget("text"):

existing\_text = child.cget("text")

existing\_qtde = int(existing\_text.split("-")[2].split()[0])

new\_qtde = existing\_qtde + qtde

child.configure(text=f"{nome\_produto} - -{new\_qtde} unidades")

break

else:

# Se não existir, cria novo item

x = len(frame\_scroll4.winfo\_children()) // 2 # Divide por 2 porque temos label + botão para cada item

label = customtkinter.CTkLabel(frame\_scroll4, text=item\_texto)

label.grid(row=x, column=0, pady=5, padx=0, sticky="w")

botao\_remover = customtkinter.CTkButton(frame\_scroll4, text="🗑️", width=5,

command=lambda l=label, b=None: remover\_item\_saida(l, b))

botao\_remover.grid(row=x, column=1, pady=5, padx=5)

# Limpa o campo de quantidade

entrada\_qtde\_retirada.delete(0, tk.END)

def remover\_item\_saida(label, botao):

label.destroy()

if botao:

botao.destr

# Coletar todos os itens da lista de saída

itens\_saida = []

for child in frame\_scroll4.winfo\_children():

if isinstance(child, customtkinter.CTkLabel):

texto = child.cget("text")

partes = texto.split(" - -")

nome = partes[0]

quantidade = int(partes[1].split()[0])

itens\_saida.append((nome, quantidade))

if not itens\_saida:

showinfo("Aviso", "Nenhum item para salvar!")

return

try:

conexao = sqlite3.connect("dados.db")

terminal\_sql = conexao.cursor()

data\_hora = datetime.now().strftime("%d/%m/%Y %H:%M")

for nome, quantidade in itens\_saida:

# Verifica se o produto existe

terminal\_sql.execute("SELECT nome FROM itens WHERE nome = ?", (nome,))

if not terminal\_sql.fetchone():

showinfo("Erro", f"Produto '{nome}' não cadastrado!")

continue

# Insere no banco de dados

terminal\_sql.execute(

"INSERT INTO saidas (nome\_produto, quantidade, data\_hora) VALUES (?, ?, ?)",

(nome, quantidade, data\_hora)

)

conexao.commit()

showinfo("Sucesso", "Saída de produtos salva com sucesso!")

# Limpa a lista de saída

for child in frame\_scroll4.winfo\_children():

child.destroy()

except Exception as e:

conexao.rollback()

showinfo("Erro", f"Ocorreu um erro ao salvar: {str(e)}")

finally:

conexao.close()

Principais melhorias:

1. Correção na função adicionar\_item\_saida() para:
   * Validar corretamente os dados de entrada
   * Atualizar quantidades se o produto já estiver na lista
   * Adicionar novos itens corretamente
2. Implementação da função salvar\_saida() para:
   * Coletar todos os itens da lista de saída
   * Validar se os produtos existem no banco de dados
   * Salvar no banco de dados com data/hora
   * Limpar a lista após o salvamento
3. Correção na função remover\_item\_saida() para:
   * Remover tanto o label quanto o botão associado

Para usar estas funções, certifique-se de que:

1. O botão "Adicionar item" na tela de saída esteja configurado para chamar adicionar\_item\_saida()
2. O botão "Salvar" na tela de saída esteja configurado para chamar salvar\_saida()
3. A tabela 'saidas' no banco de dados esteja criada corretamente com os campos:
   * nome\_produto (text)
   * quantidade (integer)
   * data\_hora (text)

Estas alterações devem resolver os problemas com a adição de itens na tela de saída e o salvamento no banco de dados.

New chat