Final Project

Super Shop Cashier application

"""

Name: Tanzilur Rahman

Student Number:200595789

Description:
This Super Shop application is a Python-based retail management system that streamlines customer purchases and backend operations. It allows users to scan product barcodes, calculate totals with tax, and generate receipts in Word format. For staff, it provides tools to add new products to the inventory and track daily sales. Transactions are logged in a text file for record-keeping, ensuring efficient and accurate management of shop activities.

"""

'''

All my imports

'''

```
import openpyxl
import pyinputplus as pyip
import random
import docx
import datetime
import subprocess
import os
import time
```

'''

all global veriables

'''

```
barcodes=[]
```

```
sheet = None  # Declare global sheet variable
myProductIndex=[]
myProducts=[]
totalPrice=0.0
logFile = "DailyTransactions.txt"

'''
In this section I'm storing my whole spreedsheet barcodes in to a list and tracking my last row
'''
productList=openpyxl.load_workbook("SuperShopItems.xlsx")
sheet=productList.active
for i in range(2,102):
   cell='D'+str(i)
   barcodes.append(int(sheet[cell].value))
lastRow = sheet.max_row
highestColumn=lastRow+1

"""
this is a interface asking that who will use it
"""
def askingForSelectingInterface():
   global logFile
   choosingInterface=pyip.inputMenu(['Start an order','Call for service','Quit'], numbered=True)
   if(choosingInterface=='Start an order'):
      result=customerBuying()
   elif(choosingInterface=='Call for service'):
      backendOptions=pyip.inputMenu(['Add A Product','Watch Total Sale','Quit'], numbered=True)
      if(backendOptions=='Add A Product'):
         addProduct()
```

```
        elif(backendOptions=='Watch Total Sale'):
            print(readTotalSales(logFile))
        elif(backendOptions=='Quit'):
            return
    elif(choosingInterface=='Quit'):
        return
    time.sleep(5)
    totalPrice=0.0
"""
```

Customers sections

```
"""

'''
in this section system asking to input the barcode of the product
and 4 additional option
1.Total will complete the transaction and will print recept in word file and in text file for company
2.Void is for delete if a user input a item mistakely
3. assist Mode id a customer want multiple same product it will make that easy tor user
4.quit means close the application
'''
def customerBuying():
    global myProducts
    global myProductIndex
    global totalSale
    productBarcode=''
```

```
  print("\n1.Total\n2.Void \n3.Assist Mode \n4.Quit Application ")
  while((productBarcode!=1)):
    productBarcode = pyip.inputInt(prompt = "Enter Item BarCode(3 digit only):", min = 1, lessThan
= 1000 )
    #adding to product list
    if(checkingProduct(productBarcode)):
      myProducts.append(productBarcode)
      myProductIndex.append(barcodes.index(productBarcode)+2)
      nameCell='B'+str(barcodes.index(productBarcode)+2)
      priceCell='C'+str(barcodes.index(productBarcode)+2)
      print(f"Product Name: {sheet[nameCell].value} | Price: {sheet[priceCell].value}")
    #for void
    elif(productBarcode==2):
      displayProductList()
      voidItem = pyip.inputInt(prompt = "Enter Void Item number:", min = 1, lessThan =
len(myProductIndex)+1 )
      voidIndexProduct=voidItem-1
      del myProducts[voidIndexProduct]
      del myProductIndex[voidIndexProduct]
      displayProductList()

    #for total
    elif(productBarcode==1):
      displayProductList()
      print(f"Total PRICE: ${productTotal():.2f}")

    # Assist Mode
    elif(productBarcode == 3):
      itemQuantity = pyip.inputInt(prompt="Enter Item quantity:", min=1)
```

```
        quantityItemBarcode = pyip.inputInt(prompt="Enter Item BarCode(3 digit only):", min=1, lessThan=1000)
        for i in range(itemQuantity):
          myProducts.append(quantityItemBarcode)
          myProductIndex.append(barcodes.index(quantityItemBarcode)+2)
        print(f"Added {itemQuantity} items with barcode {quantityItemBarcode} to the cart.")
        displayProductList()
        print(f"Total PRICE: ${productTotal():.2f}")
      #quit
      elif(productBarcode == 4):
        return
      else:
        print("Product Not Available")


   if(productBarcode==1):
        #asking which payment method user will use
        print(paymentMethod())
        #printing recipt in a word file
        recipt()
        #SAVING INFORMATION IN a text file
        logTransaction()
'''
asking user which payment method user will use
'''
def paymentMethod():
   paymentOption=pyip.inputMenu(['Debit/Credit','Cash','Gift Card','E-Scan Card'], numbered=True)
   return f"Thank You for paying via {paymentOption}."


"""
```

printing recipt in word file

"""

#Asking the user if user wants the recipt or not

```
def recipt():
    wantRecipt=pyip.inputYesNo(prompt="Do you want a recipt?(Y/N)")
    if(wantRecipt=='no'):
        print("Thank you so much.Please come back.")
    else:
        print("Printing Recipt")
        printInWordFile()
        print("Thank you so much.Please come back.")
    return
```

#if the user wants it then print in word file

```
def printInWordFile():
    doc = docx.Document()
    # Get the current date and time
    currentTime = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")

    # Replace invalid characters in filename (e.g., : with -)
    safeFilename = currentTime.replace(":", "-").replace("/", "-").replace(" ", "_")

    doc.add_heading("Super Shop Receipt", 1)
    doc.add_paragraph(f"Date and Time: {currentTime}")
    doc.add_paragraph("\nThank you for shopping with us!\n")
```

```
# Total Items section (underlined text)
underlineParagraph = doc.add_paragraph(f"Total Items Purchased: {len(myProducts)} \n \n")
underlineParagraph.runs[0].underline = True



receiptText = ""
count = 1
for row_index in myProductIndex:
   printProductName = sheet[f'B{row_index}'].value
   printProductPrice = sheet[f'C{row_index}'].value
   receiptText += f"{count}. {printProductName} | ${printProductPrice}\n"
   count += 1



# Add receipt text to the document
doc.add_paragraph(receiptText)

# Adding a total price section (bold text)
totalPriceParagraph = doc.add_paragraph(f"\nTotal Price (Including Tax): ${totalPrice:.2f}")
totalPriceParagraph.runs[0].bold = True

# Save the receipt
receiptPath = f"Receipt_({safeFilename}).docx"
doc.save(receiptPath)
print(f"Receipt saved as {receiptPath}")

openReceipt(receiptPath)
return
```

```
#after writing opening the word file
def openReceipt(receiptPath):

  # Ensure the file exists
  if os.path.exists(receiptPath):
    # Open the receipt using the default associated application
    try:
      subprocess.Popen(['start', receiptPath], shell=True)
      print("Receipt opened successfully.")
    except Exception as e:
      print(f"Failed to open the receipt: {e}")
  else:
    print("Receipt file does not exist.")

"""
Other small Fucntion used and called in other functions
"""
#checking if selected product available in the barcode list
def checkingProduct(checkBarcode):
  global barcodes
  return (checkBarcode in barcodes)
#printing all the product in the cart
def displayProductList():
  count=1
  for i in myProductIndex:
    print(f"{count}. Product Name: {sheet['B' + str(i)].value} | Price: ${sheet['C' + str(i)].value} ||
Barcode: {sheet['D' + str(i)].value}")
    count=count+1
```

```
    return 0
#calculating the product total price
def productTotal():
    global totalPrice
    for i in myProductIndex:
        totalPrice=totalPrice+ float(sheet['C' + str(i)].value)
    totalPrice=totalPrice+totalPrice*.13
    return totalPrice



#backend interface
"""
For Company here is 2 features
1. devoloper can add product to the spreedsheet
2. they can see how much tracsaction happend so far
*** there is another fuction its take all the information after 1 successful transaction the system will write the details
of the transaction in a tex file and folling transaction will be saved in the same file
and there will record total transactions

"""

'''
In this function the system will ask the name barcode price product type and here i used python input Plus
so it will validate itself and after that it will pass the information in to the spreedsheet
'''
def addProduct():
```

```
    global highestColumn
    global sheet
    global productList
    global highestColumn
    newProductBarcode=0
    # Product Name (ensures a non-empty string)
    newProductName = pyip.inputStr(prompt="Enter product name: ", allowRegexes=[r'.+'], blockRegexes=[r'^\s*$'], default="productName")
    # Product Barcode (3-digit integer)
    newProductBarcode = pyip.inputInt(prompt="Enter new Item Barcode (3 digits only): ", min=100, lessThan=1000, default=000)
    while not(checkingProduct(newProductBarcode)==False):
        newProductBarcode = pyip.inputInt(prompt="Enter new Item Barcode (3 digits only): ", min=100, lessThan=1000, default=000)
        checkingProduct(newProductBarcode)
    # Product Price (in XX.XX format)
    newProductPrice = pyip.inputRegex(r'^\d{1,2}\.\d{2}$', prompt="Enter a price in the format XX.XX: ", default=00.00)
    # Product Type (ensures a non-empty string)
    newProductType = pyip.inputStr(prompt="Enter product type: ", allowRegexes=[r'.+'], blockRegexes=[r'^\s*$'], default="productType")


    print(f"{highestColumn}. Product Name: {newProductName} | Price: ${newProductPrice} || Barcode: {newProductBarcode} | Product Type: {newProductType}")
    isEntered=pyip.inputYesNo(prompt="Do you wantto input the item into the spreedsheet?(Y/N)")
    if not(isEntered=='no'):
        sheet['A' + str(highestColumn)]=highestColumn
        sheet['B' + str(highestColumn)]=newProductName
        sheet['C' + str(highestColumn)]=newProductPrice
        sheet['D' + str(highestColumn)]=str(newProductBarcode)
        sheet['E' + str(highestColumn)]=newProductType
```

```
        productList.save("SuperShopItems.xlsx")
        print("New Product Entered in the Spreedsheet")

    else:
        print("No Product Entered in to the Spreedsheet")
    return

#it will read the total sell happend so far
def readTotalSales(logFile):
    try:
        file = open(logFile, "r")
        firstLine = file.readline().strip()  # Read the first line
        file.close()

        if "Current Total Daily Sales:" in firstLine:
            return float(firstLine.split(": $")[1])  # Extract and return the total sales as a float
        else:
            return 0.0  # If the first line doesn't contain the total sales, return 0
    except FileNotFoundError:
        print("Log file not found. Starting fresh.")
        return 0.0  # Return 0 if the file doesn't exist
    except Exception as e:
        print(f"An error occurred: {e}")
        return 0.0  # Return 0 for any other errors


# this fuction will save every tansaction in a text file

def logTransaction():
```

```
  currentTime = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")
  global logFile
  global totalPrice
  try:
    # Log daily total sales and append transaction
    totalPriceForTransaction =totalPrice
    currentTotal = logDailyTotalSales(logFile, totalPriceForTransaction)

    # Open file in append mode
    file = open(logFile, "a")
    file.write("\n=======================================\n")
    file.write(f"Transaction Time: {currentTime}\n")
    file.write(f"Total Items: {len(myProducts)}\n")
    file.write("Item Details:\n")

    for i in range(len(myProductIndex)):
      productName = sheet[f'B{myProductIndex[i]}'].value
      productBarcode = sheet[f'D{myProductIndex[i]}'].value
      productPrice = sheet[f'C{myProductIndex[i]}'].value
      file.write(f" {i + 1}. {productName} | Barcode: {productBarcode} | Price: ${productPrice:.2f}\n")

    file.write(f"Total Price (Including Tax): ${totalPriceForTransaction:.2f}\n")
    file.write("=======================================\n")
    file.close()

  except Exception as e:
    print(f"An error occurred: {e}")

"""
```

```
Here it will search in to the txt file for total sell after finding it it will
take the ammount add the last transaction into it and print the whole thing again
"""

def logDailyTotalSales(logFile, transactionTotal):
    currentTotal = 0.0

    try:
        # Read current total from file, if exists
        try:
            file = open(logFile, "r")
            firstLine = file.readline().strip()
            if "Current Total Daily Sales:" in firstLine:
                currentTotal = float(firstLine.split(": $")[1])
            file.close()
        except (FileNotFoundError, ValueError, IndexError):
            currentTotal = 0.0

        #Update total
        currentTotal += transactionTotal

       #Read the existing content of the file
        file = open(logFile, "r")
        lines = file.readlines()  # Read all lines into a list
        file.close()

        # If the file is empty, initialize it with the current total sales
        if not lines:
            lines.append(f"Current Total Daily Sales: ${currentTotal:.2f}\n")
```

```
        else:
            # Modifying the first line
            lines[0] = f"Current Total Daily Sales: ${currentTotal:.2f}\n"

        #Write the updated content back to the file (overwriting it)
        file = open(logFile, "w")
        file.writelines(lines)
        file.close()

    except Exception as e:
        print(f"An error occurred while updating daily total sales: {e}")

    return currentTotal


#Starting the system
askingForSelectingInterface()
```